INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

(Approved by AICTE | NAAC Accreditation with 'A' Grade | Accredited by NBA | Affiliated to JNTUH)

Dundigal, Hyderabad - 500 043, Telangana

OUTCOME BASED EDUCATION
WITH
CHOICE BASED CREDIT SYSTEM

BACHELOR OF TECHNOLOGY
ELECTRONICS AND COMMUNICATION ENGINEERING

ACADEMIC REGULATIONS, COURSE STRUCTURE AND SYLLABI
UG20

B.Tech Regular Four Year Degree Program
(for the batches admitted from the academic year 2020 - 2021)
&

B.Tech (Lateral Entry Scheme)
(for the batches admitted from the academic year 2021 - 2022)

These rules and regulations may be altered/changed from time to time by the academic council

FAILURE TO READ AND UNDERSTAND THE RULES IS NOT AN EXCUSE

VISION

To bring forth professionally competent and socially sensitive engineers, capable of working across cultures meeting the global standards ethically.

MISSION

To provide students with an extensive and exceptional education that prepares them to excel in their profession, guided by dynamic intellectual community and be able to face the technically complex world with creative leadership qualities.

Further, be instrumental in emanating new knowledge through innovative research that emboldens entrepreneurship and economic development for the benefit of wide spread community.

QUALITY POLICY

Our policy is to nurture and build diligent and dedicated community of engineers providing a professional and unprejudiced environment, thus justifying the purpose of teaching and satisfying the stake holders.

A team of well qualified and experienced professionals ensure quality education with its practical application in all areas of the Institute.

PROGRAM OUTCOMES (PO's)

Engineering Graduates will be able to:

PO1: Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-Long Learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

CONTENTS

“Take up one idea.
Make that one idea your life-think of it, dream of it, live on that idea.
Let the brain muscles, nerves, every part of your body be full of that idea and just leave every other idea alone. This is the way to success”

Swami Vivekananda

PRELIMINARY DEFINITIONS AND NOMENCLATURES

AICTE: Means All India Council for Technical Education, New Delhi.

Autonomous Institute: Means an institute designated as Autonomous by University Grants Commission (UGC), New Delhi in concurrence with affiliating University (Jawaharlal Nehru Technological University, Hyderabad) and State Government.

Academic Autonomy: Means freedom to an institute in all aspects of conducting its academic programs, granted by UGC for Promoting Excellence.

Academic Council: The Academic Council is the highest academic body of the institute and is responsible for the maintenance of standards of instruction, education and examination within the institute. Academic Council is an authority as per UGC regulations and it has the right to take decisions on all academic matters including academic research.

Academic Year: It is the period necessary to complete an actual course of study within a year. It comprises two main semesters i.e., (one odd + one even) and one supplementary semester.

Branch: Means specialization in a program like B.Tech degree program in Aeronautical Engineering, B.Tech degree program in Computer Science and Engineering etc.

Board of Studies (BOS): BOS is an authority as defined in UGC regulations, constituted by Head of the Organization for each of the departments separately. They are responsible for curriculum design and updation in respect of all the programs offered by a department.

Backlog Course: A course is considered to be a backlog course, if the student has obtained a failure grade (F) in that course.

Basic Sciences: The courses offered in the areas of Mathematics, Physics, Chemistry etc., are considered to be foundational in nature.

Betterment: Betterment is a way that contributes towards improvement of the students’ grade in any course(s). It can be done by either (a) re-appearing or (b) re-registering for the course.

Commission: Means University Grants Commission (UGC), New Delhi.

Choice Based Credit System: The credit based semester system is one which provides flexibility in designing curriculum and assigning credits based on the course content and hours of teaching along with provision of choice for the student in the course selection.

Certificate Course: It is a course that makes a student to have hands-on expertise and skills required for holistic development in a specific area/field.

Compulsory course: Course required to be undertaken for the award of the degree as per the program.

Continuous Internal Examination: It is an examination conducted towards sessional assessment.

Core: The courses that are essential constituents of each engineering discipline are categorized as professional core courses for that discipline.

Course: A course is a subject offered by a department for learning in a particular semester.

Course Outcomes: The essential skills that need to be acquired by every student through a course.

Credit: A credit is a unit that gives weight to the value, level or time requirements of an academic course. The number of 'Contact Hours' in a week of a particular course determines its credit value. One credit is equivalent to one lecture/tutorial hour per week.

Credit point: It is the product of grade point and number of credits for a course.

Cumulative Grade Point Average (CGPA): It is a measure of cumulative performance of a student over all the completed semesters. The CGPA is the ratio of total credit points secured by a student in various courses in all semesters and the sum of the total credits of all courses in all the semesters. It is expressed up to two decimal places.

Curriculum: Curriculum incorporates the planned interaction of students with instructional content, materials, resources, and processes for evaluating the attainment of Program Educational Objectives.

Department: An academic entity that conducts relevant curricular and co-curricular activities, involving both teaching and non-teaching staff, and other resources in the process of study for a degree.

Detention in a Course: Student who does not obtain minimum prescribed attendance in a course shall be detained in that particular course.

Dropping from Semester: Student who doesn’t want to register for any semester can apply in writing in prescribed format before the commencement of that semester.

Elective Course: A course that can be chosen from a set of courses. An elective can be Professional Elective and / or Open Elective.

Evaluation: Evaluation is the process of judging the academic performance of the student in her/his courses. It is done through a combination of continuous internal assessment and semester end examinations.

Experiential Engineering Education (ExEEd): Engineering entrepreneurship requires strong technical skills in engineering design and computation with key business skills from marketing to business model generation. Our students require sufficient skills to innovate in existing companies or create their own.

Grade: It is an index of the performance of the students in a said course. Grades are indicated by alphabets.

Grade Point: It is a numerical weight allotted to each letter grade on a 10 - point scale.

Honours: An Honours degree typically refers to a higher level of academic achievement at an undergraduate level.

Institute: Means Institute of Aeronautical Engineering, Hyderabad unless indicated otherwise by the context.

Massive Open Online Courses (MOOC): MOOC courses inculcate the habit of self learning. MOOC courses would be additional choices in all the elective group courses.

Minor: Minor are coherent sequences of courses which may be taken in addition to the courses required for the B.Tech degree.

Pre-requisite: A specific course or subject, the knowledge of which is required to complete before student register another course at the next grade level.

Professional Elective: It indicates a course that is discipline centric. An appropriate choice of minimum number of such electives as specified in the program will lead to a degree with specialization.

Program: Means, UG degree program: Bachelor of Technology (B.Tech); PG degree program: Master of Technology (M.Tech) / Master of Business Administration (MBA).

Program Educational Objectives: The broad career, professional and personal goals that every student will achieve through a strategic and sequential action plan.

Project work: It is a design or research based work to be taken up by a student during his/her final year to achieve a particular aim. It is a credit based course and is to be planned carefully by the student.

Re-Appearing: A student can reappear only in the semester end examination for theory component of a course, subject to the regulations contained herein.

Registration: Process of enrolling into a set of courses in a semester of a program.

Regulations: The regulations, common to all B.Tech programs offered by Institute, are designated as "IARE Regulations – R20" and are binding on all the stakeholders.

Semester: It is a period of study consisting of 15 to 18 weeks of academic work equivalent to normally 90 working days. Odd semester commences usually in July and even semester in December of every year.

Semester End Examinations: It is an examination conducted for all courses offered in a semester at the end of the semester.

S/he: Means "she" and "he" both.

Student Outcomes: The essential skill sets that need to be acquired by every student during her/his program of study. These skill sets are in the areas of employability, entrepreneurial, social and behavioral.

University: Means Jawaharlal Nehru Technological University Hyderabad (JNTUH), Hyderabad, is an affiliating University.

Withdraw from a Course: Withdrawing from a course means that a student can drop from a course within the first two weeks of odd or even semester (deadlines are different for summer sessions). However, s/he can choose a substitute course in place of it, by exercising the option within 5 working days from the date of withdrawal.

FOREWORD

The autonomy is conferred to Institute of Aeronautical Engineering (IARE), Hyderabad by University Grants Commission (UGC), New Delhi based on its performance as well as future commitment and competency to impart quality education. It is a mark of its ability to function independently in accordance with the set norms of the monitoring bodies including J N T University Hyderabad (JNTUH), Hyderabad and AICTE, New Delhi. It reflects the confidence of the affiliating University in the autonomous institution to uphold and maintain standards it expects to deliver on its own behalf. Thus, an autonomous institution is given the freedom to have its own **curriculum, examination system** and **monitoring mechanism**, independent of the affiliating University but under its observance.

IARE is proud to win the credence of all the above bodies monitoring the quality in education and has gladly accepted the responsibility of sustaining, if not improving upon the standards and ethics for which it has been striving for more than a decade in reaching its present standing in the arena of contemporary technical education. As a follow up, statutory bodies such as Academic Council and Board of Studies (BOS) are constituted with the guidance of the Governing Body of the institute and recommendations of the JNTUH to frame the regulations, course structure, and syllabi under autonomous status.

The autonomous regulations, course structure, and syllabi have been prepared after prolonged and detailed interaction with several expertise solicited from academics, industry and research, in accordance with the vision and mission of the institute in order to produce a quality engineering graduate to the society.

All the faculty, parents, and students are requested to go through all the rules and regulations carefully. Any clarifications needed are to be sought at appropriate time and from the principal of the institute, without presumptions, to avoid unwanted subsequent inconveniences and embarrassments. The cooperation of all the stake holders is requested for the successful implementation of the autonomous system in the larger interests of the institute and brighter prospects of engineering graduates.

PRINCIPAL

INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

ACADEMIC REGULATIONS – UG.20

B.Tech. Regular Four Year Degree Program
(for the batches admitted from the academic year 2020 - 2021)
&
B.Tech. (Lateral Entry Scheme)
(for the batches admitted from the academic year 2021 - 2022)

For pursuing four year undergraduate Bachelor of Technology (B.Tech) degree program of study in engineering offered by Institute of Aeronautical Engineering under Autonomous status.

A student shall undergo the prescribed courses as given in the program curriculum to obtain his/her degree in major in which he/she is admitted with 160 credits in the entire program of 4 years. Additional 20 credits can be acquired for the degree of B.Tech with **Honours or additional Minor in Engineering**. These additional 20 credits will have to be acquired with massive open online courses (MOOCs), to tap the zeal and excitement of learning beyond the classrooms. This creates an excellent opportunity for students to acquire the necessary skill set for employability through massive open online courses where the rare expertise of world famous experts from academics and industry are available.

Separate certificate will be issued in addition to major degree program mentioning that the student has cleared Honours / Minor specialization in respective courses.

1. CHOICE BASED CREDIT SYSTEM

The credit based semester system provides flexibility in designing program curriculum and assigning credits based on the course content and hours of teaching. The Choice Based Credit System (CBCS) provides a 'cafeteria' type approach in which the students can take courses of their choice, learn at their own pace, undergo additional courses and acquire more than the required credits, and adopt an interdisciplinary approach to learning.

A course defines learning objectives and learning outcomes and comprises lectures / tutorials / laboratory work / field work / project work / comprehensive examination / seminars / assignments / MOOCs / alternative assessment tools / presentations / self-study etc., or a combination of some of these. Under the CBCS, the requirement for awarding a degree is prescribed in terms of number of credits to be completed by the students.

2. MEDIUM OF INSTRUCTION

The medium of instruction shall be **English** for all courses, examinations, seminar presentations and project work. The program curriculum will comprise courses of study as given in course structure, in accordance with the prescribed syllabi.

3. PROGRAMS OFFERED

Presently, the institute is offering Bachelor of Technology (B.Tech) degree programs in eleven disciplines. The various programs and their two-letter unique codes are given in Table 1.

Table 1: B.Tech Programs offered

S. No	Name of the Program	Title	Code
1	Aeronautical Engineering	AE	07
2	Computer Science and Engineering	CS	05
3	Computer Science and Engineering (AI & ML)	CA	34
4	Computer Science and Engineering (Data Science)	CD	35
5	Computer Science and Engineering (Cyber Security)	CC	36
6	Computer Science and Information Technology	CI	37
7	Information Technology	IT	06
8	Electronics and Communication Engineering	EC	04
9	Electrical and Electronics Engineering	EE	02
10	Mechanical Engineering	ME	03
11	Civil Engineering	CE	01

4. SEMESTER STRUCTURE

Each academic year is divided into three semesters, TWO being **MAIN SEMESTERS** (one odd + one even) and ONE being a **SUPPLEMENTARY SEMESTER**. Main semesters are for regular class work. Supplementary Semester is primarily for failed students i.e. registration for a course for the first time is generally not permitted in the supplementary semester.

4.1 Each main semester shall be of 21 weeks (Table 1) duration and this period includes time for registration of courses, course work, examination preparation, and conduct of examinations.

4.2 Each main semester shall have a minimum of 90 working days; out of which 75 days are for teaching / practical and 15 days for conduct of exams and preparation.

4.3 The supplementary semester shall be a fast track semester consisting of eight weeks and this period includes time for registration of courses, course work, and examination preparation, conduct of examinations, assessment, and declaration of final results.

4.4 All subjects may not be offered in the supplementary semester. The student has to pay a stipulated fee prescribed by the institute to register for a course in the supplementary semester. The supplementary semester is provided to help the student in not losing an academic year. It is optional for a student to make use of supplementary semester. **Supplementary semester is a special semester and the student cannot demand it as a matter of right** and will be offered based on availability of faculty and other institute resources.

4.5 The institute may use **supplementary semester** to arrange add-on courses for regular students and / or for deputing them for practical training / FSI model. A student can register for a maximum number of 15 credits during a supplementary semester.

The registration for the supplementary semester (during May – July, every year) provides an opportunity to students to clear their backlogs (‘F’ grade) or who are prevented from appearing for SEE examinations due to shortage of attendance less than 65% in each course (‘SA’ Grade) in the earlier semesters or the courses which he / she could not register (Drop / Withdraw) due to any reason.

Students will not be permitted to register for more than 15 credits (both I and II semester) in the supplementary semester. Students required to register for supplementary semester courses are to pay a nominal fee within the stipulated time. A separate circular shall be issued at the time of supplementary semester.

It will be optional for a student to get registered in the course(s) of supplementary semester; otherwise, he / she can opt to appear directly in supplementary examination. However, if a student gets registered in a course of supplementary semester, then it will be compulsory for a student to fulfill attendance criterion (≥90%) of supplementary semester and he / she will lose option to appear in immediate supplementary examination.

The students who have earlier taken SEE examination and register afresh for the supplementary semester may revoke the CIA marks secured by them in their regular/earlier attempts in the same course. Once revoked, the students shall not seek restoration of the CIA marks.

Supplementary semester will be at an accelerated pace e.g. one credit of a course shall require two hours/week so that the total number of contact hours can be maintained same as in normal semester.

Instructions and guidelines for the supplementary semester course:

- A minimum of 36 to 40 hours will be taught by the faculty for every course.
- Only the students registered and having sufficient percentage of attendance for the course will be permitted to write the examination.
- The assessment procedure in a supplementary semester course will be similar to the procedure for a regular semester course.
- Student shall register for the supplementary semester as per the schedule given in academic calendar.
- Once registered, students will not be allowed to withdraw from supplementary semester.

4.6 The academic calendar shown in Table 2 is declared at the beginning of the academic year.

Table 2: Academic Calendar

FIRST SEMESTER (21 weeks)	I Spell Instruction Period	8 weeks	19 weeks
	I Continuous Internal Assessment Examinations (Mid-term)	1 week	
	II Spell Instruction Period	8 weeks	
	II Continuous Internal Assessment Examinations (Mid-term)	1 week	
	Preparation and Practical Examinations	1 week	
	Semester End Examinations		2 weeks
Semester Break and Supplementary Exams			2 weeks
SECOND SEMESTER (21 weeks)	I Spell Instruction Period	8 weeks	19 weeks
	I Continuous Internal Assessment Examinations (Mid-term)	1 week	
	II Spell Instruction Period	8 weeks	
	II Continuous Internal Assessment Examinations (Mid-term)	1 week	
	Preparation & Practical Examinations	1 week	
	Semester End Examinations		2 weeks
Summer Vacation, Supplementary Semester and Remedial Exams			8 weeks

4.7 Students admitted on transfer from JNTUH affiliated institutes, Universities and other institutes in the subjects in which they are required to earn credits so as to be on par with regular students as prescribed by concerned 'Board of Studies'.

5.0 REGISTRATION / DROPPING / WITHDRAWAL

The academic calendar includes important academic activities to assist the students and the faculty. These include, dates assigned for registration of courses, dropping of courses and withdrawal from courses. This enables the students to be well prepared and take full advantage of the flexibility provided by the credit system.

5.1. Each student has to compulsorily register for course work at the beginning of each semester as per the schedule mentioned in the Academic Calendar. It is compulsory for the student to register for courses in time. The registration will be organized departmentally under the supervision of the Head of the Department.

5.2. In ABSENTIA, registration will not be permitted under any circumstances.

5.3. At the time of registration, students should have cleared all the dues of Institute and Hostel for the previous semesters, paid the prescribed fees for the current semester and not been debarred from the institute for a specified period on disciplinary or any other ground.

5.4. In the first two semesters, the prescribed course load per semester is fixed and is mandated to registered all courses. Withdrawal / dropping of courses in the first and second semester is not allowed.

5.5. In higher semesters, the average load is 22 credits / semester, with its minimum and maximum limits being set at 16 and 28 credits. This flexibility enables students (**from IV semester onwards**) to cope-up with the course work considering the academic strength and capability of student.

5.6. Dropping of Courses:

Within one week after the last date of first internal assessment test or by the date notified in the academic calendar, the student may in consultation with his / her faculty mentor/adviser, drop one or more courses without prejudice to the minimum number of credits as specified in clause 5.4. The dropped courses are not recorded in the memorandum of grades. Student must complete the dropped subject by registering in the supplementary semester / forthcoming semester in order to earn the required credits. Student must complete the dropped subject by registering in the supplementary semester / forthcoming semester in order to earn the required credits.

5.7. Withdrawal from Courses:

A student is permitted to withdraw from a course by the date notified in the academic calendar. Such withdrawals will be permitted without prejudice to the minimum number of credits as specified in clause 5.4. A student cannot withdraw a course more than once and withdrawal of reregistered courses is not permitted.

6.0 CREDIT SYSTEM

The B.Tech Program shall consist of a number of courses and each course shall be assigned with credits. The curriculum shall comprise Theory Courses, Elective Courses, Laboratory Courses, Value Added Courses, Mandatory Courses, Experiential Engineering Education (ExEEd), Internship and Project work.

Depending on the complexity and volume of the course, the number of contact periods per week will be assigned. Each theory and laboratory course carries credits based on the number of hours / week.

- Contact classes (Theory): 1 credit per lecture hour per week, 1 credit per tutorial hour per week.
- Laboratory hours (Practical): 1 credit for 2 practical hours per week.
- Project work: 1 credit for 2 hours of project work per week.
- Mandatory Courses: No credit is awarded.
- Value Added Courses: No credit is awarded.
- Experiential Engineering Education (ExEEd): 1 credit for two per hours.

Credit distribution for courses offered is given in Table 5.

Table 5: Credit distribution

S. No	Course	Hours	Credits
1	Theory Course	2 / 3 / 4	2 / 3 / 4
2	Elective Courses	3	3

3	Laboratory Courses	2 / 3 / 4	1 / 1.5 / 2
4	Mandatory Course / Value Added Course	-	0
5	Project Work	-	10
6	Full Semester Internship (FSI) Project work	-	10

Major benefits of adopting the credit system are listed below:

- Quantification and uniformity in the listing of courses for all programs at College, like core, electives and project work.
- Ease of allocation of courses under different heads by using their credits to meet national /international practices in technical education.
- Convenience to specify the minimum / maximum limits of course load and its average per semester in the form of credits to be earned by a student.
- Flexibility in program duration for students by enabling them to pace their course load within minimum/maximum limits based on their preparation and capabilities.
- Wider choice of courses available from any department of the same College or even from other similar Colleges, either for credit or for audit.
- Improved facility for students to optimize their learning by availing of transfer of credits earned by them from one College to another.

7.0 CURRICULAR COMPONENTS

Courses in a curriculum may be of three kinds: **Foundation / Skill, Core and Elective Courses.**

Foundation / Skill Course:

Foundation courses are the courses based upon the content leads to enhancement of skill and knowledge as well as value based and are aimed at man making education. Skill subjects are those areas in which one needs to develop a set of skills to learn anything at all. They are fundamental to learning any subject.

Professional Core Courses:

There may be a core course in every semester. This is the course which is to be compulsorily studied by a student as a core requirement to complete the requirement of a program in the said discipline of study.

Elective Course:

Electives provide breadth of experience in respective branch and application areas. Elective course is a course which can be chosen from a pool of courses. It may be:

- Supportive to the discipline of study
- Providing an expanded scope
- Enabling an exposure to some other discipline / domain
- Nurturing student’s proficiency / skill.

An elective may be Professional Elective, is a discipline centric focusing on those courses which add generic proficiency to the students or may be Open Elective, chosen from unrelated disciplines.

There are six professional elective tracks; students can choose not more than two courses from each track. Overall, students can opt for six professional elective courses which suit their project work in consultation with the faculty advisor/mentor. Nevertheless, one course from each of the four open electives has to be selected. A student may also opt for more elective courses in his/her area of interest.

Every course of the B.Tech program will be placed in one of the eight categories with minimum credits as listed in the Table 6.

Table 6: Category Wise Distribution of Credits

S. No	Category	Breakup of Credits
1	Humanities and Social Sciences (HSMC), including Management.	6
2	Basic Science Courses (BSC) including Mathematics, Physics and Chemistry.	18.5
3	Engineering Science Courses (ESC), including Workshop, Drawing, ExEEd, Basics of Electrical / Electronics / Mechanical / Computer Engineering.	20.5
4	Professional Core Courses (PCC), relevant to the chosen specialization / branch.	78
5	Professional Electives Courses (PEC), relevant to the chosen specialization / branch.	18
6	Open Elective Courses (OEC), from other technical and/or emerging subject areas.	09
7	Project work (PROJ) / Full Semester Internship (FSI) Project work	10
8	Mandatory Courses (MC) / Value Added Courses (VAC).	Non-Credit
TOTAL		**160**

Semester wise course break-up

Following are the **TWO** models of course structure out of which any student shall choose or will be allotted with one model based on their academic performance.

i. Full Semester Internship (FSI) Model and

ii. Non Full Semester Internship (NFSI) Model

In the FSI Model, out of the selected students - half of students shall undergo Full Semester Internship in VII semester and the remaining students in VIII semester. In the Non-FSI Model, all the selected students shall carry out the course work and Project work as specified in the course structure. A student who secures a minimum CGPA of 7.5 upto IV semester with **no current arrears** and maintains the CGPA of 7.5 till VI Semester shall be eligible to opt for FSI.

8. EVALUATION METHODOLOGY

Each theory course will be evaluated for a total of 100 marks, with 30 marks for Continuous Internal Assessment (CIA) and 70 marks for Semester End Examination (SEE). Student's performance in a course shall be judged by taking into account the results of CIA and SEE together. Table-7 shows the typical distribution of weightage for CIA and SEE.

Table 7: Assessment pattern for Theory Courses

Component		Marks	Total Marks
CIA	Continuous Internal Examination – 1 (Mid-term)	10	30
	Continuous Internal Examination – 2 (End-term)	10	
	Tech talk / Quiz – 1 and Quiz – 2	5	
	Concept video / Alternative Assessment Tool (AAT)	5	
SEE	Semester End Examination (SEE)	70	70
Total Marks			**100**

8.1. Semester End Examination (SEE):

The SEE is conducted for 70 marks of 3 hours duration. The syllabus for the theory courses is divided into FIVE modules and each modules carries equal weightage in terms of marks distribution. The question paper pattern is as follows.

Two full questions with ‘either’ ‘or’ choice will be drawn from each module. Each question carries 14 marks. There could be a maximum of two sub divisions in a question.

The emphasis on the questions is broadly based on the following criteria:

50 %	To test the objectiveness of the concept
50 %	To test the analytical skill of the concept OR to test the application skill of the concept

8.1. Continuous Internal Assessment (CIA):

For each theory course the CIA shall be conducted by the faculty / teacher handling the course. CIA is conducted for a total of 30 marks, with 20 marks for Continuous Internal Examination (CIE), 05 marks for Quiz and 05 marks for Alternative Assessment Tool (AAT). **Two CIE Tests are Compulsory** and sum of the two tests, along with the scores obtained in the quizzes (average of Quiz – 1 and Quiz – 2) / AAT shall be considered for computing the final CIA of a student in a given course.

The CIE Tests/quizzes/AAT shall be conducted by the course faculty with due approval from the HOD. Advance notification for the conduction of Quiz/AAT is mandatory and the responsibility lies with the concerned course faculty.

8.1.1. Continuous Internal Examination (CIE):

Two CIE exams shall be conducted at the end of the 8th and 16th week of the semester respectively for 10 marks each of 2 hours duration consisting of five descriptive type questions out of which four questions have to be answered. The valuation and verification of answer scripts of CIE exams shall be completed within a week after the conduct of the Examination.

8.1.2. Quiz – Online Examination

Two Quiz exams shall be conducted along with CIE in online mode for 5 marks each, consisting of 10 short answers questions (Definitions and Terminology) and 10 multiple choice questions (having each question to be answered by tick marking the correct answer from the choices (commonly four) given against it. Such a question paper shall be useful in testing of knowledge, skills, application, analysis, evaluation and understanding of the students. Average of two quiz examinations shall be considered.

8.1.3. Alternative Assessment Tool (AAT)

In order to encourage innovative methods while delivering a course, the faculty members are encouraged to use the Alternative Assessment Tool (AAT). This AAT enables faculty to design own assessment patterns during the CIA. The AAT enhances the autonomy (freedom and flexibility) of individual faculty and enables them to create innovative pedagogical practices. If properly applied, the AAT converts the classroom into an effective learning centre.

The AAT may include tech talk, tutorial hours/classes, seminars, assignments, term paper, open ended experiments, concept videos, partial reproduction of research work, oral presentation of research work, developing a generic tool-box for problem solving, report based on participation in create-a-thon, make-a-thon, code-a-thon, hack-a-thon conducted by reputed organizations / any other. etc.

However, it is mandatory for a faculty to obtain prior permission from the concerned HOD and spell out the teaching/assessment pattern of the AAT prior to commencement of the classes.

8.2 Laboratory Course

Each laboratory will be evaluated for a total of 100 marks consisting of 30 marks for internal assessment and 70 marks for semester end lab examination. Out of 30 marks of internal assessment, continuous lab assessment will be done for 20 marks for the day to day performance and 10 marks for the final internal lab assessment. The semester end laboratory examination for 70 marks shall be conducted internally by the respective department with at least two faculty members as examiners, both nominated by the Principal from the panel of experts recommended by the Chairman, BOS.

All the drawing related courses are evaluated in line with laboratory courses. The distribution shall be 30 marks for internal evaluation (20 marks for day–to–day work, and 10 marks for internal tests) and 70 marks for semester end laboratory examination. There shall be ONE internal test of 10 marks in each semester.

8.3 Audit Courses

In Addition, a student can register for courses for audit only with a view to supplement his/her knowledge and/or skills. Here also, the student's grades shall have to be reflected in the Memorandum of Grades. But, these shall not be taken into account in determining the student's academic performance in the semester. In view of this, it shall not be necessary for the institute to issue any separate transcript covering the audit courses to the registrants at these courses. Its result shall be declared as "Satisfactory" or "Not Satisfactory" performance.

8.4 Mandatory Courses (MC)

These courses are among the compulsory courses but will not carry any credits. However, a pass in each such course during the program shall be necessary requirement for the student to qualify for the award of Degree. Its result shall be declared as "Satisfactory" or "Not Satisfactory" performance.

8.5 Additional Mandatory Courses for lateral entry B.Tech students

In addition to the non-credit mandatory courses for regular B.Tech students, the lateral entry students shall take up the following three non-credit mandatory bridge courses (one in III semester, one in IV semester and one in V semester) as listed in Table 8. The student shall pass the following non-credit mandatory courses for the award of the degree and must clear these bridge courses before advancing to the VII semester of the program.

Table-8: Additional Mandatory Courses for lateral entry

S. No	Additional mandatory courses for lateral entry students
1	Dip-Mathematics
2	Dip-Programming for Problem Solving
3	Dip-English Communication Skills

8.6 Value Added Courses

The value added courses are audit courses offered through joint ventures with various organizations providing ample scope for the students as well as faculty to keep pace with the latest technologies pertaining to their chosen fields of study. A plenty of value added programs will be proposed by the departments one week before the commencement of class work. The students are given the option to choose the courses according to their desires and inclinations as they choose the desired items in a cafeteria. The expertise gained through the value added programs should enable them to face the formidable challenges of the future and also assist them in exploring new opportunities. Its result shall be declared with "Satisfactory" or "Not Satisfactory" performance.

8.7 Experiential Engineering Education (ExEED)

Engineering entrepreneurship requires strong technical skills in engineering design and computation with key business skills from marketing to business model generation. Students require sufficient skills to innovate in existing companies or create their own.

This course will be evaluated for a total of 100 marks consisting of 30 marks for internal assessment and 70 marks for semester end Examination. Out of 30 marks of internal assessment, The Student has to submit Innovative Idea in a team of four members in the given format. The semester end examination for 70 marks shall be conducted internally, students has to present the Innovative Idea and it will be evaluated by internal ExEEd faculty with at least one faculty member as examiner from the industry, both nominated by the Principal from the panel of experts recommended by the Dean-CLET.

8.8 Project Work / FSI Project Work

This gives students a platform to experience a research driven career in engineering, while developing a device / systems and publishing in reputed SCI / SCOPUS indexed journals and/or filing an **Intellectual Property** (IPR-Patent/Copyright) to aid communities around the world. Students should work individually as per the guidelines issued by head of the department concerned. The benefits to students of this mode of learning include increased engagement, fostering of critical thinking and greater independence.

The topic should be so selected that the students are enabled to complete the work in the stipulated time with the available resources in the respective laboratories. The scope of the work be handling part of the consultancy work, maintenance of the existing equipment, development of new experiment setup or can be a prelude to the main project with a specific outcome.

Project report will be evaluated for 100 marks in total. Assessment will be done for 100 marks out of which, the supervisor / guide will evaluate for 30 marks based on the work and presentation / execution of the work. Subdivision for the remaining 70 marks is based on publication, report, presentation, execution and viva-voce. Evaluation shall be done by a committee comprising the supervisor, Head of the department and an examiner nominated by the Principal from the panel of experts recommended by Chairman, BOS in consultation with Head of the department.

8.8.1 Project work

The student's project activity is spread over in VII semester and in VIII semesters. A student shall carry out the project work under the supervision of a faculty member or in collaboration with an Industry, R&D organization or another academic institution/University where sufficient facilities exist to carry out the project work.

Project work (phase-I) starts in VII semester as it takes a vital role in campus hiring process. Students shall select project titles from their respective logins uploaded by the supervisors at the beginning of VII semester. Three reviews are conducted by department review committee (DRC) for 10 marks each. Student must submit a project report summarizing the work done up to design phase/prototype by the end of VII semester. The semester end examination for project work (phase-I) is evaluated based on the project report submitted and a viva-voce exam for 70 marks by a committee comprising the head of the department, the project supervisor and an external examiner nominated by the Principal.

Project Work (phase-II) starts in VIII semester, shall be evaluated for 100 marks out of which 30 marks towards continuous internal assessment and 70 marks for semester end examination. Three reviews are to be conducted by DRC on the progress of the project for 30 marks. The semester end examination shall be based on the final report submitted and a viva-voce exam for 70 marks by a committee comprising the head of the department, the project supervisor and an external examiner nominated by the Principal.

A minimum of 40% of maximum marks shall be obtained to earn the corresponding credits.

8.8.2 Full Semester Internship (FSI)

FSI is a full semester internship program carry 10 credits. The FSI shall be opted in VII semester or in VIII semester. During the FSI, student has to spend one full semester in an identified industry / firm / R&D organization or another academic institution/University where sufficient facilities exist to carry out the project work.

Following are the evaluation guidelines:

- Quizzes: 2 times
- Quiz #1 - About the industry profile, weightage: 5%
- Quiz #2 - Technical-project related, weightage: 5%
- Seminars - 2 times (once in six weeks), weightage: 7.5% + 7.5%
- Viva-voce: 2 times (once in six weeks), weightage: 7.5% + 7.5%
- Project Report, weightage: 15%

- Internship Diary, weightage: 5 %
- Final Presentation, weightage: 40%

FSI shall be open to all the branches with a ceiling of maximum 10% distributed in both semesters. The selection procedure is:

- Choice of the students.
- CGPA (> 7.5) upto IV semester having no credit arrears.
- Competency Mapping / Allotment.

It is recommended that the FSI Project work leads to a research publication in a reputed Journal/Conference or the filing of patent/design with the patent office, or, the start-up initiative with a sustainable and viable business model accepted by the incubation center of the institute together with the formal registration of the startup.

8.9 Plagiarism index for Project Report:

All project reports shall go through the plagiarism check and the plagiarism index has to be less than 20%. Project reports with plagiarism more than 20% and less than 60% shall be asked for resubmission within a stipulated period of six months. Project reports with plagiarism more than 60% shall be rejected.

9. MAKEUP EXAMINATION

The make-up examination facility shall be available to students who may have missed to attend **CIE/Quiz** of one or more courses in a semester for valid reasons. The CIE make-up examination shall have comprehensive online objective type questions for 20 marks and Quiz for 5 marks. The content for the make-up examination shall be on the whole syllabus. The Makeup examination shall be conducted at the end of the respective semester.

10. SUPPLEMENTARY EXAMINATIONS

In addition to the Regular Semester End Examinations held at the end of each semester, Supplementary Semester End Examinations will be conducted within three weeks of the commencement of the teaching of the next semester. Candidates taking the Regular / Supplementary examinations as Supplementary candidates may have to take more than one Semester End Examination per day. A student can appear for any number of supplementary examinations till he/she clears all courses which he/she could not clear in the first attempt. However the maximum stipulated period for the course shall not be relaxed under any circumstances.

11. ATTENDANCE REQUIREMENTS AND DETENTION POLICY

11.1 It is desirable for a candidate to have 100% attendance in each course. In every course (theory/laboratory), student has to maintain a minimum of 75% attendance including the days of attendance in sports, games, NCC and NSS activities to be eligible for appearing in Semester End Examination of the course.

11.2 In case of medical issues, deficiency of attendance in each course to the extent of 10% may be condoned by the College Academic Committee (CAC) on the recommendation of the Head of the Department if the attendance is between 75% and 65% in every course, subjected to the submission of medical certificates, medical case file, and other needful documents to the concerned departments.

11.3 The basis for the calculation of the attendance shall be the period prescribed by the institute by its calendar of events. For late admission, attendance is reckoned from the date of admission to the program. However, in case of a student having less than 65% attendance in any course, s/he shall be detained in the course and in no case such process will be relaxed.

11.4 A candidate shall put in a minimum required attendance in atleast 60% of (rounded to the next highest integer) theory courses for getting promoted to next higher class / semester. Otherwise, s/he shall be declared detained and has to repeat semester.

11.5 Students whose shortage of attendance is not condoned in any subject are not eligible to write their semester end examination of that courses and their registration shall stand cancelled.

11.6 A prescribed fee shall be payable towards condonation of shortage of attendance.

11.7 A student shall not be promoted to the next semester unless he satisfies the attendance requirement of the present semester, as applicable. They may seek readmission into that semester when offered next. If any candidate fails to fulfill the attendance requirement in the present semester, he shall not be eligible for readmission into the same class.

11.8 Any student against whom any disciplinary action by the institute is pending shall not be permitted to attend any SEE in that semester.

12. CONDUCT OF SEMESTER END EXAMINATIONS AND EVALUATION

12.1 Semester end examination shall be conducted by the Controller of Examinations (COE) by inviting Question Papers from the External Examiners.

12.2 Question papers may be moderated for the coverage of syllabus, pattern of questions by a Semester End Examination Committee chaired by Head of the Department one day before the commencement of semester end examinations. Internal Examiner shall prepare a detailed scheme of valuation.

12.3 The answer papers of semester end examination should be evaluated by the internal examiner immediately after the completion of exam and the award sheet should be submitted to COE in a sealed cover.

12.4 COE shall invite 3 - 9 internal/external examiners to evaluate all the semester end examination answer books on a prescribed date(s). Practical laboratory exams are conducted involving external examiners.

12.5 Examinations Control Committee shall consolidate the marks awarded by examiner/s and award grades.

13. SCHEME FOR THE AWARD OF GRADE

13.1 A student shall be deemed to have satisfied the minimum academic requirements and earn the credits for each theory course, if s/he secures

a) Not less than 35% marks for each theory course in the semester end examination, and

b) A minimum of 40% marks for each theory course considering Continuous Internal Assessment (CIA) and Semester End Examination (SEE).

13.2 A student shall be deemed to have satisfied the minimum academic requirements and earn the credits for each Laboratory / Project work / FSI Project work, if s/he secures

a) Not less than 40% marks for each Laboratory / Project work / FSI Project work course in the semester end examination,

b) A minimum of 40% marks for each Laboratory / Project work / FSI Project work course considering both internal and semester end examination.

13.3 If a candidate fails to secure a pass in a particular course, it is mandatory that s/he shall register and reappear for the examination in that course during the next semester when examination is conducted in that course. It is mandatory that s/he should continue to register and reappear for the examination till s/he secures a pass.

13.4 A student shall be declared successful or ‘passed’ in a semester, if he secures a Grade Point ≥ 5 (‘C’ grade or above) in every course in that semester (i.e. when the student gets an SGPA ≥5.0 at the end of that particular semester); and he shall be declared successful or ‘passed’ in the entire under graduate programme, only when gets a CGPA ≥5.0 for the award of the degree as required.

14. LETTER GRADES AND GRADE POINTS

14.1 Performances of students in each course are expressed in terms of marks as well as in Letter Grades based on absolute grading system. The UGC recommends a 10-point grading system with the following letter grades as given in the Table-9.

Table-9: Grade Points Scale (Absolute Grading)

Range of Marks	Grade Point	Letter Grade
100 – 90	10	S (Superior)
89 – 80	9	A+ (Excellent)
79 – 70	8	A (Very Good)
69 – 60	7	B+ (Good)
59 – 50	6	B (Average)
49 – 40	5	C (Pass)
Below 40	0	F (Fail)
Absent	0	AB (Absent)
Authorized Break of Study	0	ABS

14.2 A student is deemed to have passed and acquired to correspondent credits in particular course if s/he obtains any one of the following grades: “S”, “A+”, “A”, “B+”, “B”, “C”.

14.3 A student obtaining Grade F shall be considered Failed and will be required to reappear in the examination.

14.4 For non credit courses, ‘Satisfactory’ or “Not Satisfactory” is indicated instead of the letter grade and this will not be counted for the computation of SGPA/CGPA.

14.5 “SA” denotes shortage of attendance (as per item 11) and hence prevention from writing Semester End Examination.

14.6 “W” denotes **withdrawal** from the exam for the particular course.

14.7 At the end of each semester, the institute issues grade sheet indicating the SGPA and CGPA of the student. However, grade sheet will not be issued to the student if s/he has any outstanding dues.

14.8 **Award of Class:**

Sometimes, it is necessary to provide equivalence of these averages, viz., SGPA and CGPA with the percentages and/or Class awarded as in the conventional system of declaring the results of University examinations. This shall be done by Autonomous Colleges under the University only at one stage by prescribing certain specific thresholds in these averages for First Class with Distinction, First Class and Second Class, at the time of Degree Award. This provision given in Table-10 follows the approach of the Council for this purpose as reproduced from the AICTE Approval Process Handbook:

Table 10: Percentage Equivalence of Grade Points (for a 10 – Point Scale)

Grade Point	Percentage of Marks / Class
5.5	50
6.0	55
6.5	60
7.0	65
7.5	70
8.0	75

Note:

(1) The following Formula for Conversion of CGPA to percentage of marks to be used only after a student has successfully completed the program:
Percentage of Marks = ($CGPA$ – 0.5) × 10

(2) Class designation:
≥75% (First Class with Distinction),
≥ 60% and <75 % (First Class),
≥ 50 % and <60% (Second Class),
≥45% and <50% (Pass Class).

(3) The SGPA will be computed and printed on the Memorandum of Grades only if the candidate passes in all the courses offered and gets minimum B grade in all the courses.

(4) CGPA is calculated only when the candidate passes in all the courses offered in all the semesters.

15. COMPUTATION OF SGPA AND CGPA

The UGC recommends to compute the Semester Grade Point Average (SGPA) and Cumulative Grade Point Average (CGPA). The credit points earned by a student are used for calculating the Semester Grade Point Average (SGPA) and the Cumulative Grade Point Average (CGPA), both of which are important performance indices of the student. SGPA is equal to the sum of all the total points earned by the student in a given semester divided by the number of credits registered by the student in that semester. CGPA gives the sum of all the total points earned in all the previous semesters and the current semester divided by the number of credits registered in all these semesters. Thus,

$$SGPA = \sum_{i=1}^{n} (C_i G_i) / \sum_{i=1}^{n} C_i$$

Where, C_i is the number of credits of the i^{th} course and G_i is the grade point scored by the student in the i^{th} course and n represent the number of courses in which a student is registered in the concerned semester.

$$CGPA = \sum_{j=1}^{m} (C_j S_j) / \sum_{j=1}^{m} C_j$$

Where, S_j is the SGPA of the j^{th} semester and C_j is the total number of credits upto the semester and m represent the number of semesters completed in which a student registered upto the semester.

The SGPA and CGPA shall be rounded off to 2 decimal points and reported in the transcripts.

16. ILLUSTRATION OF COMPUTATION OF SGPA AND CGPA

16.1 Illustration for SGPA

Course Name	Course Credits	Grade letter	Grade point	Credit Point (Credit x Grade)
Course 1	3	A	8	3 x 8 = 24
Course 2	4	B+	7	4 x 7 = 28
Course 3	3	B	6	3 x 6 = 18
Course 4	3	S	10	3 x 10 = 30
Course 5	3	C	5	3 x 5 = 15
Course 6	4	B	6	4 x 6 = 24
	20			**139**

Thus, SGPA = 139 / 20 = 6.95

16.2 Illustration for CGPA

Semester 1	Semester 2	Semester 3	Semester 4
Credit: 20 SGPA: 6.9	Credit: 22 SGPA: 7.8	Credit: 25 SGPA: 5.6	Credit: 26 SGPA: 6.0
Semester 5	**Semester 6**		
Credit: 26 SGPA: 6.3	Credit: 25 SGPA: 8.0		

$$Thus,\ CGPA = \frac{20x6.9\ +\ 22x7.8\ +\ 25x5.6\ +\ 26x6.0\ +\ 26x6.3\ +\ 25x8.0}{144} = 6.73$$

17. REVIEW OF SEE THEORY ANSWER BOOKS

Semester end examination answer books are made available online in CMS portal on the day of publication of results. A student, who is not satisfied with the assessment, is directed to apply for the review of his/her semester end examination answer book(s) in the theory course(s), within 2 working days from the publication of results in the prescribed format to the Controller of Examinations through the Head of the department with prescribed fee.

The Controller of Examinations shall appoint two examiners (chief examiner of original exam and a new examiner) for the review of the semester end examination (theory) answer book. Both examiners shall jointly review and marks awarded in the previous assessment shall be kept open.

The marks obtained by the candidate after the review shall be considered for grading, only if, the change in mark is more than or equal to 10% of total mark of semester end examination (theory). Marks obtained after re-evaluation shall stand final even if it is less than the original marks. Review is not permitted to the courses other than theory courses.

18. PROMOTION POLICIES

The following academic requirements have to be satisfied in addition to the attendance requirements mentioned in item no. 11.

18.1 For students admitted into B.Tech (Regular) program

18.1.1 A student will not be promoted from II semester to III semester unless s/he fulfills the academic requirement of securing 50% of the total credits (rounded to the next lowest integer) from I and II semester examinations, whether the candidate takes the examination(s) or not.

18.1.2 A student will not be promoted from IV semester to V semester unless s/he fulfills the academic requirement of securing 60% of the total credits (rounded to the next lowest integer) upto III semester **or** 60% of the total credits (rounded to the next lowest integer) up to IV semester, from all the examinations, whether the candidate takes the examination(s) or not.

18.1.3 A student shall be promoted from VI semester to VII semester only if s/he fulfills the academic requirements of securing 60% of the total credits (rounded to the next lowest integer) up to V semester **or** 60% of the total credits (rounded to the next lowest integer) up to VI semester from all the examinations, whether the candidate takes the examination(s) or not.

18.1.4 A student shall register for all the 160 credits and earn all the 160 credits. Marks obtained in all the 160 credits shall be considered for the award of the Grade.

18.2 For students admitted into B.Tech (lateral entry students)

18.2.1 A student will not be promoted from IV semester to V semester unless s/he fulfills the academic requirement of securing 60% of the total credits (rounded to the next lowest integer) up to IV semester, from all the examinations, whether the candidate takes the examination(s) or not.

18.2.2 A student shall be promoted from VI semester to VII semester only if s/he fulfills the academic requirements of securing 60% of the total credits (rounded to the next lowest integer) up to V semester **or** 60% of the total credits (rounded to the next lowest integer) up to VI semester from all the examinations, whether the candidate takes the examination(s) or not.

18.2.3 A student shall register for all the 126 credits and earn all the 126 credits. Marks obtained in all the 126 credits shall be considered for the award of the Grade.

19. GRADUATION REQUIREMENTS

The following academic requirements shall be met for the award of the B.Tech degree.

19.1 Student shall register and acquire minimum attendance in all courses and secure 160 credits (with minimum CGPA of 5.0), for regular program and 126 credits (with minimum CGPA of 5.0), for lateral entry program.

19.2 A student of a regular program, who fails to earn 160 credits within eight consecutive academic years from the year of his/her admission with a minimum CGPA of 5.0, shall forfeit his/her degree and his/her admission stands cancelled.

19.3 A student of a lateral entry program who fails to earn 126 credits within six consecutive academic years from the year of his/her admission with a minimum CGPA of 5.0, shall forfeit his/her degree and his/her admission stands cancelled.

20. BETTERMENT OF MARKS IN THE COURSES ALREADY PASSED

Students who clear all the courses in their first attempt and wish to improve their CGPA shall register and appear for betterment of marks for one course of any theory courses within a period of subsequent two semesters. The improved marks shall be considered for classification / distinction but not for ranking. If there is no improvement, there shall not be any change in the original marks already awarded.

21. AWARD OF DEGREE

21.1 Classification of degree will be as follows:

CGPA ≥ 8.0	CGPA ≥ 6.5 and < 8.0	CGPA ≥ 5.5 and < 6.5	CGPA ≥ 5.0 and < 5.5	CGPA < 5.0
First Class with Distinction	**First Class**	**Second Class**	**Pass Class**	**Fail**

21.2 A student with final CGPA (at the end of the under graduate programme) ≥8.00, and fulfilling the following conditions - shall be placed in '**first class with distinction'**. However,

(a) Should have passed all the courses in '**first appearance'** within the first 4 academic years (or 8 sequential semesters) from the date of commencement of first year first semester.

(b) Should have secured a CGPA ≥8.00, at the end of each of the 8 sequential semesters, starting from I year I semester onwards.

(c) Should not have been detained or prevented from writing the semester end examinations in any semester due to shortage of attendance or any other reason.

A student not fulfilling any of the above conditions with final CGPA >8 shall be placed in **'first class'.**

21.3 Students with final CGPA (at the end of the B.Tech program) ≥6.50 but <8.00 shall be placed in **'first class'.**

21.4 Students with final CGPA (at the end of the B.Tech program) ≥5.50 but <6.50, shall be placed in '**second class**'.

21.5 All other students who qualify for the award of the degree (as per item 19), with final CGPA (at the end of the B.Tech program) ≥5.0 but <5.50, shall be placed in '**pass class**'.

21.6 A student with final CGPA (at the end of the B.Tech program) < 5.00 will not be eligible for the award of the degree.

21.7 Students fulfilling the conditions listed under item 21.2 alone will be eligible for award of '**Gold Medal**'.

21.8. In order to extend the benefit to the students with one/two backlogs after either VI semester or VIII semester, GRAFTING option is provided to the students enabling their placements and fulfilling graduation requirements. Following are the guidelines for the Grafting:

(a) Grafting will be done among the courses within the semester shall draw a maximum of 7 marks from the any one of the cleared courses in the semester and will be grafted to the failed course in the same semester.

(b) Students shall be given a choice of grafting only once in the 4 years program, either after VI semester (Option #1) or after VIII semester (Option #2).

(c) Option#1: Applicable to students who have maximum of TWO theory courses in V and / or VI semesters.

Option#2: Applicable to students who have maximum of TWO theory courses in VII and / or VIII semesters.

(d) Eligibility for grafting:

i. Prior to the conduct of the supplementary examination after the declaration of VI or VIII semester results.

ii. S/he must appear in all regular or supplementary examinations as per the provisions laid down in regulations for the courses s/he appeals for grafting.

iii. The marks obtained by her/him in latest attempt shall be taken into account for grafting of marks in the failed course(s).

21.9 Student, who clears all the courses upto VII semester, shall have a chance to appear for Quick Supplementary Examination to clear the failed courses of VIII semester.

21.10 By the end of VI semester, all the students (regular and lateral entry students) shall complete one of the Value added course and mandatory course with acceptable performance.

21.11 In case, a student takes more than one attempt in clearing a course, the final marks secured shall be indicated by * mark in the grade sheet.

All the candidates who register for the semester end examination will be issued a memorandum of grades sheet by the institute. Apart from the semester wise memorandum of grades sheet, the institute will issue the provisional certificate and consolidated grades memorandum subject to the fulfillment of all the academic requirements.

22. B.TECH WITH HONOURS OR ADDITIONAL MINORS IN ENGINEERING

Students acquiring 160 credits are eligible to get B.Tech degree in Engineering. A student will be eligible to get B.Tech degree with Honours or additional Minors in Engineering, if s/he completes an additional 20 credits (3/4 credits per course). These could be acquired through MOOCs from SWAYAM / NPTEL / edX / Coursera / Udacity / PurdueNext / Khan Academy / QEEE etc. The list for MOOCs will be a dynamic one, as new courses are added from time to time. Few essential skill sets required for employability are also identified year wise. Students interested in doing MOOC courses shall register the course title at their department office at the start of the semester against the courses that are announced by the department. Any expense incurred for the MOOC course / summer program should be met by the students.

Only students having no credit arrears and a CGPA of 7.5 or above at the end of the fourth semester are eligible to register for B.Tech (Honours / Minor). After registering for the B.Tech (Honours / Minor) program, if a student fails in any course, s/he will not be eligible for B.Tech (Honours / Minor).

Every Department to develop and submit a Honours / Minors – courses list of 5 - 6 theory courses.

Honours Certificate for Vertical in his/her OWN Branch for Research orientation; Minor in any other branch for Improving Employability.

For the MOOCs platforms, where examination or assessment is absent (like SWAYAM) or where certification is costly (like Coursera or edX), faculty members of the institute prepare the examination question papers, for the courses undertaken by the students of respective Institutes, so that examinations Control Office (ECO) can conduct examination for the course. There shall be one Continuous Internal Examination (Quiz exam for 30 marks) after 8 weeks of the commencement of the course and semester end examination (Descriptive exam for 70 marks) shall be done along with the other regular courses.

A student can enroll for both Minor & Honours or for two Minors. The final grade sheet will only show the basic CGPA corresponding to the minimum requirement for the degree. The Minors/Honours will be indicated by a separate CGPA. The additional courses taken will also find separate mention in the grade sheet.

If a student drops (or terminated) from the Minor/Honours program, they cannot convert the earned credits into free or core electives; they will remain extra. These additional courses will find mention in the grade sheet (but not in the degree certificate). In such cases, the student may choose between the actual grade or a "Pass (P)" grade and also choose to omit the mention of the course as for the following:

- All the courses done under the dropped Minor/Honours will be shown in the grade sheet
- None of the courses done under the dropped Minor/Honours will be shown in the grade sheet.

Honours will be reflected in the degree certificate as "B.Tech (Honours) in XYZ Engineering". Similarly, Minor as "B.Tech in XYZ Engineering with Minor in ABC". If a student has done both Honours & Minor, it will be acknowledged as "B.Tech (Honours) in XYZ Engineering with Minor in ABC". And two minors will be reflected as "B.Tech in XYZ Engineering with Minor in ABC and Minor in DEF".

22.1. B.Tech with Honours

The total of 20 credits required to be attained for B.Tech Honours degree are distributed from V semester to VII semester in the following way:

For V semester	:	4 – 8 credits
For VI semester	:	4 – 8 credits
For VII semester	:	4 – 8 credits

Following are the details of such Honours which include some of the most interesting areas in the profession today:

S. No	Department	Honours scheme
1	Aeronautical Engineering	Aerospace Engineering / Space Science etc.
2	Computer Science and Engineering / Information Technology	Big data and Analytics / Cyber Physical Systems, Information Security / Cognitive Science / Artificial Intelligence/ Machine Learning / Data Science / Internet of Things (IoT) etc.
3	Electronics and Communication Engineering	Digital Communication / Signal Processing / Communication Networks / VLSI Design / Embedded Systems etc.
4	Electrical and Electronics Engineering	Renewable Energy systems / Energy and Sustainability / IoT Applications in Green Energy Systems etc.
5	Mechanical Engineering	Industrial Automation and Robotics / Manufacturing Sciences and Computation Techniques etc.
6	Civil Engineering	Structural Engineering / Environmental Engineering etc.

22.2 B.Tech with additional Minor in Engineering

Every department to develop and submit Minor courses list of 5 - 6 Theory courses. Student from any department is eligible to apply for Minor from any other department. The total of 20 credits to complete the B.Tech (Minor) program by registering for MOOC courses each having a minimum of 3/4 credits offered by reputed institutions / organization with the approval of the department. Registration of the student for B.Tech (Minor), is from V Semester to VII Semester of the program in the following way:

For V semester	:	4 – 8 credits
For VI semester	:	4 – 8 credits
For VII semester	:	4 – 8 credits

Only students having no credit arrears and a CGPA of 7.5 or above at the end of the fourth semester are eligible to register for B.Tech (Minor). After registering for the B.Tech (Minor) program, if a student fails in any course, s/he will not be eligible for B.Tech (Minor).

Every student shall also have the option to do a minor in engineering. A major is a primary focus of study and a minor is a secondary focus of study. The minor has to be a subject offered by a department other than the department that offers the major of the student or it can be a different major offered by the same department. For example, a student with the declared major in Computer Science and Engineering (CSE) may opt to do a minor in Physics; in which case, the student shall receive the degree B.Tech, Computer Science and Engineering with a minor in Physics. A student can do Majors in chosen filed as per the career goal, and a minor may be chosen to enhance the major thus adding the diversity, breadth and enhanced skills in the field.

22.3 Advantages of Minor in Engineering:

The minors mentioned above are having lots of advantages and a few are listed below:

1. To apply the inter-disciplinary knowledge gained through a Major (Stream) + Minor.
2. To enable students to pursue allied academic interest in contemporary areas.
3. To provide an academic mechanism for fulfilling multidisciplinary demands of industries.
4. To provide effective yet flexible options for students to achieve basic to intermediate level competence in the Minor area.
5. Provides an opportunity to students to become entrepreneurs and leaders by taking business/ management minor.
6. Combination in the diverse fields of engineering e.g., CSE (Major) + Electronics (Minor) combination increases placement prospects in chip designing companies.
7. Provides an opportunity to Applicants to pursue higher studies in an inter-disciplinary field of study.
8. Provides opportunity to the Applicants to pursue interdisciplinary research.
9. To increase the overall scope of the undergraduate degrees.

22.4 Following are the details of such Minor / Honours which include some of the most interesting areas in the profession today:

1. Aerospace Engineering
2. Space Science
3. Industrial Automation and Robotics
4. Computer Science and Engineering
5. Data Analytics
6. Machine Learning
7. Data Science
8. Artificial Intelligence
9. Information Security
10. Internet of Things
11. Cyber Physical Systems
12. Electronic System Design
13. Renewable Energy Sources
14. Energy and Sustainability
15. Manufacturing Sciences and Computation Techniques
16. Structural Engineering
17. Environmental Engineering
18. Technological Entrepreneurship
19. Materials Engineering
20. Physics (Materials / Nuclear / Optical / Medical)
21. Mathematics (Combinatorics / Logic / Number theory / Dynamical systems and differential equations/ Mathematical physics / Statistics and Probability).

23.0 TEMPORARY BREAK OF STUDY FROM THE PROGRAM

23.1 A candidate is normally not permitted to take a break from the study. However, if a candidate intends to temporarily discontinue the program in the middle for valid reasons (such as accident or hospitalization due to prolonged ill health) and to rejoin the program in a later respective semester, s/he shall seek the approval from the Principal in advance. Such application shall be submitted before the last date for payment of examination fee of the semester in question and forwarded through the Head of the Department stating the reasons for such withdrawal together with supporting documents and endorsement of his / her parent / guardian.

23.2 The institute shall examine such an application and if it finds the case to be genuine, it may permit the student to temporarily withdraw from the program. Such permission is accorded only to those who do not have any outstanding dues / demand at the College / University level including tuition fees, any other fees, library materials etc.

23.3 The candidate has to rejoin the program after the break from the commencement of the respective semester as and when it is offered.

23.4 The total period for completion of the program reckoned from the commencement of the semester to which the candidate was first admitted shall not exceed the maximum period specified in clause 19. The maximum period includes the break period.

23.5 If any candidate is detained for any reason, the period of detention shall not be considered as ‘Break of Study’.

24. TERMINATION FROM THE PROGRAM

The admission of a student to the program may be terminated and the student is asked to leave the institute in the following circumstances:

a. The student fails to satisfy the requirements of the program within the maximum period stipulated for that program.

b. A student shall not be permitted to study any semester more than three times during the entire program of study.

c. The student fails to satisfy the norms of discipline specified by the institute from time to time.

25. TRANSCRIPT

The Transcript will be issued to the student as and when required and will contain a consolidated record of all the courses undergone by him/her, grades obtained and CGPA upto the date of issue of transcript. Only last letter grade obtained in a course by the student upto the date of issue of transcript will be shown in the Transcript.

26. WITH-HOLDING OF RESULTS

If the candidate has not paid any dues to the institute / if any case of indiscipline / malpractice is pending against him, the results and the degree of the candidate will be withheld.

27. GRADUATION DAY

The institute shall have its own annual Graduation Day for the award of degrees to the students completing the prescribed academic requirements in each case, in consultation with the University and by following the provisions in the Statute. The college shall institute prizes and medals to meritorious students and award them annually at the Graduation Day. This will greatly encourage the students to strive for excellence in their academic work.

28. DISCIPLINE

Every student is required to observe discipline and decorum both inside and outside the institute and are expected not to indulge in any activity which will tend to bring down the honour of the institute. If a student indulges in malpractice in any of the theory / practical examination, continuous assessment examinations, he/she shall be liable for punitive action as prescribed by the institute from time to time.

29. GRIEVANCE REDRESSAL COMMITTEE

The institute shall form a Grievance Redressal Committee for each course in each department with the Course

Teacher and the HOD as the members. This Committee shall solve all grievances related to the course under consideration.

30. TRANSITORY REGULATIONS

A candidate, who is detained or has discontinued a semester, on readmission shall be required to do all the courses in the curriculum prescribed for the batch of students in which the student joins subsequently. However, exemption will be given to those candidates who have already passed such courses in the earlier semester(s) he was originally admitted into and substitute subjects are offered in place of them as decided by the Board of Studies. However, the decision of the Board of Studies will be final.

a) Four Year B.Tech Regular course:

A student who is following Jawaharlal Nehru Technological University (JNTUH) curriculum and detained due to the shortage of attendance at the end of the first semester shall join the autonomous batch of first semester. Such students shall study all the courses prescribed for the batch in which the student joins and considered on par with regular candidates of Autonomous stream and will be governed by the autonomous regulations.

A student who is following JNTUH curriculum, detained due to lack of credits or shortage of attendance at the end of the second semester or at the subsequent semesters shall join with the autonomous batch in the appropriate semester. Such candidates shall be required to pass in all the courses in the program prescribed by the Board of Studies concerned for that batch of students from that semester onwards to be eligible for the award of degree. However, exemption will be given in the courses of the semester(s) of the batch which he had passed earlier and substitute courses will be offered in place of them as decided by the Board of Studies. The student has to clear all his backlog courses up to previous semester by appearing for the supplementary examinations conducted by JNTUH for the award of degree. The total number of credits to be secured for the award of the degree will be sum of the credits up to previous semester under JNTUH regulations and the credits prescribed for the semester in which a candidate seeks readmission and subsequent semesters under the autonomous stream. The class will be awarded based on the academic performance of a student in the autonomous pattern.

b) Three Year B.Tech program under Lateral Entry Scheme:

A student who is following JNTUH curriculum and detained due to the shortage of attendance at the end of the first semester of second year shall join the autonomous batch of third semester. Such students shall study all the courses prescribed for the batch in which the student joins and considered on par with Lateral Entry regular candidates of Autonomous stream and will be governed by the autonomous regulations.

A student who is following JNTUH curriculum, if detained due to lack of credits or shortage of attendance at the end of the second semester of second year or at the subsequent semesters shall join with the autonomous batch in the appropriate semester. Such candidates shall be required to pass in all the courses in the program prescribed by the Board of Studies concerned for that batch of students from that semester onwards to be eligible for the award of degree. However, exemption will be given in the courses of the semester(s) of the batch which he had passed earlier and substitute courses are offered in place of them as decided by the Board of Studies. The student has to clear all his backlog courses up to previous semester by appearing for the supplementary examinations conducted by JNTUH for the award of degree. The total number of credits to be secured for the award of the degree will be sum of the credits up to previous semester under JNTUH regulations and the credits prescribed for the semester in which a candidate seeks readmission and subsequent semesters under the autonomous status. The class will be awarded based on the academic performance of a student in the autonomous pattern.

c) Transfer candidates (from non-autonomous college affiliated to JNTUH):

A student who is following JNTUH curriculum, transferred from other college to this institute in third semester or subsequent semesters shall join with the autonomous batch in the appropriate semester. Such candidates shall be required to pass in all the courses in the program prescribed by the Board of Studies concerned for that batch of students from that semester onwards to be eligible for the award of degree. However, exemption will be given in the courses of the semester(s) of the batch which he had passed earlier and substitute courses are offered in their place as decided by the Board of Studies. The student has to clear all his backlog courses up to previous semester by appearing for the supplementary examinations conducted by JNTUH for the award of degree. The total number of credits to be secured for the award of the degree will be the sum of the credits up to the previous semester under JNTUH regulations and the credits prescribed for the semester in which a candidate joined after transfer and subsequent semesters under the autonomous status. The class will be awarded based on the academic performance of a student in the autonomous pattern.

d) Transfer candidates (from an autonomous college affiliated to JNTUH):

A student who has secured the required credits up to previous semesters as per the regulations of other autonomous institutions shall also be permitted to be transferred to this institute. A student who is transferred from the other autonomous colleges to this institute in third semester or subsequent semesters shall join with the autonomous batch in the appropriate semester. Such candidates shall be required to pass in all the courses in the program prescribed by the Board of Studies concerned for that batch of students from that semester onwards to be eligible for the award of degree. However, exemption will be given in the courses of the semester(s) of the batch which he had passed earlier and substitute subjects are offered in their place as decided by the Board of Studies. The total number of credits to be secured for the award of the degree will be the sum of the credits up to previous semester as per the regulations of the college from which he is transferred and the credits prescribed for the semester in which a candidate joined after transfer and subsequent semesters under the autonomous status. The class will be awarded based on the academic performance of a student in the autonomous pattern.

e) Readmission from IARE-R16/R18 to IARE-UG.20 regulations

A student took admission in IARE-R18 Regulations, detained due to lack of required number of credits or percentage of attendance at the end of any semester is permitted to take re-admission at appropriate level under any regulations prevailing in the institute subject to the following rules and regulations.

1. Student shall pass all the courses in the earlier scheme of regulations (IARE - R18). However, in case of having backlog courses, they shall be cleared by appearing for supplementary examinations conducted under IARE - R18 regulations from time to time.
2. After rejoining, the student is required to study the courses as prescribed in the new regulations for the re-admitted program at that level and thereafter.
3. If the student has already passed any course(s) of readmitted program in the earlier regulation / semester of study, such courses are exempted in the new scheme to appear for the course(s).
4. The courses that are not done in the earlier regulations / semester as compared with readmitted program need to be cleared after readmission by appearing for the examinations conducted time to time under the new regulations.
5. In general, after transition, course composition and number of credits / semester shall be balanced between earlier and new regulations on case to case basis.
6. In case, the students who do not have option of acquiring required credits with the existing courses offered as per the new curriculum, credit balance can be achieved by clearing the additional courses offered by the respective departments (approved in Academic Council meeting). The additional courses that are offered can be of theory or laboratory courses and shall be offered during semester.
7. Students re-joined in III semester shall be treated on par with “Lateral Entry” students for credits and graduation requirements. However, the student shall clear all the courses in B.Tech I Semester and B.Tech II Semester as per IARE-R18 regulations.

31. REVISION OF REGULATIONS AND CURRICULUM

The Institute from time to time may revise, amend or change the regulations, scheme of examinations and syllabi if found necessary and on approval by the Academic Council and the Governing Body and shall be binding on the students, faculty, staff, all authorities of the Institute and others concerned.

1. **Who grants Autonomy? UGC, Govt., AICTE or University**

FREQUENTLY ASKED QUESTIONS AND ANSWERS ABOUT AUTONOMY

1. **Who grants Autonomy? UGC, Govt., AICTE or University**

In case of Colleges affiliated to a university and where statutes for grant of autonomy are ready, it is the respective University that finally grants autonomy but only after concurrence from the respective state Government as well as UGC. The State Government has its own powers to grant autonomy directly to Govt. and Govt. aided Colleges.

2 Shall IARE award its own Degrees?

No. Degree will be awarded by Jawaharlal Nehru Technological University, Hyderabad with a mention of the name IARE on the Degree Certificate.

3 What is the difference between a Deemed University and an Autonomy College?

A Deemed University is fully autonomous to the extent of awarding its own Degree. A Deemed University is usually a Non-Affiliating version of a University and has similar responsibilities like any University. An Autonomous College enjoys Academic Autonomy alone. The University to which an autonomous college is affiliated will have checks on the performance of the autonomous college.

4 How will the Foreign Universities or other stake – holders know that we are an Autonomous College?

Autonomous status, once declared, shall be accepted by all the stake holders. The Govt. of Telangana mentions autonomous status during the First Year admission procedure. Foreign Universities and Indian Industries will know our status through our website.

5 What is the change of Status for Students and Teachers if we become Autonomous?

An autonomous college carries a prestigious image. Autonomy is actually earned out of our continued past efforts on academic performances, our capability of self- governance and the kind of quality education we offer.

6 Who will check whether the academic standard is maintained / improved after Autonomy? How will it be checked?

There is a built in mechanism in the autonomous working for this purpose. An Internal Committee called Academic Program Evaluation Committee, which will keep a watch on the academics and keep its reports and recommendations every year. In addition the highest academic council also supervises the academic matters. The standards of our question papers, the regularity of academic calendar, attendance of students, speed and transparency of result declaration and such other parameters are involved in this process.

7 Will the students of IARE as an Autonomous College qualify for University Medals and Prizes for academic excellence?

No. IARE has instituted its own awards, medals, etc. for the academic performance of the students. However for all other events like sports, cultural on co-curricular organized by the University the students shall qualify.

8 Can IARE have its own Convocation?

No. Since the University awards the Degree the Convocation will be that of the University, but there will be Graduation Day at IARE.

9 Can IARE give a provisional degree certificate?

Since the examinations are conducted by IARE and the results are also declared by IARE, the college sends a list of successful candidates with their final Grades and Grade Point Averages including CGPA to the University. Therefore with the prior permission of the University the college will be entitled to give the provisional certificate.

10 Will Academic Autonomy make a positive impact on the Placements or Employability?

Certainly. The number of students qualifying for placement interviews is expected to improve, due to rigorous and repetitive classroom teaching and continuous assessment. Also the autonomous status is more responsive to the needs of the industry. As a result therefore, there will be a lot of scope for industry oriented skill development built-in into the system. The graduates from an autonomous college will therefore represent better employability.

11 What is the proportion of Internal and External Assessment as an Autonomous College?

Presently, it is 60% external and 40% internal. As the autonomy matures the internal assessment component shall be increased at the cost of external assessment.

12 Is it possible to have complete Internal Assessment for Theory or Practicals?

Yes indeed. We define our own system. We have the freedom to keep the proportion of external and internal assessment component to choose.

13 Why Credit based Grade System?

The credit based grade system is an accepted standard of academic performance the world over in all Universities. The acceptability of our graduates in the world market shall improve.

14 What exactly is a Credit based Grade System?

The credit based grade system defines a much better statistical way of judging the academic performance. One Lecture Hour per week of Teaching Learning process is assigned One Credit. One hour of laboratory work is assigned half credit. Letter Grades like A, B,C,D, etc. are assigned for a Range of Marks. (e.g. 91% and above is A+, 80 to 90 % could be A etc.) in Absolute Grading System while grades are awarded by statistical analysis in relative grading system. We thus dispense with sharp numerical boundaries. Secondly, the grades are associated with defined Grade Points in the scale of 1 to 10. Weighted Average of Grade Points is also defined Grade Points are weighted by Credits and averaged over total credits in a Semester. This process is repeated for all Semesters and a CGPA defines the Final Academic Performance

15 What are the norms for the number of Credits per Semester and total number of Credits for UG/PG program?

These norms are usually defined by UGC or AICTE. Usually around 25 Credits per semester is the accepted norm.

16 What is a Semester Grade Point Average (SGPA)?

The performance of a student in a semester is indicated by a number called SGPA. The SGPA is the weighted average of the grade points obtained in all the courses registered by the student during the semester.

$$SGPA = \sum_{i=1}^{n} \left(C_i \, G_i \right) / \sum_{i=1}^{n} C_i$$

Where, C_i is the number of credits of the i^{th} course and G_i is the grade point scored by the student in the i^{th} course and i represent the number of courses in which a student registered in the concerned semester. SGPA is rounded to two decimal places.

17 What is a Cumulative Grade Point Average (CGPA)?

An up-to-date assessment of overall performance of a student from the time of his first registration is obtained by calculating a number called CGPA, which is weighted average of the grade points obtained in all the courses registered by the students since he entered the Institute.

$$CGPA = \sum_{j=1}^{m} \left(C_j \, S_j \right) / \sum_{j=1}^{m} C_j$$

Where, S_j is the SGPA of the j^{th} semester and C_j is the total number of credits upto the semester and m represent the number of semesters completed in which a student registered upto the semester. CGPA is rounded to two decimal places.

18 Is there any Software available for calculating Grade point averages and converting the same into Grades?

Yes, The institute has its own MIS software for calculation of SGPA, CGPA, etc.

19 Will the teacher be required to do the job of calculating SGPAs etc. and convert the same into Grades?

No. The teacher has to give marks obtained out of whatever maximum marks as it is. Rest is all done by the computer.

20 Will there be any Revaluation or Re-Examination System?

No. There will double valuation of answer scripts. There will be a makeup Examination after a reasonable preparation time after the End Semester Examination for specific cases mentioned in the Rules and Regulations. In addition to this, there shall be a ‘summer term’ (compressed term) followed by the End Semester Exam, to save the precious time of students.

21 How fast Syllabi can be and should be changed?

Autonomy allows us the freedom to change the syllabi as often as we need.

22 Will the Degree be awarded on the basis of only final year performance?

No. The CGPA will reflect the average performance of all the semester taken together.

23 What are Statutory Academic Bodies?

Governing Body, Academic Council, Examination Committee and Board of Studies are the different statutory bodies. The participation of external members in everybody is compulsory. The institute has nominated professors from IIT, NIT, University (the officers of the rank of Pro-vice Chancellor, Deans and Controller of Examinations) and also the reputed industrialist and industry experts on these bodies.

24 Who takes Decisions on Academic matters?

The Governing Body of institute is the top academic body and is responsible for all the academic decisions. Many decisions are also taken at the lower level like Boards of Studies. Decisions taken at the Boared of Studies level are to be ratified at the Academic Council and Governing Body.

25 What is the role of Examination committee?

The Examinations Committee is responsible for the smooth conduct of internal, End Semester and make up Examinations. All matters involving the conduct of examinations spot valuations, tabulations preparation of Grade Sheet etc fall within the duties of the Examination Committee.

26 Is there any mechanism for Grievance Redressal?

The institute has grievance redressal committee, headed by Dean - Student affairs and Dean - IQAC.

27 How many attempts are permitted for obtaining a Degree?

All such matters are defined in Rules & Regulation

28 Who declares the result?

The result declaration process is also defined. After tabulation work wherein the SGPA, CGPA and final Grades are ready, the entire result is reviewed by the Moderation Committee. Any unusual deviations or gross level discrepancies are deliberated and removed. The entire result is discussed in the Examinations and Result Committee for its approval. The result is then declared on the institute notice boards as well put on the web site and Students Corner. It is eventually sent to the University.

29 Who will keep the Student Academic Records, University or IARE?

It is the responsibility of the Dean, Academics of the Autonomous College to keep and preserve all the records.

30 What is our relationship with the JNT University?

We remain an affiliated college of the JNT University. The University has the right to nominate its members on the academic bodies of the college.

31 Shall we require University approval if we want to start any New Courses?

Yes, It is expected that approvals or such other matters from an autonomous college will receive priority.

32 Shall we get autonomy for PG and Doctoral Programs also?

Yes, presently our PG programs also enjoying autonomous status.

MALPRACTICE RULES

DISCIPLINARY ACTION FOR / IMPROPER CONDUCT IN EXAMINATIONS

S. No	Nature of Malpractices/Improper conduct	Punishment
	If the candidate:	
1. (a)	Possesses or keeps accessible in examination hall, any paper, note book, programmable calculator, cell phone, pager, palm computer or any other form of material concerned with or related to the subject of the examination (theory or practical) in which he is appearing but has not made use of (material shall include any marks on the body of the candidate which can be used as an aid in the subject of the examination)	Expulsion from the examination hall and cancellation of the performance in that subject only.
(b)	Gives assistance or guidance or receives it from any other candidate orally or by any other body language methods or communicates through cell phones with any candidate or persons in or outside the exam hall in respect of any matter.	Expulsion from the examination hall and cancellation of the performance in that subject only of all the candidates involved. In case of an outsider, he will be handed over to the police and a case is registered against him.
2.	Has copied in the examination hall from any paper, book, programmable calculators, palm computers or any other form of material relevant to the subject of the examination (theory or practical) in which the candidate is appearing.	Expulsion from the examination hall and cancellation of the performance in that subject and all other subjects the candidate has already appeared including practical examinations and project work and shall not be permitted to appear for the remaining examinations of the subjects of that Semester/year. The Hall Ticket of the candidate is to be cancelled and sent to the Controller of Examinations.
3.	Impersonates any other candidate in connection with the examination.	The candidate who has impersonated shall be expelled from examination hall. The candidate is also debarred and forfeits the seat. The performance of the original candidate, who has been impersonated, shall be cancelled in all the subjects of the examination (including practicals and project work) already appeared and shall not be allowed to appear for examinations of the remaining subjects of that semester/year. The candidate is also debarred for two consecutive semesters from class work and all semester end examinations. The continuation of the course by the candidate is subject to the academic regulations in connection with forfeiture of seat. If the imposter is an outsider, he will be handed over to the police and a case is registered against him.
4.	Smuggles in the Answer book or additional sheet or takes out or arranges to send out the question paper during the examination or answer book or	Expulsion from the examination hall and cancellation of performance in that subject and all the other subjects the candidate has already

	additional sheet, during or after the examination.	appeared including practical examinations and project work and shall not be permitted for the remaining examinations of the subjects of that semester/year. The candidate is also debarred for two consecutive semesters from class work and all semester end examinations. The continuation of the course by the candidate is subject to the academic regulations in connection with forfeiture of seat.
5.	Uses objectionable, abusive or offensive language in the answer paper or in letters to the examiners or writes to the examiner requesting him to award pass marks.	Cancellation of the performance in that subject.
6.	Refuses to obey the orders of the Controller of Examinations /Additional Controller of Examinations/any officer on duty or misbehaves or creates disturbance of any kind in and around the examination hall or organizes a walk out or instigates others to walk out, or threatens the COE or any person on duty in or outside the examination hall of any injury to his person or to any of his relations whether by words, either spoken or written or by signs or by visible representation, assaults the COE or any person on duty in or outside the examination hall or any of his relations, or indulges in any other act of misconduct or mischief which result in damage to or destruction of property in the examination hall or any part of the Institute premises or engages in any other act which in the opinion of the officer on duty amounts to use of unfair means or misconduct or has the tendency to disrupt the orderly conduct of the examination.	In case of students of the college, they shall be expelled from examination halls and cancellation of their performance in that subject and all other subjects the candidate(s) has (have) already appeared and shall not be permitted to appear for the remaining examinations of the subjects of that semester/year. The candidates also are debarred and forfeit their seats. In case of outsiders, they will be handed over to the police and a police case is registered against them.
7.	Leaves the exam hall taking away answer script or intentionally tears off the script or any part thereof inside or outside the examination hall.	Expulsion from the examination hall and cancellation of performance in that subject and all the other subjects the candidate has already appeared including practical examinations and project work and shall not be permitted for the remaining examinations of the subjects of that semester/year. The candidate is also debarred for two consecutive semesters from class work and all semester end examinations. The continuation of the course by the candidate is subject to the academic regulations in connection with forfeiture of seat.
8.	Possess any lethal weapon or firearm in the examination hall.	Expulsion from the examination hall and cancellation of the performance in that subject and all other subjects the candidate has already appeared including practical examinations and project work and shall not be permitted for the remaining examinations of the subjects of that semester/year. The candidate is also debarred and forfeits the seat.

9.	If student of the college, who is not a candidate for the particular examination or any person not connected with the college indulges in any malpractice or improper conduct mentioned in clause 6 to 8.	Student of the colleges expulsion from the examination hall and cancellation of the performance in that subject and all other subjects the candidate has already appeared including practical examinations and project work and shall not be permitted for the remaining examinations of the subjects of that semester/year. The candidate is also debarred and forfeits the seat. Person(s) who do not belong to the College will be handed over to police and, a police case will be registered against them.
10.	Comes in a drunken condition to the examination hall.	Expulsion from the examination hall and cancellation of the performance in that subject and all other subjects the candidate has already appeared including practical examinations and project work and shall not be permitted for the remaining examinations of the subjects of that semester/year.
11.	Copying detected on the basis of internal evidence, such as, during valuation or during special scrutiny.	Cancellation of the performance in that subject and all other subjects the candidate has appeared including practical examinations and project work of that semester/year examinations.
12.	If any malpractice is detected which is not covered in the above clauses 1 to 11 shall be reported to the University for further action to award suitable punishment.	

FAILURE TO READ AND UNDERSTAND THE REGULATIONS IS NOT AN EXCUSE

INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad – 500043

COURSE CATALOG

(ELECTRONICS AND COMMUNICATION ENGINEERING)

I SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AHSC01	English	HSMC	Foundation	3	0	0	3	30	70	100
AHSC02	Linear Algebra and Calculus	BSC	Foundation	3	1	0	4	30	70	100
AHSC03	Engineering Physics	BSC	Foundation	3	0	0	3	30	70	100
ACSC01	Python Programming	ESC	Foundation	3	0	0	3	30	70	100
PRACTICAL										
AHSC04	English Language and Communication Skills Laboratory	HSMC	Foundation	0	0	2	1	30	70	100
AHSC05	Physics Laboratory	BSC	Foundation	0	0	3	1.5	30	70	100
ACSC02	Python Programming Laboratory	ESC	Foundation	0	0	3	1.5	30	70	100
TOTAL				**12**	**01**	**08**	**17**	**210**	**490**	**700**

II SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AHSC06	Chemistry	BSC	Foundation	2	0	0	2	30	70	100
AHSC07	Mathematical Transform Techniques	BSC	Foundation	3	1	0	4	30	70	100
AEEC02	Electrical Circuits	ESC	Foundation	3	0	0	3	30	70	100
ACSC04	Programming for Problem Solving using C	ESC	Foundation	3	0	0	3	30	70	100
ACSC06	Experiential Engineering Education (ExEEd) – Academic Sucess	ESC	Foundation	2	0	0	1	30	70	100
PRACTICAL										
AEEC03	Electrical Circuits Laboratory	ESC	Foundation	0	0	3	1.5	30	70	100
ACSC05	Programming for Problem Solving using C Laboratory	ESC	Foundation	0	0	3	1.5	30	70	100
AMEC04	Engineering Workshop Practice	ESC	Foundation	0	0	2	1	30	70	100
TOTAL				**13**	**01**	**08**	**17**	**240**	**560**	**800**

III SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AECC01	Electronic Devices and Circuits	PCC	Core	3	1	0	4	30	70	100
AECC02	Signals and Systems	PCC	Core	3	1	0	4	30	70	100
AECC03	Digital System Design	PCC	Core	3	0	0	3	30	70	100
AECC04	Probability Theory and Stochastic Processes	BSC	Foundation	3	0	0	3	30	70	100
ACSC08	Data Structures	PCC	Core	3	0	0	3	30	70	100
ACSC09	ExEEd - Prototype / Design Building	ESC	Foundation	2	0	0	1	30	70	100
PRACTICALS										
AECC05	Electronic Devices and Circuits Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC06	Digital System Design Laboratory	PCC	Core	0	0	2	1	30	70	100
ACSC10	Data Structures Laboratory	PCC	Core	0	0	3	1.5	30	70	100
MANDATORY / VALUE ADDED COURSES										
AHSC10	Essence of Indian Traditional Knowledge	MC	MC-I	Ref: 8.4 Academic Regulations, UG.20						
TOTAL				**17**	**02**	**08**	**22**	**270**	**630**	**900**

IV SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AHSC12	Complex Analysis and Special Functions	BSC	Foundation	3	1	0	4	30	70	100
AECC09	Analog and Pulse Circuits	PCC	Core	3	0	0	3	30	70	100
AECC10	Analog and Digital Communications	PCC	Core	3	0	0	3	30	70	100
AECC11	Electromagnetic Waves and Transmission Lines	PCC	Core	3	1	0	4	30	70	100
AECC12	IC Applications	PCC	Core	3	0	0	3	30	70	100
ACSC14	ExEEd - Fabrication / Model Development	ESC	Foundation	2	0	0	1	30	70	100
PRACTICALS										
AECC13	Analog and Pulse Circuits Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC14	Analog and Digital Communications Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC15	IC Applications Laboratory	PCC	Core	0	0	2	1	30	70	100
MANDATORY / VALUE ADDED COURSES										
ACSC18	Fundamentals of Database Systems	VAC-I	Skill	Ref: 8.6, Academic Regulations-UG.20						
	TOTAL			**17**	**02**	**08**	**22**	**270**	**630**	**900**

V SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AECC18	Antennas and Wave Propagation	PCC	Core	3	1	0	4	30	70	100
AECC19	Microprocessors and Microcontrollers	PCC	Core	3	1	0	4	30	70	100
AECC20	Electronic Measurements and Instrumentation	PCC	Core	3	1	0	4	30	70	100
AECC21	Control Systems	PCC	Core	3	0	0	3	30	70	100
	Professional Elective - I	PEC	Elective	3	0	0	3	30	70	100
ACSC20	ExEEd - Project Based Learning	ESC	Foundation	2	0	0	1	30	70	100
PRACTICALS										
AECC30	Virtual Instrumentation Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC31	Microprocessors and Microcontrollers Laboratory	PCC	Core	0	0	3	1.5	30	70	100
MANDATORY / VALUE ADDED COURSES										
ACSC23	Object Oriented Programming Development and Languages	VAC-II	Skill	Ref: 8.6, Academic Regulations-UG.20						
TOTAL				**17**	**03**	**06**	**22**	**240**	**560**	**800**

VI SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AECC32	Microwave and Radar Engineering	PCC	Core	3	1	0	4	30	70	100
AECC33	Digital Signal Processing	PCC	Core	3	1	0	4	30	70	100
AHSC13	Business Economics and Financial Analysis	HSMC	Foundation	3	0	0	3	30	70	100
	Professional Elective - II	PEC	Elective	3	0	0	3	30	70	100
	Open Elective – I	OEC	Elective	3	0	0	3	30	70	100
ACSC27	ExEEd - Research Based Learning	ESC	Foundation	2	0	0	1	30	70	100
PRACTICALS										
AECC41	Antennas and Microwave Engineering Laboratory	PCC	Core	0	0	4	2	30	70	100
AECC42	Digital Signal Processing Laboratory	PCC	Core	0	0	4	2	30	70	100
MANDATORY / VALUE ADDED COURSES										
ACSC29	Design of Algorithms	VAC-III	Skill	Ref: 8.6, Academic Regulations-UG.20						
TOTAL				**17**	**02**	**06**	**22**	**240**	**560**	**800**

VII SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
AECC43	Embedded System Design	PCC	Core	3	0	0	3	30	70	100
AECC44	VLSI Design	PCC	Core	3	1	0	4	30	70	100
	Professional Elective – III	PEC	Elective	3	0	0	3	30	70	100
	Professional Elective - IV	PEC	Elective	3	0	0	3	30	70	100
	Open Elective - II	OEC	Elective	3	0	0	3	30	70	100
PRACTICALS										
AECC53	Embedded System Design Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC54	VLSI Design Laboratory	PCC	Core	0	0	3	1.5	30	70	100
AECC55	Project Work (Phase - I)	PROJ	Project	0	0	4	2	30	70	100
TOTAL				**15**	**01**	**10**	**21**	**240**	**560**	**800**

VIII SEMESTER

Course Code	Course Name	Subject Area	Category	Periods per week			Credits	Scheme of Examination Max. Marks		
				L	T	P		CIA	SEE	Total
THEORY										
	Professional Elective - V	PEC	Elective	3	0	0	3	30	70	100
	Professional Elective - VI	PEC	Elective	3	0	0	3	30	70	100
	Open Elective - III	OEC	Elective	3	0	0	3	30	70	100
PRACTICALS										
AECC64	Project Work (Phase - II)	PCC	Project	0	0	16	8	30	70	100
TOTAL				**09**	**00**	**16**	**17**	**120**	**280**	**400**

PROFESSIONAL ELECTIVES

Course code	PE - I	Course code	PE - II	Course code	PE - III
	Advanced Communications		**Data Processing Systems**		**Communication Networks**
AECC22	Cellular and Mobile Communications	AECC34	Information Theory and Coding Techniques	AECC45	Adhoc Wireless Sensor Networks
AECC23	Optical Communications	AECC35	Software radio and Cognitive radio	AECC46	Artificial Neural Networks
AECC24	Satellite Communications	AECC36	Mobile Communication System	AECC47	Wireless Sensor Networks
AECC25	Wireless Communications and Networks	AECC37	Computer Organization	AECC48	RF Circuit Design

Course code	PE - IV	Course code	PE - V	Course code	PE - VI
	VLSI		**Signal And Image Processing**		**Embedded Systems**
AECC49	Digital design through Verilog	AECC56	Digital Image Processing	AECC60	Microcontrollers and Applications
AECC50	Scripting Languages for VLSI Design	AECC57	Signal Processing for Communication and Biomedical Applications	AECC61	Embedding Sensors and Motors
AECC51	Microelectronics	AECC58	Wavelets and Applications	AECC62	Internet of Things
AECC52	Advanced Programmable Logic Device Architectures	AECC59	Digital Signal Processors and Architectures	AECC63	Foundations of Robot Motion and Robot Dynamics

OPEN ELECTIVES COURSES

OPEN ELECTIVES – I

Course Code	Course Title
ACSC24	Computer Architecture
ACSC25	Advanced Data Structures
ACSC26	Artificial Intelligence
AITC19	Cyber Crime and Computer Forensics
AITC20	Ethical Hacking
AITC21	Mobile Computing

OPEN ELECTIVE - II

Course Code	Course Title
AHSC15	Soft Skills and Interpersonal Communication
AHSC16	Cyber Law and Ethics
AHSC17	Economic Policies in India
AHSC18	Global Warming and Climate Change
AHSC19	Intellectual Property Rights
AHSC20	Entrepreneurship

OPEN ELECTIVE - III

Course Code	Course Title
AAEC30	Flight Control Theory
AAEC31	Airframe Structural Design
AMEC34	Industrial Management
AMEC35	Elements of Mechanical Engineering
ACEC30	Modern Construction Materials
ACEC31	Disaster Management

VALUE ADDED COURSES / MANDATORY COURSES

Course Code	Course Title
AHSC10	Essence of Indian Traditional Knowledge (MC)
ACSC18	Fundamentals of Database Systems (VAC)
ACSC23	Object Oriented Programming Development and Languages (VAC)
ACSC29	Design of Algorithms (VAC)

SYLLABUS
(I - VIII SEMESTERS)

ENGLISH

I Semester: AE / ECE / EEE / ME / CE

II Semester : CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC01	Foundation	2	-	-	2	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisite: Standard applicability of vocabulary and grammer								

I. COURSE OVERVIEW:

The sole aim of the course is to enhance the communication skills of upcoming engineering graduates to meet the requirements and challenges in a competitive global world. This course is designed to provide a well-rounded introduction to English language learning. Moreover, the course pays special attention to the typical problems and challenges confronted by the Indian learners of English like mispronunciation, spellings, and structures of English due to their mother tongue influence. This course includes General Introduction to Listening Skills, Speaking Skills, Vocabulary and Grammar, Reading Skills, and Writing Skills.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The theoretical and fundamental inputs to communicate intelligibly in English through standard Pronunciation.
II. The four language skills i.e., Listening, Speaking, Reading and Writing effectively and their application in real-life situations.
III. The Writing strategies of English using correct spelling, grammar, punctuation and appropriate vocabulary.
IV. Different mechanics of writing styles forms of writing emails, reports, formal and informal letters.

III. COURSE SYLLABUS:

MODULE-I: GENERAL INTRODUCTION AND LISTENING SKILLS (09)

Introduction to communication skills; Communication process; Elements of communication; Soft skills vs hard skills; Listening skills; Significance; Stages of listening; Barriers to listening and effectiveness of listening; Listening comprehension.

MODULE –II: SPEAKING SKILLS (09)

Significance; Essentials; Barriers and effectiveness of speaking; Verbal and non-verbal communication; Generating talks based on visual prompts; Public speaking; Exposure to structured talks; Addressing a small group or a large formal gathering; Oral presentation.

MODULE –III: VOCABULARY & GRAMMAR (09)

Vocabulary: The concept of Word Formation; Root words from foreign languages and their use in English;
Acquaintance with prefixes and suffixes from foreign languages in English to form derivatives;
Idioms and phrases; One-word substitutes.

Grammar: Sentence structure; Uses of phrases and clauses; Punctuation; Subject verb agreement; Modifiers; Articles; Prepositions.

MODULE –IV: READING SKILLS (09)

Significance; Techniques of reading; Skimming-Reading for the gist of a text; Scanning - Reading for specific information; Intensive; Extensive reading; Reading comprehension; Reading for information transfer; Text to diagram; Diagram to text.

MODULE –V: WRITING SKILLS (09)

Significance; Effectiveness of writing; Organizing principles of Paragraphs in documents; Techniques for writing precisely; Letter writing; Formal and Informal letter writing; E-mail writing, Report Writing.

IV. TEXT BOOKS:

1. Handbook of English for Communication (Prepared by Faculty of English, IARE).

V. REFERENCE BOOKS:

1. Sanjay Kumar and Pushp Lata. “Communications Skills”. Oxford University Press. 2011.
2. Michael Swan. “Practical English Usage”, Oxford University Press, 1995.
3. F.T. Wood. “Remedial English Grammar”. Macmillan. 2007.
4. William Zinsser. “On Writing Well”. Harper Resource Book, 2001.
5. Raymond Murphy, “Essential English Grammar with Answers”, Cambridge University Press 2nd Edition, 2011.

VI. WEB REFERENCES:

1. www.edufind.com
2. www.myenglishpages.com
3. http:grammar.ccc.comment.edu
4. http:owl.english.prudue.edu

VII. E-TEXT BOOKS:

1. http://bookboon.com/en/communication-ebooks-zip
2. http://www.bloomsbury-international.com/images/ezone/ebook/writing-skills-pdf.pdf
3. https://americanenglish.state.gov/files/ae/resource_files/developing_writing.pdf
4. http://learningenglishvocabularygrammar.com/files/idiomsandphraseswithmeaningsandexamplespdf
5. http://www.robinwood.com/Democracy/GeneralEssays/CriticalThinking.pdf

LINEAR ALGEBRA AND CALCULUS

I Semester: Common for All Branches								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AHSC02	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	1	-	4	30	70	100
Contact Classes: 45	**Tutorial Classes: 15**	**Practical Classes: Nil**				**Total Classes: 60**		
Prerequisite: Basic principles of algebra and calculus								

I. COURSE OVERVIEW:

Linear algebra is a sub-field of mathematics concerned with vectors, matrices, and linear transforms. Calculus is the branch of mathematics which majorly deals with derivatives and integrals. Linear algebra is a key foundation to the field of machine learning. Matrices are used in computer animations, color image processing. Eigenvalues are used by engineers to discover new and better designs for the future. Differential equations have wide applications in various engineering and science disciplines as the laws of physics are generally written down as differential equations.The Fourier series has many applications in electrical engineering, image processing etc.The course includes types of Matrices, Rank, methods of finding rank, eigen values and eigen vectors, maxima and minima of functions of several variables, solutions of higher order ordinary differential equations and Fourier series.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The principles of Eigen value analysis and linear transformations, Matrix rank finding methods

II. The calculus of functions of several variables and the concept of maxima-minima for a three-dimensional surface.

III. The analytical methods for solving higher order differential equations with constant coefficients.

IV. Fourier series expansions in standard intervals as well as arbitrary intervals.

III. COURSE SYLLABUS:

MODULE-I: THEORY OF MATRICES (09)

Real matrices: Symmetric, skew-symmetric and orthogonal matrices; Complex matrices: Hermitian, Skew- Hermitian and unitary matrices; Elementary row and column transformations, finding rank of a matrix by reducing to Echelon form and Normal form; Finding the inverse of a matrix using Gauss-Jordan method;

MODULE –II: LINEAR TRANSFORMATIONS (09)

Cayley-Hamilton theorem: Statement, verification, finding inverse and powers of a matrix; Linear dependence and independence of vectors; Linear transformation; Eigen values and Eigen vectors of a matrix; Diagonalization of matrix by linear transformation.

MODULE –III: FUNCTIONS OF SINGLE AND SEVERAL VARIABLES (09)

Mean value theorems: Rolle’s theorem, Lagrange’s theorem, Cauchy’s theorem-without proof.

Functions of several variables: Partial differentiation, Jacobian, functional dependence, maxima and minima of functions with two variables and three variables. Method of Lagrange multipliers.

MODULE –IV: HIGHER ORDER LINEAR DIFFERENTIAL EQUATIONS (09)

Linear differential equations of second and higher order with constant coefficients.

Non-homogeneous term of the type $f(x)=e^{ax}, \sin ax, \cos ax$ and $f(x)=x^{n}, e^{ax}v(x)$, Method of variation of parameters.

MODULE –V: FOURIER SERIES (09)

Fourier expansion of periodic function in a given interval of length 2π; Fourier series of even and odd functions; Fourier series in an arbitrary interval, Half- range Fourier sine and cosine expansions.

IV. TEXT BOOKS

1. B.S. Grewal, Higher Engineering Mathematics, Khanna Publishers, 36th Edition, 2010.
2. N.P. Bali and Manish Goyal, A text book of Engineering Mathematics, Laxmi Publications, Reprint, 2008.
3. Ramana B.V., Higher Engineering Mathematics, Tata McGraw Hill New Delhi, 11th Reprint 2010.

V. REFERENCE BOOKS:

1. Erwin Kreyszig, Advanced Engineering Mathematics, 9th Edition, John Wiley & Sons, 2006.
2. Veerarajan T., Engineering Mathematics for first year, Tata McGraw-Hill, New Delhi, 2008.
3. D. Poole, Linear Algebra: A Modern Introduction, 2nd Edition, Brooks/Cole, 2005.

VI. WEB REFERENCES:

1. http://www.efunda.com/math/math_home/math.cfm
2. http://www.ocw.mit.edu/resourcs/#Mathematics
3. http://www.sosmath.com
4. http://www.mathworld.wolfram.com

VII. E-TEXT BOOKS:

1. http://www.e-booksdirectory.com/details.php?ebook=10166
2. http://www.e-booksdirectory.com/details.php?ebook=7400re

ENGINEERING PHYSICS

I Semester: AE / ME / CE / EEE / ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AHSC03	**Foundation**	3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisite: Basic principles of waves								

I. COURSE OVERVIEW:

This course develops abstract and critical reasoning by studying mathematical and logical proofs and assumptions as applied in basic physics and to make connections between physics and other branches of sciences and technology. The topics covered include waves, non-dispersive transverse and longitudinal waves, light and optics, wave optics, lasers, introduction to quantum mechanics, solution of wave equation and introduction to solids and semiconductors. The course helps students to gain knowledge of basic principles and appreciate the diverse applications in technological fields in respective branches.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic formulations in wave mechanics for the evolution of energy levels and quantization of energies for a particle in a potential box with the help of mathematical description

II. The fundamental properties of semiconductors including the band gap, charge carrier concentration, doping and charge carrier transport mechanisms

III. The simple optical setups and experimental approaches of Light and Laser using its interaction with matter

IV. The basic studies between different harmonic oscillators and different waves for using those relationships on practical problems.

III. COURSE OBJECTIVES:

MODULE-I: QUANTUM MECHANICS (09)

Introduction to quantum physics, de-broglie’s hypothesis, Wave-particle duality, Davisson and Germer experiment, Time-independent Schrodinger equation for wave function, Physical significance of the wave function, Schrodinger equation for one dimensional problems–particle in a box.

MODULE –II: INTRODUCTION TO SOLIDS AND SEMICONDUCTORS (09)

Introduction to classical free electron theory and quantum theory, Bloch’s theorem for particles in a periodic potential (Qualitative treatment), Kronig-Penney model (Qualitative treatment), classification: metals, semiconductors, and insulators. Intrinsic and extrinsic semiconductors, Carrier concentration, Dependence of Fermi level on carrier-concentration and temperature, Hall effect.

MODULE –III: LASERS AND FIBER OPTICS (09)

Characteristics of lasers, Spontaneous and stimulated emission of radiation, Metastable state, Population inversion, Lasing action, Ruby laser, He-Ne laser and applications of lasers.

Principle and construction of an optical fiber, Acceptance angle, Numerical aperture, Types of optical fibers (Single mode, multimode, step index, graded index), Optical fiber communication system with block diagram and Applications of optical fibers.

MODULE –IV: LIGHT AND OPTICS (09)

Principle of superposition of waves, Young’s double slit experiment, Fringe width, Newton’s rings.
Fraunhofer diffraction from a single slit, double slit (extension to N slits) and diffraction grating experiment.

MODULE –V: HARMONIC OSCILLATIONS AND WAVES IN ONE DIMENSION (09)

Simple harmonic oscillator, Damped harmonic oscillator, Forced harmonic oscillator. Transverse waves and Longitudinal wave equation, Reflection and transmission of waves at a boundary, Harmonic waves.

IV. TEXT BOOKS:

1. G. Main, “Vibrations and Waves in Physics”, Cambridge University Press, 1993.
2. R. K. Gaur, S. L. Gupta, “Engineering Physics”, Dhanpat Rai Publications, 8th Edition, 2001.
3. Dr. K. Vijaya Kumar, Dr. S. Chandralingam, “Modern Engineering Physics”, Chand & Co. New Delhi, 1st Edition, 2010.

V. REFERENCE BOOKS:

1. H.J. Pain, “The Physics of Vibrations and Waves”, Wiley, 2006.
2. Ghatak, “Optics”, McGraw Hill Education, 2012.
3. O. Svelto, “Principles of Lasers”, Springer Science & Business Media, 2010.

VI. WEB REFERENCES:

1. http://link.springer.com/book
2. http://www.thphys.physics.ox.ac.uk
3. http://www.sciencedirect.com/science
4. http://www.e-booksdirectory.com

VII.E-TEXT BOOKS:

1. http://www.peaceone.net/basic/Feynman/
2. http://physicsdatabase.com/free-physics-books/
3. http://www.damtp.cam.ac.uk/user/tong/statphys/sp.pdf
4. www.freebookcentre.net/Physics/Solid-State-Physics-Books.html

PYTHON PROGRAMMING

I Semester: **Common for all branches**								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC01	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisites: **There are no prerequisites to take this course.**								

I. COURSE OVERVIEW:

This course introduces students to writing computer programs. This course presents the principles of structured programming using the Python language, one of the most increasingly preferred languages for programming today. Because of its ease of use, it is ideal as a first programming language and runs on both the PC and Macintosh platforms. However, the knowledge gained in the course can be applied later to other languages such as C and Java. The course uses iPython Notebook to afford a more interactive experience. Topics include fundamentals of computer programming in Python, object-oriented programming and graphical user interfaces.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Acquire programming skills in core Python.
II. Acquire Object-oriented programming skills in Python.
III. Develop the skill of designing graphical-user interfaces (GUI) in Python.
IV. Develop the ability to write database applications in Python.
V. Acquire Python programming skills to move into specific branches - Internet of Things (IoT), Data Science, Machine Learning (ML), Artificial Intelligence (AI) etc.

III. SYLLABUS:

MODULE – I: INTRODUCTION TO PYTHON (09)

Introduction to Python: Features of Python, History and Future of Python, Working with Python – interactive and script mode, Identifiers and Keywords, Comments, Indentation and Multi-lining, Data types – built-in data types, Operators and Expressions, Console Input/Output, Formatted printing, Built-in Functions, Library Functions.

MODULE – II: DECISION CONTROL STATEMENTS (09)

Selection/Conditional Branching Statements: if, if-else, nested if, if-elif-else statement(s), Basic Loop Structures/ Iterative Statements – while and for loop, Nested loops, break and continue statement, pass Statement, else Statement used with loops.

MODULE – III: CONTAINER DATA TYPES (09)

Lists: Accessing List elements, List operations, List methods, List comprehension; Tuples: Accessing Tuple elements, Tuple operations, Tuple methods, Tuple comprehension, Conversion of List comprehension to Tuple, Iterators and Iterables, zip() function.

Sets: Accessing Set elements, Set operations, Set functions, Set comprehension; Dictionaries: Accessing Dictionary elements, Dictionary operations, Dictionary Functions, Nested Dictionary, Dictionary comprehension.

MODULE - IV STRINGS AND FUNCTIONS (09)

Strings: Accessing String elements, String properties, String operations.
Functions: Communicating with functions, Variable Scope and lifetime, return statement, Types of arguments, Lambda functions, Recursive functions.

MODULE - V CLASSES AND OBJECTS (09)

Classes and Objects – Defining Classes, Creating Objects, Data Abstraction and Hiding through Classes, Class Method and self Argument, Class variables and Object variables, __init()__ and __del__() method, Public and private data members, Built-in Class Attributes, Garbage Collection. OOPs Features: Abstraction, Encapsulation, Inheritance, and Polymorphism.

IV. TEXT BOOKS:

1. Reema Thareja, “Python Programming - Using Problem Solving Approach”, Oxford Press, 1st Edition, 2017.
2. Dusty Philips, “Python 3 Object Oriented Programming”, PACKT Publishing, 2nd Edition, 2015.

V. REFERENCE BOOKS:

1. Yashavant Kanetkar, Aditya Kanetkar, “Let Us Python”, BPB Publications, 2nd Edition, 2019.
2. Martin C. Brown, “Python: The Complete Reference”, Mc. Graw Hill, Indian Edition, 2018.
3. Michael H.Goldwasser, David Letscher, “Object Oriented Programming in Python”, Prentice Hall, 1st Edition, 2007.
4. Taneja Sheetal, Kumar Naveen, “Python Programming – A Modular Approach”, Pearson, 1st Edition, 2017.
   R Nageswar Rao, “Core Python Programming”, Dreamtech Press, 2018.

VI. WEB REFERENCES:

1. https://realPython.com/Python3-object-oriented-programming/
2. https://Python.swaroopch.com/oop.html
3. https://Python-textbok.readthedocs.io/en/1.0/Object_Oriented_Programming.html
4. https://www.programiz.com/Python-programming/

ENGLISH LANGUAGE AND COMMUNICATION SKILLS LABORATORY

I SEMESTER: AE / ECE / EEE / ME / CE

II SEMESTER: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AHSC04	**Foundation**	-	-	2	1	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: 45**				**Total Classes: 45**		

Prerequisite: There are no prerequisites to take this course.

I. COURSE OVERVIEW

The sole aim of the course is to enhance the communication skills of upcoming engineering graduates to meet the requirements and challenges in a competitive global world. This course includes General Introduction to Listening Skills, Speaking Skills, Vocabulary and Grammar, Reading Skills, and Writing Skills.

II. COURSE OBJECTIVES:

The students will try to:

I. Improve their ability to listen and comprehend a given text.
II. Upgrade the fluency and acquire a functional knowledge of English Language.
III. Enrich thought process by viewing a problem through multiple angles.

III. COURSE SYLLABUS:

Week-l: LISTENING SKILL

a. Listening to conversations and interviews of famous personalities in various fields; Listening practice related to the TV talk shows and news.
b. Listening for specific information; Listening for summarizing information – Testing..

Week-2: LISTENING SKILL

a. Listening to films of short duration and monologues for taking notes; Listening to answer multiple choice questions.
b. Listening to telephonic conversations; Listening to native Indian: Abdul Kalam, British: Helen Keller and American: Barrack Obama speakers to analyze intercultural differences – Testing.

Week-3: SPEAKING SKILL

a. Functions of English Language; Introduction to pronunciation; Vowels and Consonants
b. Tips on how to develop fluency, body language and communication; Introducing oneself: Talking about yourself, others, leave taking.

Week-4: SPEAKING SKILL

a. Sounds - Speaking exercises involving the use of Vowels and Consonant sounds in different contexts;Exercises on Homophones and Homographs
b. Just a minute (JAM) session.

Week-5: SPEAKING SKILL

a. Stress patterns.
b. Situational Conversations: common everyday situations; Acting as a compere and newsreader; Greetings for differentoccasionswithfeedbackpreferablythroughvideorecording.

Week-6: READING SKILL

a. Intonation.
b. Reading newspaper and magazine articles; Reading selective autobiographies for critical commentary.

Week-7: READING SKILL

a. Improving pronunciation through tongue twisters.
b. Reading advertisements, pamphlets; Reading comprehension exercises with critical and analytical questions based on context.

Week-8: WRITING SKILL

a. Listening to inspirational short stories and Writing messages
b. Writing leaflets, Notice; Writing tasks; Flashcards – Exercises

Week-9: WRITING SKILL

a. Write the review on a video clipping of short duration (5 to 10minutes).
b. Write a slogan related to the image; Write a short story of 6-10 lines based on the hints given.

Week-10: WRITING SKILL

a. Minimising Mother Tongue interference to improve fluency through watching educational videos.
b. Writing practices – précis writing; Essay writing

Week-11: THINKING SKILL

a. Correcting common errors in day to day conversations.

Practice in preparing thinking blocks to decode diagrammatical representations into English words,expressions, idioms, proverbs.

IV. TEXT BOOK:

1. "English Language and Communication Skills" Lab Manual - Prepared by the faculty of English, IARE.

V. REFERENCE BOOKS:

1. Meenakshi Raman, Sangeetha Sharma, "Technical Communication Principles and Practices", Oxford University Press, New Delhi, 3rd Edition, 2015.
2. Rhirdion, Daniel, "Technical Communication", Cengage Learning, New Delhi, 1st Edition, 2009.

PHYSICS LABORATORY

I Semester: **AE / ME / CE / ECE / EEE**

II Semester: **CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AHSC05	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	-	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes: 36**		
Pre-Requisites: **Basic principles of Physics**								

I. COURSE OVERVIEW:

This course is designed to lay a strong foundation in Engineering Physics that forms a basis to various branches of Engineering. It helps the students to perform experiments, to correlate theory with experimental data, analyse using graphical representations and present them as part of a clear, well-organized lab report. At the end of the course, students will be able to demonstrate a working knowledge of fundamentals of Physics and communicate their ideas effectively, both orally and in writing.

II. COURSE OBJECTIVES:

The students will try to learn:

1. Experimental skills in using optical instruments to determine physical constants.
2. The real time applications of electromagnetic theory.
3. The working principles of various electronic devices.

III. COURSE SYLLABUS:

Week-1: HALL EFFECT (LORENTZ FORCE)
Determination of charge carrier density.

Week-2: MELDE'E EXPERIMENT
Determination of frequency of a given tuning fork.

Week-3: STEWART GEE'S APPARATUS
Magnetic field along the axis of current carrying coil-Stewart and Gee‛s method.

Week-4: B-H CURVE WITH CRO
To determine the energy loss per unit volume of a given magnetic material per cycle by tracing the Hysteresis loop (B-H curve).

Week-5: ENERGY GAP OF A SEMICONDUCTOR DIODE
Determination of energy gap of a semiconductor diode.

Week-6: PHOTO DIODE
Studying V-I characteristics of photo diode.

Week-7: OPTICAL FIBER
Evaluation of numerical aperture of a given optical fiber.

Week-8: WAVE LENGTH OF LASER LIGHT
Determination of wavelength of a given laser light using diffraction grating.

Week-9: PLANCK'S CONSTANT
Determination of Planck‛s constant using LED.

Week-10: LIGHT EMITTING DIODE
Studying V-I characteristics of LED

Week-11: NEWTONS RINGS
Determination of radius of curvature of a given plano-convex lens.

Week-12: SINGLE SLIT DIFFRACTION

Determination of width of a given single slit.

IV. MANUALS:

1. C. L. Arora, “Practical Physics”, S. Chand & Co., New Delhi, 3rd Edition, 2012.
2. VijayKumar, Dr.T.Radhakrishna, “Practical Physics for Engineering Students”, SM Enterprises, 2nd Edition, 2014.

V. WEB REFERENCE:

http://www.iare.ac.in

PYTHON PROGRAMMING LABORATORY

I Semester: Common from all branches								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC02	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes:36**		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This course introduces students to writing computer programs. This course presents the principles of structured programming using the Python language, one of the most increasingly preferred languages for programming today. Because of its ease of use, it is ideal as a first programming language and runs on both the PC and Macintosh platforms. However, the knowledge gained in the course can be applied later to other languages such as C and Java. The course uses iPython Notebook to afford a more interactive experience. Topics include fundamentals of computer programming in Python, object-oriented programming and graphical user interfaces.

II. COURSE OBJECTIVES:

The students will try to learn:

VI. Acquire programming skills in core Python.
VII. Acquire Object-oriented programming skills in Python.
VIII. Develop the skill of designing graphical-user interfaces (GUI) in Python.
IX. Develop the ability to write database applications in Python.
X. Acquire Python programming skills to move into specific branches - Internet of Things (IoT), Data Science, Machine Learning (ML), Artificial Intelligence (AI) etc.

III. COURSE SYLLABUS:

Week – 1: OPERATORS

a. Read a list of numbers and write a program to check whether a particular element is present or not using membership operators.
b. Read your name and age and write a program to display the year in which you will turn 100 years old.
c. Read radius and height of a cone and write a program to find the volume of a cone.
d. Write a program to compute distance between two points taking input from the user (Hint: use Pythagorean theorem)

Week – 2: CONTROL STRUCTURES

a. Read your email id and write a program to display the no of vowels, consonants, digits and white spaces in it using if…elif…else statement.
b. Write a program to create and display a dictionary by storing the antonyms of words. Find the antonym of a particular word given by the user from the dictionary using while loop.
c. Write a Program to find the sum of a Series 1/1! + 2/2! + 3/3! + 4/4! +…….+ n/n!. (Input :n = 5, Output : 2.70833)
d. In number theory, an abundant number or excessive number is a number for which the sum of its proper divisors is greater than the number itself. Write a program to find out, if the given number is abundant. (Input: 12, Sum of divisors of 12 = 1 + 2 + 3 + 4 + 6 = 16, sum of divisors 16 > original number 12)

Week – 3: LIST

a. Read a list of numbers and print the numbers divisible by x but not by y (Assume x = 4 and y = 5).
b. Read a list of numbers and print the sum of odd integers and even integers from the list.(Ex: [23, 10, 15, 14, 63], odd numbers sum = 101, even numbers sum = 24)
c. Read a list of numbers and print numbers present in odd index position. (Ex: [10, 25, 30, 47, 56, 84, 96], The numbers in odd index position: 25 47 84).
d. Read a list of numbers and remove the duplicate numbers from it. (Ex: Enter a list with duplicate elements: 10 20 40 10 50 30 20 10 80, The unique list is: [10, 20, 30, 40, 50, 80])

Week – 4: TUPLE

a. Given a list of tuples. Write a program to find tuples which have all elements divisible by K from a list of tuples. test_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)], K = 6, Output : [(6, 24, 12), (60, 12, 6)]
b. Given a list of tuples. Write a program to filter all uppercase characters tuples from given list of tuples. (Input: test_list = [("GFG", "IS", "BEST"), ("GFg", "AVERAGE"), ("GfG",), ("Gfg", "CS")], Output : [('GFG', 'IS', 'BEST')]).
c. Given a tuple and a list as input, write a program to count the occurrences of all items of the list in the tuple. (Input : tuple = ('a', 'a', 'c', 'b', 'd'), list = ['a', 'b'], Output : 3)

Week – 5: SET

a. Write a program to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).
b. Write a program to perform union, intersection and difference using Set A and Set B.
c. Write a program to count number of vowels using sets in given string (Input : "Hello World", Output: No. of vowels : 3)
d. Write a program to form concatenated string by taking uncommon characters from two strings using set concept (Input : S1 = "aacdb", S2 = "gafd", Output : "cbgf").

Week – 6: DICTIONARY

a. Write a program to do the following operations:
   i. Create a empty dictionary with dict() method
   ii. Add elements one at a time
   iii. Update existing key's value
   iv. Access an element using a key and also get() method
   v. Deleting a key value using del() method
b. Write a program to create a dictionary and apply the following methods:
   i. pop() method
   ii. popitem() method
   iii. clear() method
c. Given a dictionary, write a program to find the sum of all items in the dictionary.
d. Write a program to merge two dictionaries using update() method.

Week – 7: STRINGS

a. Given a string, write a program to check if the string is symmetrical and palindrome or not. A string is said to be symmetrical if both the halves of the string are the same and a string is said to be a palindrome string if one half of the string is the reverse of the other half or if a string appears same when read forward or backward.
b. Write a program to read a string and count the number of vowel letters and print all letters except 'e' and 's'.
c. Write a program to read a line of text and remove the initial word from given text. (Hint: Use split() method, Input : India is my country. Output : is my country)
d. Write a program to read a string and count how many times each letter appears. (Histogram).

Week – 8: USER DEFINED FUNCTIONS

a. A generator is a function that produces a sequence of results instead of a single value. Write a generator function for Fibonacci numbers up to n.
b. Write a function merge_dict(dict1, dict2) to merge two Python dictionaries.
c. Write a fact() function to compute the factorial of a given positive number.
d. Given a list of n elements, write a linear_search() function to search a given element x in a list.

Week – 9: BUILT-IN FUNCTIONS

a. Write a program to demonstrate the working of built-in statistical functions mean(), mode(), median() by importing statistics library.
b. Write a program to demonstrate the working of built-in trignometric functions sin(), cos(), tan(), hypot(), degrees(), radians() by importing math module.
c. Write a program to demonstrate the working of built-in Logarithmic and Power functions exp(), log(), log2(), log10(), pow() by importing math module.
d. Write a program to demonstrate the working of built-in numeric functions ceil(), floor(), fabs(), factorial(), gcd()

by importing math module.

Week – 10: CLASS AND OBJECTS

a. Write a program to create a BankAccount class. Your class should support the following methods for
   i) Deposit
   ii) Withdraw
   iii) GetBalanace
   iv) PinChange
b. Create a SavingsAccount class that behaves just like a BankAccount, but also has an interest rate and a method that increases the balance by the appropriate amount of interest (Hint:use Inheritance).
c. Write a program to create an employee class and store the employee name, id, age, and salary using the constructor. Display the employee details by invoking employee_info() method and also using dictionary (__dict__).
d. Access modifiers in Python are used to modify the default scope of variables. Write a program to demonstrate the 3 types of access modifiers: public, private and protected.

Week – 11: MISCELLANEOUS PROGRAMS

a. Write a program to find the maximum and minimum K elements in Tuple using slicing and sorted() method (Input: test_tup = (3, 7, 1, 18, 9), k = 2, Output: (3, 1, 9, 18))
b. Write a program to find the size of a tuple using getsizeof() method from sys module and built-in __sizeof__() method.
c. Write a program to check if a substring is present in a given string or not.
d. Write a program to find the length of a string using various methods:
   i. Using len() method
   ii. Using for loop and in operator
   iii. Using while loop and slicing

Week – 12: ADDITIONAL PROGRAMS - FILE HANDLING

1. Write a program to read a filename from the user, open the file (say firstFile.txt) and then perform the following operations:
   i. Count the sentences in the file.
   ii. Count the words in the file.
   iii. Count the characters in the file.
2. Create a new file (Hello.txt) and copy the text to other file called target.txt. The target.txt file should store only lower case alphabets and display the number of lines copied.
3. Write a Python program to store N student's records containing name, roll number and branch. Print the given branch student's details only.

IV. REFERENCE BOOKS:

1. Michael H Goldwasser, David Letscher, "Object Oriented Programming in Python", Prentice Hall, 1st Edition, 2007.
2. Yashavant Kanetkar, Aditya Kanetkar, "Let us Python", BPB publication, 1st Edition, 2019.
3. Ashok Kamthane, Amit Kamthane, "Programming and Problem Solving with Python", McGraw Hill Education (India) Private Limited, 2018.
4. Taneja Sheetal, Kumar Naveen, "Python Programming – A modular approach", Pearson, 2017.
5. R Nageswara Rao, "Core Python Programming", Dreamtech press, 2017 Edition.

V. WEB REFERENCES:

1. https://realpython.com/python3-object-oriented-programming/
2. https://python.swaroopch.com/oop.html
3. https://python-textbok.readthedocs.io/en/1.0/Object_Oriented_Programming.html
4. https://www.programiz.com/python-programming/
5. https://www.geeksforgeeks.org/python-programming-language/

CHEMISTRY

I Semester: CSE / CSE (AI&ML) / CSE (DS) / CSE (CS) / / CSIT / IT								
II Semester: AE / ME / CE / ECE / EEE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AHSC06	**Foundation**	2	-	-	2	30	70	100
Contact Classes: 45	**Tutorial Classes: 0**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

The concepts developed in this course involve elements and compounds and their applied industrial applications. It deals with topics such as batteries, corrosion and control of metallic materials, water and its treatment for different purposes, engineering materials such as plastics, elastomers and biodegradable polymers, their preparation, properties and applications, energy sources and environmental science. Sustainable chemistry that focuses on the design of the products and processes that minimize or eliminate the use and generation of hazardous substances is also included.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The concepts of electrochemical principles and causes of corrosion in the new developments and breakthroughs efficiently in engineering and technology.

II. The different parameters to remove causes of hardness of water and their reactions towards complexometric method.

III. The polymerization reactions with respect to mechanisms and its significance in industrial applications.

IV. The Significance of Green chemistry to reduce pollution in environment by using natural resources.

III. COURSE SYLLABUS

MODULE-I: ELECTROCHEMISTRY AND CORROSION (09)

Electro chemical cells: Electrode potential, standard electrode potential, Calomel electrode and Nernst equation; Electrochemical series and its applications; Numerical problems; Batteries: Primary (Dry cell) and secondary batteries (Lead-acid storage battery, Li-ion battery). Corrosion: Causes and effects of corrosion: Theories of chemical and electrochemical corrosion, mechanism of electrochemical corrosion; Corrosion control methods: Cathodic protection, sacrificial anode and impressed current Cathodic protection; Surface coatings: Metallic coatings- Methods of coating- Hot dipping- galvanization and tinning, electroplating.

MODULE –II: WATER TECHNOLOGY (09)

Introduction: Hardness of water, causes of hardness; types of hardness: temporary and permanent hardness, expression and units of hardness; estimation of hardness of water by complexometric method; potable water and its specifications, Steps involved in the treatment of water, disinfection of water by chlorination and ozonization; External treatment of water; Ion-exchange process; Desalination of water: Reverse osmosis, numerical problems.

MODULE-III: ENGINEERING MATERIALS (09)

Polymers-classification with examples, polymerization-addition, condensation and co-polymerization;
Plastics: Thermoplastics and thermosetting plastics; Compounding of plastics; Preparation, properties and applications of polyvinyl chloride, Teflon, Bakelite and Nylon-6, 6; Elastomers: Natural rubber, processing of natural rubber, vulcanization; Buna-s and Thiokol rubber; Biodegradable polymers.

Lubricants: characteristics of lubricants, mechanism of lubrication – thick film, thin film, extreme pressure lubrication, properties – flash and fire point, cloud and pour point, viscosity and oiliness of lubricants.

MODULE –IV: GREEN CHEMISTRY AND FUELS (09)

Introduction: Definition of green chemistry, methods of green synthesis: aqueous phase, microwave method, phase transfer catalyst and ultra sound method. Fuels: definition, classification of fuels ; Solid fuels: coal; analysis of coal: proximate and ultimate analysis; Liquid fuels: Petroleum and its refining; Gaseous fuels: Composition, characteristics and applications of LPG and CNG; Calorific value: Gross Calorific value(GCV) and Net Calorific value(NCV), numerical problems.

MODULE –V: NATURAL RESOURCES AND ENVIRONMENTAL POLLUTION (09)

Natural resources: Classification of resources, living and nonliving resources; Water resources: Use and over utilization of surface and ground water, floods and droughts, dams, benefits and problems; Land resources; Energy resources: renewable and non-renewable energy sources, use of alternate energy source. Environmental pollution: Causes, effects and control measures of air pollution, water pollution, soil pollution and noise pollution.

IV. TEXT BOOKS:

1. P. C. Jain and Monica Jain, “Engineering Chemistry”, DhanpatRai Publishing Company, 16th Edition, 2017.
2. ShashiChawla, “Text Book of Engineering Chemistry” DhanatRai and Company, 2017.
3. Prashanthrath, B.Rama Devi, Ch.VenkataRamana Reddy, Subhendu Chakroborty, Cengage Learning Publishers, 1st Edition, 2018.

V. REFERENCE BOOKS:

1. Bharathi Kumari, “Engineering Chemistry”, VGS Book Links, 10th Edition, 2018.
2. B. Siva Shankar, “Engineering Chemistry”, Tata McGraw Hill Publishing Limited, 3rd Edition, 2015.
3. S. S. Dara, Mukkanti, “Text of Engineering Chemistry”, S. Chand & Co, New Delhi, 12th Edition, 2006.

VI. WEB REFERENCES:

1. Engineering chemistry (NPTEL Web-book), by B.L.Tembe, Kamaluddin and M.S.Krishnan. http://www.cdeep.iitb.ac.in/webpage_data/nptel/Core%20Science/Engineering%20Chemistry%201/About-Faculty.html
2. Polymer Science (NPTEL Web-book), by Prof. Dibakar Dhara https://onlinecourses.nptel.ac.in/noc20_cy21/preview
3. Environmental Chemistry and Analysis(NPTEL Web-book), by Prof. M.S.Subramanian https://nptel.ac.in/courses/122/106/122106030/

MATHEMATICAL TRANSFORM TECHNIQUES

II Semester: AE / ME / CE / ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC07	Foundation	3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisite: Basic principles of calculus

I. COURSE OVERVIEW:

This course focuses on transformations from theoretical based mathematical laws to its practical applications in the domain of various branches of engineering field. The course includes the transformations such as Laplace, Fourier, applications of scalar and vector field over surface, volume and multiple integrals. The course is designed to extract the mathematical developments, skills, from basic concepts to advance level of engineering problems to meet the technological challenges.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The transformation of ordinary differential equations in Laplace field and its applications.
II. The operation of the non-periodic functions by Fourier transforms.
III. The concepts of *multiple integration for finding* areas and volumes of physical quantities.
IV. The Integration of the several functions by transforming the co-ordinate system in scalar and vector fields.

III. COURSE SYLLABUS

MODULE-I: LAPLACE TRANSFORMS (09)

Definition of Laplace transform, linearity property, piecewise continuous function, existence of Laplace transform, function of exponential order, first and second shifting theorems, change of scale property, Laplace transforms of derivatives and integrals, multiplied by t, divided by t, Laplace transform of periodic functions.

Inverse Laplace transform: Definition of Inverse Laplace transform, linearity property, first and second shifting theorems, change of scale property, multiplied by s, divided by s; Convolution theorem and applications to ordinary differential equations.

MODULE –II: FOURIER TRANSFORMS (09)

Fourier integral theorem, Fourier sine and cosine integrals; Fourier transforms; Fourier sine and cosine transform, properties, inverse transforms, finite Fourier transforms. .

MODULE –III: MULTIPLE INTEGRALS (09)

Double Integrals: Evaluation of double integrals in Cartesian coordinates and Polar coordinates; Change of order of integration; Area as a double integral; Transformation of coordinate system.

Triple Integrals: Evaluation of triple integrals in Cartesian coordinates; volume of a region using triple integration.

MODULE –IV: VECTOR DIFFERENTIAL CALCULUS (09)

Scalar and vector point functions; Definitions of Gradient, divergent and curl with examples; Solenoidal and irrotational vector point functions; Scalar potential function. Line integral, surface integral and volume integral, Green‘s theorem in a plane, Stoke‘s theorem and Gauss divergence theorem without proofs.

MODULE –V: PARTIAL DIFFERENTIAL EQUATIONS (09)

Formation of partial differential equations by elimination of arbitrary constants and arbitrary functions, solutions of first order linear equations; Charpit's method;

IV. TEXT BOOKS:

1. B.S. Grewal, "Higher Engineering Mathematics", Khanna Publishers, 36th Edition, 2010.
2. N.P. Bali and Manish Goyal, "A text book of Engineering Mathematics", Laxmi Publications, Reprint, 2008.
3. Ramana B.V., "Higher Engineering Mathematics", Tata McGraw Hill New Delhi, 11th Reprint, 2010

V. REFERENCE BOOKS:

1. Erwin Kreyszig, “Advanced Engineering Mathematics”, John Wiley & Sons, 9thEdition, 2006.
2. Veerarajan T., “Engineering Mathematics for first year”, Tata McGraw-Hill, New Delhi, 2008.
3. D. Poole, “Linear Algebra: A Modern Introduction”, Brooks/Cole, 2ndEdition, 2005.
4. Dr. M Anita, “Engineering Mathematics-I”, Everest Publishing House, Pune, First Edition, 2016.

VI. WEB REFERENCES:

1. http://www.efunda.com/math/math_home/math.cfm
2. http://www.ocw.mit.edu/resourcs/#Mathematics
3. http://www.sosmath.com
4. http://www.mathworld.wolfram.com.

VII. E-TEXT BOOKS:

1. http://www.e-booksdirectory.com/details.php?ebook=10166
2. http://www.e-booksdirectory.com/details.php?ebook=7400re

ELECTRICAL CIRCUITS

II Semester: EEE / ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AEEC02	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisites: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

The course introduces the basic concepts of circuit analysis which is the foundation for all subjects of the electrical and electronics engineering. It includes the basic fundamental laws of electricity and magnetism with an emphasis on resistors, inductors and capacitors (RLC) circuits applied to alternating current (AC) or direct current (DC) of electrical networks. Further This course provides network theorems with different excitations, two-port network and network topology to solve for real-time applications.

II. COURSE OBJECTIVES:

The students will try to learn

I. The network reduction techniques such as source transformation, mesh analysis, nodal analysis and network theorems to solve different networks.

II. The basic concept of AC circuits for optimization of household and industrial circuitry.

III. The various configurations of electromagnetic induction used in magnetic circuits helps in the winding of electrical machines.

IV. The characteristics of two-port networks and network topologies suitable in power system.

III. COURSE OBJECTIVES:

MODULE-I: INTRODUCTION TO ELECTRICAL CIRCUITS (09)

Circuit concept: Basic definitions, Ohm’s law at constant temperature, classifications of elements, independent and dependent sources, voltage and current relationships for passive elements,

Single phase AC circuits: Representation of alternating quantities, properties of different periodic wave forms, phase and phase difference, concept of impedance and admittance, power in AC circuits.

MODULE-II: ANALYSIS OF ELECTRICAL CIRCUITS (09)

Circuit analysis: Source transformation, Kirchhoff’s laws, total resistance, inductance and capacitance of circuits, Star - delta transformation technique, mesh analysis and nodal analysis, inspection method, super mesh, super node analysis.

MODULE-III: NETWORK THEOREMS (DC AND AC) (10)

Network Theorems: Tellegen’s, superposition, reciprocity, Thevenin’s, Norton’s, maximum power transfer, Milliman’s and compensation theorems for DC excitations, numerical problems.

Network Theorems: Tellegen’s, superposition, reciprocity, Thevenin’s, Norton’s, maximum power transfer, Milliman’s and compensation theorems for AC excitations, numerical problems.

MODULE-IV: MAGNETIC CIRCUITS (09)

Magnetic circuits: Faraday’s laws of electromagnetic induction, concept of self and mutual inductance, dot convention, coefficient of coupling, composite magnetic circuit, analysis of series and parallel magnetic circuits.

MODULE-V: TWO PORT NETWORK AND GRAPH THEORY (08)

Two Port Network: Two port parameters, interrelations, Two port Interconnections.

Network topology: Definitions, incidence matrix, basic tie set and basic cut set matrices for planar networks, duality and dual networks.

IV. TEXT BOOKS:

1. A Sudhakar, Shyammohan S Palli, “Circuits and Networks”, Tata McGraw-Hill, 4th Edition, 2010.
2. M E Van Valkenberg, “Network Analysis”, PHI, 3rd Edition, 2014.

V. REFERENCE BOOKS:

1. John Bird, “Electrical Circuit Theory and Technology”, Newnes, 2nd Edition, 2003.
2. C L Wadhwa, “Electrical Circuit Analysis including Passive Network Synthesis”, New Age International, 2nd Edition, 2009.
3. David A Bell, “Electric circuits”, Oxford University Press, 7th Edition, 2009.
4. E Hughes, “Electrical and Electronics Technology”, Pearson Education, 2010.
5. A Chakrabarthy, “Electric Circuits”, Dhanipat Rai & Sons, 6th Edition, 2010.
6. V D Toro, “Electrical Engineering Fundamentals”, Prentice Hall India, 1989.

VI. WEB REFERENCES:

1. https://www.igniteengineers.com
2. https://www.ocw.nthu.edu.tw
3. https://www.uotechnology.edu.iq
4. https://www.iare.ac.in

VII. E-TEXT BOOKS:

1. https://www.bookboon.com/en/concepts-in-electric-circuits-ebook
2. https://www.www.jntubook.com
3. https://www.allaboutcircuits.com
4. https://www.archive.org

PROGRAMMING FOR PROBLEM SOLVING USING C

II Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC04	Foundation	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisite: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

The main emphasis of the course will be on problem solving aspects in through C programming. The students will understand programming language, programming, concepts of loops, reading a set of data, step wise refinements, functions, control structures, arrays, dynamic memory allocations, enumerated data types, structures, unions, and file handling. This course provides adequate knowledge to solve problems in their respective domains.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Problem-solving through programming

II. Programming language, programming, reading a set of Data, stepwise refinement, concepts of Loops, Functions, Control structure, Arrays, Structure, Pointer and File concept.

III. To build efficient programs in ‘C’ language essential for future programming and software engineering courses.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION (10)

Introduction to components of a computer: Memory, processor, I/O Devices, storage, operating system; Concept of assembler, compiler, interpreter, loader and linker.

Idea of Algorithms: Algorithms, Flowcharts, Pseudo code with examples, From algorithm to Programs and Source Code

Introduction to C Programming Language: History of C, Basic structure of a C program, Process of compiling and running a C program; C Tokens: Keywords, Identifiers, Constants, Strings, Special symbols, Variables, Data types; Operators, Precedence of Operators, Expression evaluation, Formatted Input/Output functions, Type Conversion and type casting.

MODULE-II: CONTROL STRUCTURES (08)

Decision Making Statements: Simple if, if-else, else if ladder, Nested if, switch case statement;
Loop control statements: for, while and do while loops, nested loops;
Unconditional Control Structures: break, continue and goto statements.

MODULE-III: ARRAYS AND FUNCTIONS (10)

Arrays: Introduction, Single dimensional array and multi-dimensional array: declaration, initialization, accessing elements of an array; Operations on arrays: traversal, reverse, insertion, deletion, merge, search; **Strings:** Arrays of characters, Reading and writing strings, String handling functions, Operations on strings; array of strings.

Functions: Concept of user defined functions, Function declaration, return statement, Function prototype, Types of functions, Inter function communication, Function calls, Parameter passing mechanisms; Recursion; Passing arrays to functions, passing strings to functions; Storage classes.

MODULE-IV: POINTER AND STRUCTURES (10)

Pointers: Basics of pointers, Pointer arithmetic, pointer to pointers, array of pointers, Generic pointers, Null pointers, Pointers as functions arguments, Functions returning pointers; Dynamic memory allocation.
Structures: Structure definition, initialization, structure members, nested structures, arrays of structures, structures and functions, structures and pointers, self-referential structures; Unions: Union definition, initialization, accessing union members; bit fields, typedef, enumerations, Preprocessor directives.

MODULE-V: FILE HANDLING AND APPLICATIONS IN C (07)

File Handling: Concept of a file, text files and binary files, streams, standard I/O, formatted I/O, file I/O operations, error handling, Line I/O, miscellaneous functions; Applications in C.

IV.TEXT BOOKS:

1. Byron Gottfried, “Programming with C”, Schaum's Outlines Series, McGraw Hill Education, 3rd Edition, 2017.
2. Reema Thareja, “Programming in C”, Oxford university press, 2nd Edition, 2016.

V. REFERENCE BOOKS:

I. W. Kernighan Brian, Dennis M. Ritchie, “The C Programming Language”, PHI Learning, 2nd Edition, 1988.
II. Yashavant Kanetkar, “Exploring C”, BPB Publishers, 2nd Edition, 2003.
III. Schildt Herbert, “C: The Complete Reference”, Tata McGraw Hill Education, 4th Edition, 2014.
IV. R. S. Bichkar, “Programming with C”, Universities Press, 2nd Edition, 2012.
V. Dey Pradeep, Manas Ghosh, “Computer Fundamentals and Programming in C”, Oxford University Press, 2nd Edition, 2006.
VI. Stephen G. Kochan, “Programming in C”, Addison-Wesley Professional, 4th Edition, 2014.

VI.WEB REFERENCES:

1. https://www.calvin.edu/~pribeiro/courses/engr315/EMFT_Book.pdf
2. https://www.web.mit.edu/viz/EM/visualizations/coursenotes/modules/guide02.pdf
3. https://www.nptel.ac.in/courses/108106073/
4. https://www.iare.ac.in

VII.E-TEXT BOOKS:

1. http://www.freebookcentre.net/Language/Free-C-Programming-Books-Download.htm
2. http://www.imada.sdu.dk/~svalle/courses/dm14-2005/mirror/c/
3. http://www.enggnotebook.weebly.com/uploads/2/2/7/1/22718186/ge6151-notes.pdf

EXPERIENTIAL ENGINEERING EDUCATION (ExEEd) - ACADEMIC SUCCESS

I Semester: CSE / CSE (AI&ML) / CSE (DS) / CSE(CS) / IT / CSIT

II Semester: AE / ME / CE / ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC06	Foundation	L	T	P	C	CIA	SEE	Total
		2	-	-	1	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes: 36		

Prerequisite: There are no prerequisites to take this course

I.COURSE OVERVIEW:

The course aims to provide students with an understating of the different learning –coding platforms, role of the entrepreneur, innovation and technology in customer centric engineering.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The different ways in engaging continuous learning process through the interaction with peers in related topics.
II. The skills and potential opportunities using well know frameworks and analytical tools.
III. The attitudes, values, characteristics, behavior and processes with processing an entrepreneurial mindset.

III. COURSE OBJECTIVES:

WEEK – I
Introduction to ExEED - Dr. Ch. Srinivasulu

WEEK – II:
Skill Development - Dr. G Ramu

WEEK – III:
Skill Development - Dr. G Ramu

WEEK – IV:
Open Source platforms for Learning , Practice and Excel in their field - Dr. M MadhuBala

WEEK – V:
Opportunities and challenges - Respective Department HOD's

WEEK – VI:
Skill Development - Dr. G Ramu

WEEK – VII:
Skill Development - Dr. G Ramu

WEEK –VIII:
Entrepreneurial Mindset - Dr. J Sirisha Devi

WEEK – IX:
Entrepreneurial Mindset - Dr. J Sirisha Devi

WEEK – X:
Innovation Culture - Dr. M Pala Prasad Reddy

WEEK – XI:
Support & Funding from various organizations - Dr. M Pala Prasad Reddy

WEEK – XII:

Rapid Prototyping - Prof. V V S H Prasad

WEEK – XIII:
Intellectual Property Rights - Mr. K Aditya Nag

WEEK – XIV:
Story Telling by Students - Dr. Ch. Srinivasulu

ELECTRICAL CIRCUITS LABORATORY

II Semester: ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AEEC03	Foundation	L	T	P	C	CIA	SEE	Total
		-	-	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes: 36		

Prerequisites: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

Electrical circuits laboratory examines the basic laws, network reduction techniques, network theorems, characteristics of AC and two port network, design of transformer, measurement of electrical parameters. It includes the basic concepts of MATLAB.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic law, network reduction techniques and network theorems for circuit analysis.
II. The characteristics of AC and two port networks.
III. The measurement of electrical quantities using various electrical devices.
IV. To built the prototype of transformer and study its properties.

III. COURSE SYLLABUS:

Expt. 1: VERIFICATION OF OHM'S LAW AND KIRCHOFF LAWS

Draw the V-I characteristics of resistor element, examine voltage and current division in an electrical circuit using hardware and digital simulation.

Expt. 2: MESH ANALYSIS

Determination of mesh currents in complex electrical circuit using hardware and digital simulation.

Expt. 3: NODAL ANALYSIS

Determination of nodal voltages in complex electrical circuit using hardware and digital simulation.

Expt. 4: CHARECTERISTICS OF PERIODIC WAVEFORMS

Calculate Instantaneous, Peak, Peak to peak, Average and RMS values of periodic wave form using hardware and digital simulation.

Expt. 5: DETERMINATION OF CIRCUIT IMPEDANCE

Find the impedance of series RL, RC and RLC circuits using hardware and digital simulation.

Expt. 6: THEVENIN'S THEOREM

Determine load or unknown current using Thevenin's equivalent circuit using hardware and digital simulation.

Expt. 7: NORTON'S THEOREM

Determine load or unknown current using Norton's equivalent circuit using hardware and digital simulation.

Expt. 8: SUPERPOSITION THEOREM

Verify of superposition theorem using hardware and digital simulation.

Expt. 9: RECIPROCITY THEOREM

Verify of reciprocity theorem using hardware and digital simulation.

Expt. 10: SERIES AND PARALLEL RESONANCE

Verification of series and parallel resonance using hardware and digital simulation.

Expt. 11: MEASUREMENT OF POWER CONSUMED BY A FLUORESCENT LAMP

Examine the power consumed by Fluorescent lamp using electrical devices using hardware and digital simulation.

Expt. 12: **DESIGN OF CHOKE AND SMALL TRANSFORMER**
Measure resistance and inductance of coil and construct the winding of transformer using winding machine using hardware and digital simulation.

Expt. 13: Z AND Y PARAMETERS
Determine the open circuit and short circuit parameters for two port network using hardware and digital simulation.

Expt. 14: **H AND ABCD PARAMETERS**
Determine the hybrid and transmission line parameters for two port network using hardware and digital simulation.

IV. REFERENCE BOOKS:

1. A Chakrabarti, "Circuit Theory", Dhanpat Rai Publications, 6th Edition, 2006.
2. William Hayt, Jack E Kemmerly S.M. Durbin, "Engineering Circuit Analysis", Tata McGraw Hill, 7th Edition, 2010.
3. K S Suresh Kumar, "Electric Circuit Analysis", Pearson Education, 1st Edition, 2013.

V. WEB REFERENCES:

1. https://www.ee.iitkgp.ac.in
2. https://www.citchennai.edu.in
3. https://www.iare.ac.in

PROGRAMMING FOR PROBLEM SOLVING USING C LABORATORY

II Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE /EEE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC05	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		0	0	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes:36**		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW

This course introduces students to writing computer programs. This course presents the principles of structured programming using the Python language, one of the most increasingly preferred languages for programming today. Because of its ease of use, it is ideal as a first programming language and runs on both the PC and Macintosh platforms. However, the knowledge gained in the course can be applied later to other languages such as C and Java. The course uses iPython Notebook to afford a more interactive experience.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Acquire logical thinking and identify efficient ways of solving problems using C programming language.
II. Develop programs by using decision making, branching and looping constructs.
III. Implement real time applications using the concept of array, pointers, functions and structures.

III. COURSE SYLLABUS:

Week – 1: OPERATORS AND EVALUATION OF EXPRESSIONS

e. Design and develop a flowchart and algorithm to read a number and implement using a C program to check whether the given number is even or odd using ternary operator.

f. Design and develop a flowchart and algorithm to read two integers and implement using a C program to perform the addition of two numbers without using + operator.

g. Develop a C program to evaluate the following arithmetic expressions by reading appropriate input from the standard input device. Understand the priority of operators while evaluating expressions.

   i. 6*2/(2+1 * 2/3 +6) +8 * (8/4)
   ii. 17 – 8 / 4 * 2 + 3 - ++2
   iii. ! (x > 10) && (y = = 2)

h. Develop a C program to display the size of various built-in data types in C language.

Week – 2: CONTROL STRUCTURES

a. Design and develop a flowchart and algorithm to read a year as an input and find whether it is leap year or not. Implement a C program for the same and execute for all possible inputs with appropriate messages. Also consider end of the centuries.

b. Design and develop a flowchart and algorithm to find the square root of a given number N. Implement a C program for the same and execute for all possible inputs with appropriate messages. (Note: Don't use library function sqrt(n), Hint: Use Newton-Raphson method to find the square root).

c. Design and develop a flowchart and algorithm to generate a Fibonacci sequence up to a given number N. A Fibonacci sequence is defined as follows: The first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Implement a C program for the developed flowchart/algorithm and execute the same to generate the first N terms of the sequence.

d. Design and develop a flowchart and algorithm that takes three coefficients (a, b, and c) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.

Week – 3: CONTROL STRUCTURES

a. Design and develop an algorithm to find the reverse of an integer number N and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: N: 2020, Reverse: 0202, Not a Palindrome.

b. Draw the flowchart and write C Program to compute sin(x) using Taylor series approximation given by

$$\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \ldots\ldots$$

Compare the result with the built- in Library function and print both the results with appropriate messages.

c. Design and develop an algorithm and flowchart to read a three digit number and check whether the given number is Armstrong number or not. Write a C program to implement the same and also display the Armstrong numbers between the ranges 1 to 1000.

d. Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0$, for a given value of x and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and x.

Week – 4: ARRAYS

e. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to print the second largest integer in a list of integers.

f. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to count and display positive, negative, odd and even numbers in an array.

g. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular number in a list of integers.

h. Develop, implement and execute a C program that reads two matrices A (m x n) and B (p x q) and Compute the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

Week – 5: STRINGS

a. Develop a user-defined function **STRCOPY (str1, str2)** to simulate the built-in library function **strcpy (str1, str2)** that copies a string str2 to another string str1. Write a C program that invokes this function to perform string copying. Also perform the same operation using built-in function.

b. Develop a user-defined function **STRCONCT (str1, str2)** to simulate the built-in library function **strcat (str1, str2)** that takes two arguments str1 and str2, concatenates str2 and str1 and stores the result in str1. Write a C program that invokes this function to perform string concatenation. Also perform the same operation using built-in function.

c. Develop a C program that returns a pointer to the first occurrence of the string in a given string using built-in library function **strstr()**. Example: **strstr()** function is used to locate first occurrence of the string "test" in the string "This is a test string for testing". Pointer is returned at first occurrence of the string "test".

d. Develop a C program using the library function **strcmp (str1, str2)** that compares the string pointed to by str1 to the string pointed to by str2 and returns an integer. Display appropriate messages based on the return values of this function as follows −

if return value < 0 then it indicates str1 is less than str2.
if return value > 0 then it indicates str2 is less than str1.
if return value = 0 then it indicates str1 is equal to str2.

Week – 6: FUNCTIONS

a. Design and develop a recursive and non-recursive function **FACT(num)** to find the factorial of a number, n!, defined by FACT(n) = 1, if n = 0. Otherwise FACT(n) = n * FACT(n-1). Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages

b. Design and develop a recursive function **GCD (num1, num2**) that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.

c. Design and develop a recursive function **FIBO (num)** that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to num.

d. Design and develop a C function **ISPRIME** (num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.

e. Design and develop a function **REVERSE (str**) that accepts a string arguments. Write a C program that invokes this function to find the reverse of a given string.

Week – 7: POINTERS

a. Develop a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.
b. Develop a C program to read a list of integers and store it in an array. Then read the array elements using a pointer and print the value along with the memory addresses.
c. Design and develop non-recursive functions **input_matrix(matrix, rows, col**s) and **print_matrix(matrix, rows, cols)** that stores integers into a two-dimensional array and displays the integers in matrix form. Write a C program to input and print elements of a two dimensional array using pointers and functions.
d. Develop a C program to a store a list of integers in a single dimensional array using dynamic memory allocation (limit will be at run time) using malloc() function. Write a C program to read the elements and print the sum of all elements along with the entered elements. Also use free() function to release the memory.

Week – 8: STRUCTURES AND UNIONS

e. Write a C program that uses functions to perform the following operations:
   i. Reading a complex number
   ii. Writing a complex number
   iii. Addition and subtraction of two complex numbers

   Note: represent complex number using a structure.
f. Write a C program to compute the monthly pay of 100 employees using each employee_s name, basic pay. The DA is computed as 52% of the basic pay. Gross-salary (basic pay + DA). Print the employees name and gross salary.
g. Create a Book structure containing book_id, title, author name and price. Write a C program to pass a structure as a function argument and print the book details.
h. Create a union containing 6 strings: name, home_address, hostel_address, city, state and zip. Write a C program to display your present address.

Week – 9: ADDITIONAL PROGRAMS

a. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: $1+x+x^2+x^3+\ldots$ .... $+x^n$. For example: if n is 3 and x is 5, then the program computes 1+5+25+125. Print x, n, the sum. Perform error checking. For example, the formula does not make sense for negative exponents – if n is less than 0. Have your program print an error message if n<0, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.
b. Develop a C program to find the 2's complement of a given binary number. 2's complement is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2's complement of 11100 is 00100. Write a C program to find the 2's complement of a binary number.
c. Develop a C program to convert a Roman numeral to its decimal equivalent. E.g. check for the inputs - Roman number IX is equivalent to 9 and Roman number XI is equivalent to 11.

Week – 10: PREPROCESSOR DIRECTIVES

a. Define a macro with one parameter to compute the volume of a sphere. Write a C program using this macro to compute the volume for spheres of radius 5, 10 and 15meters.
b. Define a macro that receives an array and the number of elements in the array as arguments. Write a C program for using this macro to print the elements of the array.
c. Write symbolic constants for the binary arithmetic operators +, -, *, and /. Write a C program to illustrate the use of these symbolic constants.

Week – 11: FILES

a. Create an employee file **employee.txt** and write 5 records having employee name, designation, salary, branch and city. Develop a C program to display the contents of **employee.txt** file.
b. Create a **studentolddata.txt** file containing student name, roll no, branch, section, address. Develop a C program to copy the contents of **studentolddata.txt** file to another file **studentnewdata.txt**.
c. Develop a C program to create a text file **info.txt** to store the information given below. Implement using a C program to count the number of words and characters in the file **info.txt**.

   Test Data:

   Input the file name to be opened : info.txt

Expected Output:

```
The content of the file info.txt are :
Welcome to IARE
Welcome to Computer Programming

The number of words in the  file info.txt are : 7
The number of characters in the  file info.txt are : 46
```

d. Given two university information files “**studentname.txt**” and “**roll_number.txt**” that contains students Name and Roll numbers respectively. Write a C program to create a new file called “**output.txt**” and copy the content of files “**studentname.txt**” and “**roll_number.txt**” into output file. Display the contents of output file “**output.txt**” on to the screen.

studname.txt	**roll_number.txt**
Asha	20951A1201
Bharath	20951A0502
Uma	20951A0456
Shilpa	20951A0305

Week – 12: **COMMAND LINE ARGUMENTS**

a. Develop a C program to read a set of arguments and display all arguments given through command line.
b. Develop a C program to read a file at command line argument and display the contents of the file.
c. Develop a C program to read N integers and find the sum of N integer numbers using command line arguments.
d. Develop a C program to read three integers and find the largest integer among three using command line argument.

IV. REFERENCE BOOKS:

1. Yashavant Kanetkar, “Let Us C”, BPB Publications, New Delhi, 13th Edition, 2012.
2. Oualline Steve, “Practical C Programming”, O’Reilly Media, 3rd Edition, 1997.
3. King KN, “C Programming: A Modern Approach”, Atlantic Publishers, 2nd Edition, 2015.
4. Kochan Stephen G, “Programming in C: A Complete Introduction to the C Programming Language”, Sam’s Publishers, 3rd Edition, 2004.
5. Linden Peter V, “Expert C Programming: Deep C Secrets”, Pearson India, 1st Edition, 1994.

V. WEB REFERENCES:

1. http://www.sanfoundry.com/c-programming-examples
2. http://www.geeksforgeeks.org/c
3. http://www.cprogramming.com/tutorial/c
4. http://www.cs.princeton.edu

ENGINEERING WORKSHOP PRACTICE

I Semester: **CSE / CSE (AI&ML) / CSE (DS) / CSE (CS) / CSIT / IT**

II Semester: **ECE / EEE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AMEC04	Foundation	L	T	P	C	CIA	SEE	Total
		-	-	2	1	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 28			Total Classes: 28			

Prerequisite: **There are no prerequisites to take this course.**

I. COURSE OVERVIEW:

The course is intended to provide the basic concepts about Engineering tools for cutting and measuring used in a workshop. The students will be benefited from hands on training process as well as knowledge to carry out a particular process for making a product. This course provides wider perspective of manufacturing, processes to learn and introduces major trades as well as digital manufacturing facilities.

II.COURSE OBJECTIVES:

The students will try to learn:

I. The application of jigs and fixtures, measuring, marking and cutting tools in various types of manufacturing processes.

II. The preparation of different joints in carpentry and fitting and also familiarizes wood working machinery.

III. The concepts of forming processes by forging, black-smithy and tin-smithy with an application extracts of Engineering Drawing.

IV. The standard electrical wiring practices for domestic and industrial appliances.

V. The current advancements in developing the prototype models through digital manufacturing facilities.

III. COURSE SYLLABUS:

Week-1: CARPENTRY-I

Batch I: Preparation of Tenon joint as per given dimensions.

Batch II: Preparation of Mortise joint as per given taperangle.

Week -2: CARPENTRY-II

Batch I: Preparation of dove tail joint as per given taper angle.

Batch II: Preparation of lap joint as per given dimensions.

Week-3: FITTING - I

Batch I: Make a straight fit for given dimensions.

Batch II: Make a square fit for given dimensions.

Week-4: FITTING - II

Batch I : Make a V fit for given dimensions

Batch II: Make a semicircular fit for given dimensions.

Week-5: BLACKSMITHY- I

Batch I: Prepare S-bend for given MS rod using open hearth furnace.

Batch II: Prepare J-bend for given MS rod using open hearth furnace.

Week-6: BLACKSMITHY- II

Batch I: Prepare Fan hook for given dimensions.

Batch II: Prepare Round to Square for given dimensions

Week-7: MOULD PREPARATION

Batch I: Prepare a wheel flange mould using a given wooden pattern.
Batch II: Prepare a bearing housing using an aluminum pattern.

Week-8: MOULD PREPARATION
Batch I: Prepare a bearing housing using an aluminum pattern.
Batch II: Prepare a wheel flange mould using a given wooden pattern.

Week-9: TINSMITHY- I
Batch I: Prepare the development of a surface and make a rectangular tray for given dimensions.
Batch II: Prepare the development of a surface and make a round tin for given dimensions.

Week-10: TINSMITHY- II
Batch I: Prepare the development of a surface and make a Square Tin, for given dimensions.
Batch II: Prepare the development of a surface and make a Conical Funnel for given dimensions.

Week-11: ELECTRICAL WIRING-I
Batch I: Make an electrical connection of two bulbs connected in series.
Batch II:Make an electrical connection of two bulbs connected in parallel

Week-12: ELECTRICAL WIRING-II
Batch I: Make an electrical connection of one bulb controlled by two switches connected.
Batch II: Make an electrical connection of tube light.

IV. REFERENCE BOOKS:

1. Hajra Choudhury S.K., Hajra Choudhury A.K. and Nirjhar Roy S.K., “Elements of Workshop Technology”, Media promoters and publishers private limited, Mumbai, Vol. I 2008 and Vol. II 2010.
2. Kalpakjian S, Steven S. Schmid, “Manufacturing Engineering and Technology”, Pearson Education India Edition, 4th Edition, 2002.
3. Gowri P. Hariharan, A. Suresh Babu, “Manufacturing Technology – I”, Pearson Education, 2008.
4. Roy A. Lindberg, “Processes and Materials of Manufacture”, Prentice Hall India, 4th Edition, 1998.
5. Rao P.N., “Manufacturing Technology”, Vol. I and Vol. II, Tata McGraw-Hill House, 2017.

V. WEB REFERENCES:

http://www.iare.ac.in

ELECTRONIC DEVICES AND CIRCUITS

III Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC01	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	1	0	4	30	70	100
Contact Classes: 45	**Tutorial Classes: 15**	**Practical Classes: Nil**				**Total Classes: 60**		

Prerequisites: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

This course provides the basic knowledge over the construction and functionality of the basic electronic devices such as diodes and transistors. It also provides the information about the electronic switches and the flow of current through these switches in different biasing conditions. This course is intended to describe the different configurations to provide temperature stability and how these electronic devices can be configured to work as rectifiers, clippers, voltage regulators, clampers and amplifiers.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The operational principles, characteristics of semiconductor devices and circuits.

II. The principles of operating semiconductor devices for rectification, amplification, conditioning and voltage regularization of signals.

III. The analytical skills needed to model analog and digital integrated circuits (IC) at discrete and micro circuit level.

IV. The foundations of basic electronic circuits necessary for building complex electronic hardware.

III. COURSE SYLLABUS:

MODULE – I: DIODE AND APPLICATIONS (12)

Diode - Static and Dynamic resistances, Equivalent circuit, Load line analysis, Diffusion and Transition Capacitances, Diode Applications: Switch-Switching times. Rectifier - Half Wave Rectifier, Full Wave Rectifier, Bridge Rectifier, Rectifiers with Capacitive Filter, Clippers-Clipping at two independent levels, Clampers-Clamping Operation, types, Clamping Circuit Theorem, Comparators.

MODULE – II: BIPOLAR JUNCTION TRANSISTOR (BJT) (12)

Principle of Operation and characteristics - Common Emitter, Common Base, Common Collector Configurations, Operating point, DC & AC load lines, Transistor Hybrid parameter model, Determination of h-parameters from transistor characteristics, Conversion of h-parameters.

MODULE – III: TRANSISTOR BIASING AND STABILIZATION (12)

Bias Stability, Fixed Bias, Collector to Base bias, Self-Bias, Bias Compensation using Diodes and Transistors.

Analysis and Design of Small Signal Low Frequency BJT Amplifiers: Analysis of CE, CC, CB Amplifiers and CE Amplifier with emitter resistance, low frequency response of BJT Amplifiers, effect of coupling and bypass capacitors on CE Amplifier.

MODULE – IV: JUNCTION FIELD EFFECT TRANSISTOR (12)

Construction, Principle of Operation, Pinch-Off Voltage, Volt- Ampere Characteristic, Comparison of BJT and FET, Biasing of FET, FET as Voltage Variable Resistor, MOSFET Construction and its Characteristics in Enhancement and Depletion modes.

MODULE – V: FET AMPLIFIERS (12)

Small Signal Model, Analysis of CS, CD, CG JFET Amplifiers. Basic Concepts of MOSFET Amplifiers. **Special Purpose Devices:** Zener Diode - Characteristics, Voltage Regulator; Principle of Operation - SCR, Tunnel diode, UJT, Varactor Diode.

IV. TEXT BOOKS:

1. Jacob Millman, “Electronic Devices and Circuits”, McGraw Hill Education, 3rd Edition, 2014.
2. Robert L. Boylestead, Louis Nashelsky, “Electronic Devices and Circuits Theory, Pearson, 11th Edition, 2009.

V. REFERENCE BOOKS:

1. Horowitz, “The Art of Electrionics, Cambridge University Press, 3rd Edition, 2018.
2. David A. Bell, “Electronic Devices and Circuits”, Oxford, 5th Edition, 2016.
3. J. Millman, H. Taub and Mothiki S. Prakash Rao, “Pulse, Digital and Switching Waveforms”, McGraw Hill, 2nd Edition, 2008.
4. S. Salivahanan, N.Suresh Kumar, A Vallvaraj, “Electronic Devices and Circuits, TMH, 2nd Edition, 2017.

VI. WEB REFERENCES:

1. http://www-mdp.eng.cam.ac.uk/web/library/enginfo/electrical/hong1.pdf
2. https://archive.org/details/ElectronicDevicesCircuits
3. http://nptel.ac.in/courses/Webcourse-contents/IIT-ROORKEE/BASIC ELECTRONICS/home_page.htm
4. http://www.vidyarthiplus.in/2011/11/electronic-device-and-circuits-edc.html
5. http://www.satishkashyap.com/2013/03/video-lectures-on-electron-devices-by.html

SIGNALS AND SYSTEMS

III Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC02	Core	L	T	P	C	CIA	SEE	Total
		3	1	0	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisites: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

This course integrates the basic concepts of both continuous and discrete time signals and systems. It covers the linear time invariant systems and their analysis in time and frequency domain, mathematical tools, correlation and convolution of signals, sampling techniques. It provides the necessary background needed for understanding the signal processing and communications.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The representation, classification and analysis of continuous, discrete time signals in time and frequency domains.

II. The Fourier transform, Laplace and Z- transforms and their properties to analyze the signals and systems.

III. The temporal and spectral characteristics of Random process and the extraction of Signal from Noise by filtering.

IV. The sampling, quantization and reconstruction requirements for digital signal processing applications

III. COURSE SYLLABUS:

MODULE – I: SIGNAL ANALYSIS (12)

Analogy between Vectors and Signals, Orthogonal Signal Space, Signal approximation using Orthogonal functions, Mean Square Error, Closed or complete set of Orthogonal functions, Orthogonality in Complex functions, Classification of Signals and systems, Exponential and Sinusoidal signals, Concepts of Impulse.
function, Unit Step function, Signum function.

MODULE – II: FOURIER SERIES (12)

Representation of Fourier series, Continuous time periodic signals, Properties of Fourier Series, Dirichlet"s conditions, Trigonometric Fourier Series and Exponential Fourier Series, Complex Fourier spectrum.

Fourier Transforms:

Deriving Fourier Transform from Fourier series, Fourier Transform of arbitrary signal, Fourier Transform of standard signals, Fourier Transform of Periodic Signals, Properties of Fourier Transform, Fourier Transforms involving Impulse function and Signum function.

MODULE – III: SIGNAL TRANSMISSION THROUGH LINEAR SYSTEMS (12)

Linear System, Impulse response, Response of a Linear System, Linear Time Invariant(LTI) System, Linear Time Variant (LTV) System, Transfer function of a LTI System, Filter characteristic of Linear System, Distortion less transmission through a system, Signal bandwidth, System Bandwidth, Ideal LPF, HPF, and BPF characteristics

Causality and Paley-Wiener criterion for physical realization, Relationship between Bandwidth and rise time, Convolution and Correlation of Signals, Concept of convolution in Time domain and Frequency domain, Graphical representation of Convolution.

MODULE - IV LAPLACE TRANSFORM AND Z-TRANSFORM (12)

Laplace Transforms (L.T), Inverse Laplace Transform, Concept of Region of Convergence (ROC) for Laplace Transforms, Properties of L.T, Relation between L.T and F.T of a signal, Laplace Transform of certain signals using waveform synthesis. Z–Transforms: Concept of Z- Transform of a Discrete Sequence, Distinction between Laplace, Fourier and Z Transforms, Region of Convergence in Z-Transform, Constraints on ROC for various classes of signals, Inverse Z-transform, Properties of Z-transforms.

MODULE - V SAMPLING THEOREM (12)

Graphical and analytical proof for Band Limited Signals, Impulse Sampling, Natural and Flat top Sampling, Reconstruction of signal from its samples, Effect of under sampling – Aliasing, Introduction to Band Pass Sampling. Correlation: Cross Correlation and Auto Correlation of Functions, Properties of Correlation Functions, Energy Density Spectrum, Parsevals Theorem, Power Density Spectrum, Relation between Autocorrelation Function and Energy/Power Spectral Density Function, Relation between Convolution and Correlation

IV. TEXT BOOKS:

3. B.P. Lathi, “Signals, Systems & Communications”, BSP, 2013.
4. A.V. Oppenheim, A.S. Willsky and S.H. Nawabi, “Signals and Systems”, 2nd Edition 2010.

V. REFERENCE BOOKS:

1. Simon Haykin and Van Veen, “Signals and Systems”, Wiley Publications, 2nd Edition, 2010.
2. Michel J. Robert, “Fundamentals of Signals and Systems”, MGH International Edition. 2nd Edition, 2008.

VI. WEB REFERENCES:

1. https://www.edx.org/course/discrete-time-signal-processing-mitx-6-341x-1
2. https://www.mooc-list.com/course/digital-signal-processing-coursera

DIGITAL SYSTEM DESIGN

III Semester: **ECE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC03	Core	3	0	0	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

The course will make them learn the basic theory of switching circuits and their applications in detail. Starting from a problem statement they will learn to design circuits of logic gates that have a specified relationship between signals at the input and output terminals. They will be able to design combinational and sequential circuits. They will learn to design counters, adders, sequence detectors. This course provides a platform for advanced courses like Computer architecture, Microprocessors & Microcontrollers and VLSI design. Greater Emphasis is placed on the use of programmable logic devices and State machines.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Simplification of the logic functions using Boolean algebraic theorems and techniques.
II. Implementation of conventional combinational and sequential circuits including conversions of flip-flops
III. The exploration of the logic families and semiconductor memories.
IV. The realization of the micro and macro circuits using VHDL programming.

III. SYLLABUS:

MODULE – I: LOGIC SIMPLIFICATION AND COMBINATIONAL LOGIC DESIGN (08)

Review of Boolean Algebra and De Morgan’s Theorem, SOP & POS forms, Canonical forms, Karnaugh maps up to 6 variables, Binary codes, Code Conversion

MODULE – II: MSI DEVICES (10)

MSI devices like Comparators, Multiplexers, Encoder, Decoder, Driver & Multiplexed Display, Half and Full Adders, Subtractors, Serial and Parallel Adders, BCD Adder, Barrel shifter and ALU

MODULE – III: SEQUENTIAL LOGIC DESIGN (10)

Building blocks like S-R, JK and Master-Slave JK FF, Edge triggered FF, Ripple and Synchronous counters, Shift registers.

Finite state machines, Design of synchronous FSM, Algorithmic State Machines charts. Designing synchronous circuits like Pulse train generator, Pseudo Random Binary Sequence generator, Clock generation

MODULE - IV LOGIC FAMILIES AND SEMICONDUCTOR MEMORIES (08)

TTL NAND gate, Specifications, Noise margin, Propagation delay, fan-in, fan-out, Tristate TTL, ECL, CMOS families and their interfacing, Memory elements, Concept of Programmable logic devices like FPGA. Logic implementation using Programmable Devices.

MODULE - V VLSI DESIGN FLOW (09)

Design entry: Schematic, FSM & HDL, different modeling styles in VHDL, Data types and objects, Dataflow, Behavioral and Structural Modeling, Synthesis and Simulation VHDL constructs and codes for combinational and sequential circuits.

IV. TEXT BOOKS:

1. R.P. Jain, “Modern digital Electronics”, Tata McGraw Hill, 4th Edition, 2009.
2. Douglas Perry, “VHDL”, Tata McGraw Hill, 4th Edition, 2002.
3. W.H. Gothmann, “Digital Electronics- An introduction to theory and practice”, PHI, 2nd Edition, 2006.

V. REFERENCE BOOKS:

1. D.V. Hall, “Digital Circuits and Systems”, Tata McGraw Hill, 1989.
2. Charles Roth, “Digital System Design using VHDL”, Tata McGraw Hill, 2nd Edition, 2012.

.

VI. WEB REFERENCES:

1. mcsbzu.blogspot.com
2. http://books.askvenkat.com
3. http://worldclassprogramme.com
4. http://www.daenotes.com
5. http://nptel.ac.in/courses/117106086/1

PROBABILITY THEORY AND STOCHASTIC PROCESSES

III Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC04	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 60**		
Prerequisites: **There are no prerequisites to take this course.**								

I. COURSE OVERVIEW:

Stochastic or random processes are mathematical objects defined on probability space. The study of these processes is of primary importance in all science and engineering specializations. To electronics and communications, the stochastic processes are useful in the analysis and the design of communication systems. The study involves the theory of probability and the theory of signals and systems. This course comprises two parts. The first part introduces the fundamental principles of probability theory and random variables necessary to understand the stochastic processes. The second part introduces the basic concepts of random processes, random signals, and their interaction with the electrical or electronic systems. The temporal and spectral properties of the stochastic processes at the input and the output of a linear time invariant (LTI) are discussed in detail. The course forms the basis for the next level courses of an electronics engineer such as Analog communication (AC), Digital communication (DC) and Digital Signal Processing (DSP), Radar Systems (RS) and Digital Image Processing (DIP). It is also useful for a data science engineer in designing the machine learning algorithms

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of the 1-dimensional and 2-dimensional random variables and their characterization in probability space.

II. The stationary random process, its framework and application for analysing random signals and noises.

III. The characteristics of 1-dimensional stationary random signals in time and frequency domains.

IV. Analysis of the response of a linear time invariant (LTI) system driven by 1- dimensional stationary random signals useful for subsequent design and analysis of communication systems.

III. SYLLABUS:

MODULE – I: PROBABILITY, RANDOM VARIABLES AND OPERATIONS ON RANDOM VARIABLES (09)

Random Experiments, Sample Spaces, Events, Probability, Axioms, Joint, Conditional and Total Probabilities, Bay‟s Theorem, Independent Events. Random Variables: Definition, Conditions for mapping function of a Random Variable, Types of Random Variable, Distribution and Density functions: Definition and Properties, Binomial, Poisson, Uniform, Gaussian, Exponential, Rayleigh, random variables, Methods of defining Conditioning Event, Conditional Distribution, Conditional Density and their Properties, Expected Value of a Random Variable, Function of a Random Variable, Standard and Central Moments, Variance and Skew, Chebychev‟s Inequality

MODULE – II: SINGLE RANDOM VARIABLE TRANSFORMATIONS - MULTIPLE RANDOM VARIABLES (09)

Characteristic Function, Moment Generating Function, Monotonic and Non-monotonic Transformations of Single Random Variables (Continuous and Discrete), Vector Random Variables, Joint Distribution Function and its Properties, Marginal Distribution Functions, Joint Density Function and its Properties, Marginal Density Functions, Conditional Distribution and Density – Point Conditioning, Conditional Distribution and Density – Interval conditioning, Statistical Independence, Sum of Two and more Random Variables, Central Limit Theorem: Equal and Unequal Distribution.

MODULE – III: OPERATIONS ON MULTIPLE RANDOM VARIABLES – EXPECTATIONS (09)

Expected value of a function of multiple random variables, Correlation and Covariance, Correlation Coefficient, Joint Moments about the origin, Joint Central moments, Joint characteristic function, Joint moment generating function.

Jointly Gaussian random variables: Two random variables case and N random variable case, Properties, Transformations of Multiple Random Variables, Jacobian Matrix, Linear Transformations of Gaussian Random Variables

MODULE – IV: RANDOM PROCESSES – TEMPORAL CHARACTERISTICS (09)

Random Process: Definition and Classification, Distribution and Density Functions, Stationarity and Statistical Independence., First- Order, Second- Order, Wide-Sense Stationarities (N-Order) and Strict-Sense Stationarity, Time Averages and Ergodicity, Mean-Ergodic and Correlation-Ergodic Processes, Autocorrelation Function and Its Properties, Cross-Correlation Function and Its Properties, Covariance Functions, Gaussian and Poisson Random Processes. Response of Linear Systems to Random Process input, Mean and MS value of System Response, Autocorrelation Function of Response, Cross- Correlation between Input and Output.

MODULE – V: RANDOM PROCESSES – SPECTRAL CHARACTERISTICS (09)

Power Density Spectrum: Definition and Properties, Relationship between Power Density Spectrum and Autocorrelation Function, Cross Power Spectral Density: Definition and Properties, Relationship between Cross Power Spectrum and Cross-Correlation Function, System Evaluation using Random Noise, Spectral Characteristics of System Response: Power Density Spectrum of Response, Cross-Power Density Spectra of Input and Output, Noise Bandwidth, White and Colored Noises

IV. TEXT BOOKS:

1. Peyton Z. Peebles, "Probability, Random Variables & Random Signal Principles", TMH, 4th Edition, 2001.

V. REFERENCE BOOKS:

1. Bruce Hajck, "Random Processes for Engineers", Cambridge Unipress, 2015.
2. Athanasios Papoulis and S. Unnikrishna Pillai, "Probability, Random Variables and Stochastic Processes", PHI, 4th Edition,2002.
3. K.Murugesan, P. Guruswamy, "Probability, Statistics & Random Processes", Anuradha Agencies, 3rd Edition, 2003.
4. B.P. Lathi, "Signals, Systems & Communications" B.S. Publications, 2003.

VI. WEB REFERENCES:

1. www.britannica.com/topic/probability-theory
2. www.math.uiuc.edu/~r-ash/BPT.html
3. https://www.ma.utexas.edu/users/gordanz/.../introduction_to_stochastic_processes.pdf
4. nptel.ac.in/courses/111102014/
5. http://vceece2k10.blogspot.in/p/semester-2-1.html

DATA STRUCTURES

III Semester: Common for all branches								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC08	Core	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Python Programming								

I. COURSE OVERVIEW:

The course covers some of the general-purpose data structures and algorithms, and software development. Topics covered include managing complexity, analysis, static data structures, dynamic data structures and hashing mechanisms. The main objective of the course is to teach the students how to select and design data structures and algorithms that are appropriate for problems that they might encounter in real life. This course reaches to student by power point presentations, lecture notes, and lab which involve the problem solving in mathematical and engineering areas.

II. COURSE OBJECTIVES:

The students will try to learn:

I. To provide students with skills needed to understand and analyze performance trade-offs of different algorithms / implementations and asymptotic analysis of their running time and memory usage.

II. To provide knowledge of basic abstract data types (ADT) and associated algorithms: stacks, queues, lists, tree, graphs, hashing and sorting, selection and searching.

III. The fundamentals of how to store, retrieve, and process data efficiently.

IV. To provide practice by specifying and implementing these data structures and algorithms in Python.

V. Understand essential for future programming and software engineering courses.

III. SYLLABUS:

MODULE – I: INTRODUCTION TO DATA STRUCTURES, SEARCHING AND SORTING (09)

Basic concepts: Introduction to data structures, classification of data structures, operations on data structures; Algorithm Specification, Recursive algorithms, Data Abstraction, Performance analysis- time complexity and space complexity, Asymptotic Notation-Big O, Omega, and Theta notations. Introduction to Linear and Non Linear data structures, Searching techniques: Linear and Binary search; Sorting techniques: Bubble, Selection, Insertion, Quick and Merge Sort and comparison of sorting algorithms.

MODULE – II: LINEAR DATA STRUCTURES (09)

Stacks: Stack ADT, definition and operations, Implementations of stacks using array, applications of stacks, Arithmetic expression conversion and evaluation; Queues: Primitive operations; Implementation of queues using Arrays, applications of linear queue, circular queue and double ended queue (deque).

MODULE – III: LINKED LISTS (09)

Linked lists: Introduction, singly linked list, representation of a linked list in memory, operations on a single linked list; Applications of linked lists: Polynomial representation and sparse matrix manipulation.

Types of linked lists: Circular linked lists, doubly linked lists; Linked list representation and operations of Stack, linked list representation and operations of queue.

MODULE - IV NON LINEAR DATA STRUCTURES (09)

Trees: Basic concept, binary tree, binary tree representation, array and linked representations, binary tree traversal, binary tree variants, threaded binary trees, application of trees, Graphs: Basic concept, graph terminology, Graph Representations - Adjacency matrix, Adjacency lists, graph implementation, Graph traversals – BFS, DFS, Application of graphs, Minimum spanning trees – Prims and Kruskal algorithms.

MODUE - V BINARY TREES AND HASHING (09)

Binary search trees: Binary search trees, properties and operations; Balanced search trees: AVL trees; Introduction to M-

Way search trees, B trees; Hashing and collision: Introduction, hash tables, hash functions, collisions, applications of hashing.

IV. TEXT BOOKS:

1. Rance D. Necaise, “Data Structures and Algorithms using Python”, Wiley Student Edition.
2. Benjamin Baka, David Julian, “Python Data Structures and Algorithms”, Packt Publishers, 2017.

V. REFERENCE BOOKS:

1. S. Lipschutz, “Data Structures”, Tata McGraw Hill Education, 1st Edition, 2008.
2. D. Samanta, “Classic Data Structures”, PHI Learning, 2nd Edition, 2004.

VI. WEB REFERENCES:

1. https://www.tutorialspoint.com/data_structures_algorithms/algorithms_basics.htm
2. https://www.codechef.com/certification/data-structures-and-algorithms/prepare
3. https://www.cs.auckland.ac.nz/software/AlgAnim/dsToC.html
4. https://online-learning.harvard.edu/course/data-structures-and-algorithms

EXPERIENTIAL ENGINEERING EDUCATION (EXEED) – PROTOTYPE / DESIGN BUILDING

III Semester: CSE / CSE (AI&ML) / CSE (DS) / CSE(CS) / IT / CSIT / AE / ME / CE / ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC09	Foundation	L	T	P	C	CIA	SEE	Total
		2	0	0	1	30	70	100
Contact Classes: 28	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 28		

Prerequisite: There are no prerequisites to take this course

I. COURSE OVERVIEW:

This course provides an overall exposure to the various methods and tools of prototyping. This course discusses Low-Fidelity, paper, wireframing and tool based prototyping techniques along with design principles and patterns.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic principles and design aspect of prototyping.
II. The various techniques, design guidelines and patterns.
III. The applications of prototyping using various tools and platforms.

WEEK NO	TOPIC
WEEK – I	An introduction to Prototyping
WEEK – II	Low - Fidelity Prototyping and Paper Prototyping
WEEK – III	Wireframing and Tool based Prototyping
WEEK – IV	Physical Low- Fidelity Prototyping
WEEK – V	Tool based prototyping
WEEK – VI	Design Principles and Patterns- Graphic Design
WEEK – VII	Design Principles and Patterns- Interaction Design
WEEK –VIII	Commercial design guidelines and standards.
WEEK - IX	Universal design: Sensory and cognitive impairments
WEEK - X	Universal design: Tools, Limitations and standards
WEEK - XI	Introduction platforms and context : Mobile UI design, Wearable
WEEK - XII	Introduction platforms and context : Automotive user interface
WEEK - XIII	Introduction platforms and context : IoT and Physical Computing
WEEK - XIV	Assessment

ELECTRONIC DEVICES AND CIRCUITS LABORATORY

III Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AECC05	**Core**	0	0	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes:36**		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This course provides the hands-on experience on designing circuits using Diodes, Bipolar Junction Transistors, Field Effect Transistors, UJTs and SCRs. Determine the gain, bandwidth and input output impedances of BJT and FET amplifiers. Provides the capability to extract the characteristics of semiconductor devices with simulation tools.

II. COURSES OBJECTIVES:

The students will try to learn

I. The behavior and characteristics of semiconductor devices for designing the semiconductor circuits such as amplifier and rectifiers.

II. Estimation of device characteristics like gain, bandwidth, input and output resistance of bipolar junction transistors and field effect transistors amplifiers to derive appropriate small-signal model analysis of basic amplifier circuits.

III. The analytical skills to model analog and digital integrated circuits at discrete and micro circuit level.

III. COURSE OUTCOMES:

At the end of the course students should be able to:

CO1 Demonstrate the electronic instruments for measuring voltage, current and phase parameters.

CO2 Experiment and determine the parameters of rectifiers and voltage regulators using the diode characteristics.

CO3 Examine the input and output characteristics of transistor (BJT and FET) configurations for determining input - output resistances.

CO4 Characterize BJT and FET amplifiers for estimating the voltage gain and Current gain.

CO5 Demonstrate the intrinsic stand-off ratio of the uni-junction transistor using volt ampere characteristics

CO6 Build and determine holding, latching current and break over voltage of silicon-controlled rectifier using volt - ampere characteristics.

DO’s

1. Once the operation is completed pull the plug itself rather chord attached to it.
2. Repair the equipment switch-off the supply and go on.
3. Operate the equipment on supply; see that hands are dry, if that is not possible, hide the hand in the pockets.
4. If a person comes in contact with current unexpectedly don’t touch the person with hands but immediately use any insulator material and shut down the power (like leather belts, wood and plastic bars etc).
5. If water is nozzles on the equipment, immediately shutdown the power using circuit breaker or pull out the plug.
6. Use the connecting wires of good continuity, short circuit of connecting wire leads damage of circuit parameters.

DON’Ts

1. Do not wear loose clothing and do not hold any conducting materials in contact with skin when the power is on.
2. Do not pull out the connections until unless all the currents are dead.
3. Do not overload the circuit by plugging in too many appliances.
4. If you are mentally and physically stressed don’t operate the power equipment.
5. Never operate the equipment under wet conditions.
6. Do not interconnect two or more wires, take appropriate length of wire.

SAFETY NORMS

1. The lab must be equipped with fire extinguisher.
2. See that the connections are made tight.
3. Use single plug for each equipment.
4. Cover the body completely to avoid arc effect.
5. To change the connections during the experiment, switch off the supply and carry on.
6. Used equipment may get heated, so take care handling the equipment after it is used.
7. Do the wiring, all setups and check the circuit connections before the supply is on

EXERCISES FOR ELECTRONIC DEVICES AND CIRCUITS LABORATORY

Note: Students are encouraged to bring their own laptops for laboratory practice session

1. Getting Started Exercises

1.1 Introduction to MULTISIM

Step 1: Open Multisim.

Step 2: Place components.

Step 3: Wire components.

Step 4: Place a simulation source.

Step 5: Place measurement instruments.

Step 6: Run a simulation

1.2 *V-I* characteristics of p-n junction diode

A p-n junction diode conducts only in one direction. The *V-I* characteristic of the p-n diode is a curve plot between voltage measured across the diode and current flowing through the diode. When external voltage is zero, circuit is open and the potential barrier does not allow the current to flow.

1.2.1 *V-I* characteristics of p-n junction diode under forward bias

When p-type (Anode) is connected to +ve terminal and n-type (cathode) is connected to –ve terminal of the supply voltage is known as forward bias.

Draw the *V-I* characteristics curve of p-n junction diode under forward bias as shown in Figure 1.2.1.

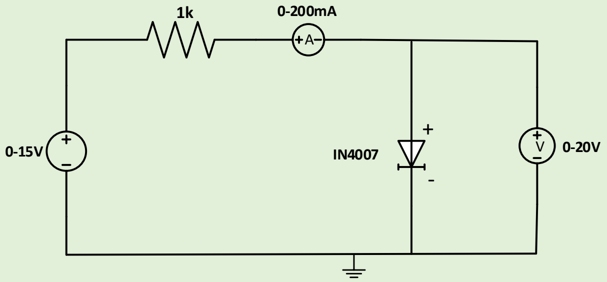


Figure 1.2.1: p-n junction diode forward bias

1.2.2 *V-I* characteristics of p-n junction diode under reverse bias

When n-type (cathode) is connected to +ve terminal and p-type (Anode) is connected – ve terminal of the supply voltage is known as reverse bias and the potential barrier across the junction increases.

Draw the *V-I* characteristics curve of p-n junction diode under reverse bias as in Figure 1.2.2.

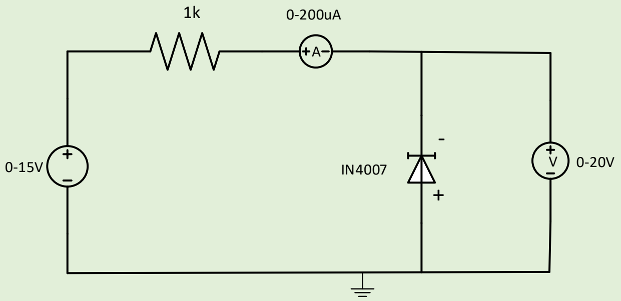


Figure 1.2.2: p-n junction diodes reverse bias

Try

1. The reverse saturation current of a silicon p-n junction diode is 10 mA. Calculate the diode current for the forward bias voltage of 0.6 V at 25 °C.
2. A diode connected to an external resistance and an e.m.f assuming that the barrier potential developed in diode is 0.5V. Obtain the value of current shown in Figure 1.2.3 in milli-ampere.

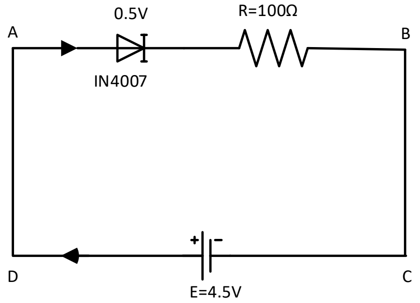


Figure 1.2.3

3. The diode used in the Figure 1.2.4 has a constant voltage drop of 0.5 V at all current and a maximum power rating of 100mill watts. What should be the value of the resistor R, connected in series with the diode for obtaining maximum current?

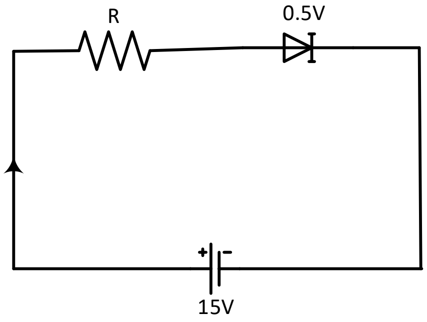


Figure 1.2.4

2. Exercises on Zener Diode Characteristics and Voltage Regulator

A Zener diode is heavily doped p-n junction diode, specially made to operate in the break down region. A p-n junction diode normally does not conduct when reverse biased. But if the reverse bias is increased, at a particular voltage it starts conducting heavily. This voltage is called Break down Voltage. High current through the diode can permanently damage the device. To avoid high current, we connect a resistor in series with Zener diode.

2.1 Zener diode under forward bias

The forward bias characteristic of Zener diode is same as the normal p-n junction diode.

Examine the Zener diode characteristics under forward bias from the Figure 2.1.

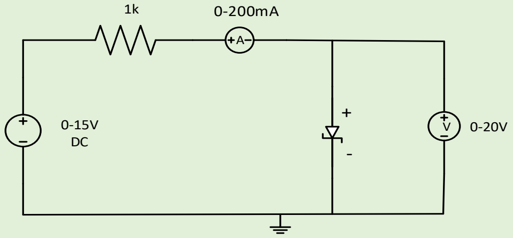


Figure 2.1: Zener diode forward bias

2.2 Zener diode under reverse bias

Examine the Zener diode characteristics under reverse bias from the Figure 2.2.

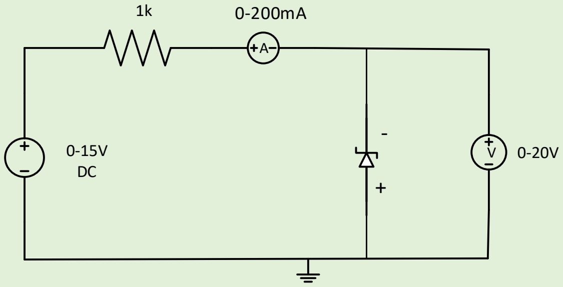


Figure 2.2: Zener diode reverse bias

2.3 Zener diode as voltage regulator

Examine the Zener diode as a Voltage Regulator from the Figure 2.3.

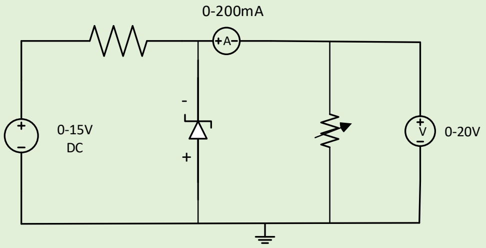


Figure 2.3: Zener diode as voltage regulator

Try

1. Design a zener voltage regulator circuit to drive a load of 6V, 100 mW from an unregulated input supply of Vmin = 8V, Vmax = 12V using a 6V zener diode.
2. Find the maximum zener current for the zener diode which is connected in voltage regulator to protect the load. Given V_Z=6V, R_Z=1.5Ω, R=400Ω.
3. The potential of the battery is varied from 10V to 16V. If by zener diode breakdown voltage is 6V, find maximum current through zener diode.

3. Exercises on Half wave rectifier with and without filter

Rectifier is an electronic circuit, which offers low resistance in one direction and high resistance in opposite direction. Rectifiers are used to convert AC voltages and DC voltages and currents.

3.1 Half wave rectifier without filter

Design a half-wave rectifier circuit and analyze the rectifier output without a filter in Figure 3.1

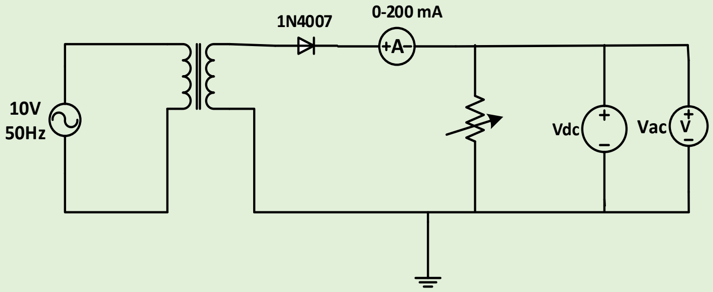


Figure 3.1: Half wave rectifier without filter

3.2 Half wave rectifier with filter

Design a half-wave rectifier circuit and analyze the rectifier output with filter as in Figure 3.2.

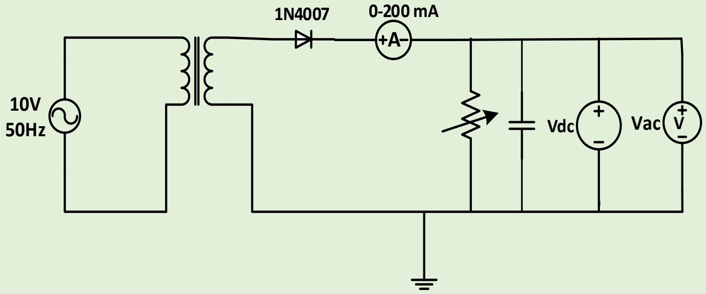


Figure 3.2: Half wave rectifier with filter

Try

1. An ac supply of 230 V is applied to a half-wave rectifier circuit through transformer of turn's ratio 5:1. Assume the diode is an ideal one. The load resistance is 300V. Find (a)d.c. output voltage (b) PIV (c) maximum (d) average values of power delivered to the load using multisim.
2. Design half wave rectifier with an a.c. supply of 230V is applied through a transformer of turn ratio 10:1. Observe the output d.c. voltage, peak inverse voltage and identify d.c. output voltage if transformer turns ratio changed to 20:1.
3. An AC supply of 230 V is applied to a half-wave rectifier circuit through a transformer of turn ratio 10:1. Find (i) the output d.c voltage and (ii) the peak inverse voltage. Assume the diode to be ideal.
4. A half-wave rectifier is used to supply 50V d.c. to a resistive load of 800 Ω. The diode has a resistance of 25 Ω. Calculate a.c. voltage required.
5. A half wave rectifier has a load of 3.5k. If the diode resistance and secondary coil resistance together have a resistance of 800 and the input voltage has a signal voltage of peak value 240 V, calculate (i) Peak, average and rms value of current flowing (ii) d.c. power output (iii) a.c. power input (iv) Efficiency of the rectifier.

4. Exercises on Full wave rectifier with and without filter

The full-wave rectifier consists of a center-tapped transformer, which results in equal voltages above and below the center-tap. During the positive half cycle, a positive voltage appears at the anode of D1 while a negative voltage appears at the anode of D2. Due to this diode D1 is forward biased. It results a current I_{d1} through the load R.

4.1 Full wave rectifier without filter

Design a full-wave rectifier circuit and analyze the rectifier output without filter in Figure 4.1.

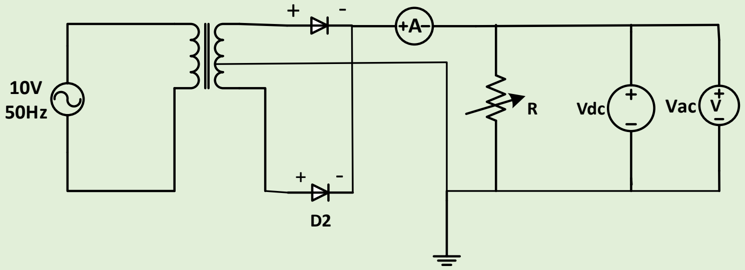


Figure 4.1: Full wave rectifier without filter

4.2 Full wave rectifier with filter

Design a full-wave rectifier circuit and analyze the rectifier output with filter in Figure 4.2.

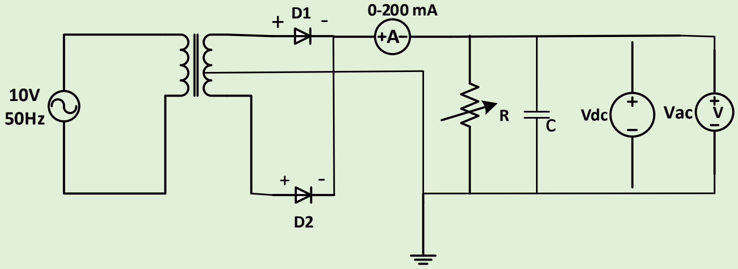


Figure 4.2: Full wave rectifier with filter

Try

1. Design a bridge rectifier using four identical diodes having forward resistance of 5 V and the secondary voltage is 30 V_{rms}. Determine the dc output voltage for I_{dc} 200 mA and value of the output ripple voltage.
2. Design a $12V_{dc}$. power supply out of an a.c. line source (120 V_{ac}), a transformer with N_p / N_s = 10, some diodes, and a low pass filter. Calculate the ripple voltage at the output of your supply when it has a 50 Ohms load.
3. Design the circuit of a full-wave rectifier using center tapped transformer to obtain an output d.c. voltage of V_{dc} = 18 at 200mA and V_{dc} no load equals to 20V. Assume suitable value of rf and transformer resistance and also mention transformer rating and sketch the input and output waveforms.

5. Exercises on Transistor Common Base Characteristics

Common base (CB) configuration (or) Grounded base configuration. In this circuit arrangement, input is applied between the emitter and base, and output is taken from the collector and base. Here, the base of the transistor is common to both input and output circuits and hence the name common base connection.

Plot the input and output characteristics of a transistor in common base configuration shown in Figure 5.1 and to compute the h – parameters.

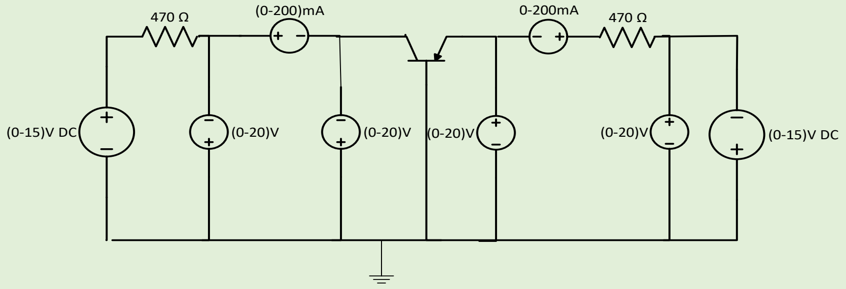


Figure 5.1: Common base configuration

Try

1. Plot the *I-V* curves of n-p-n transistor based on observations. What collector-emitter voltage (***V_{CE}***) does the transition from saturation to active region occur approximately?
2. Demonstrate the characteristics of common base using p-n-p transistor to determine the h-parameters.
3. For the common base circuit, determine **I_C** and **V_{CB}**. Assume the transistor to be of Germanium.

6. Exercises on Transistor Common Emitter Characteristics

The configuration in which the emitter is connected between the collector and base is known as a common emitter configuration. The variation of emitter current (I_B) with Base-emitter voltage (V_{BE}), keeping Collector-emitter voltage (V_{CE}) constant.

Plot the input and output characteristics of a transistor in common emitter configuration from the Figure 6.1 and compute the h – parameters.

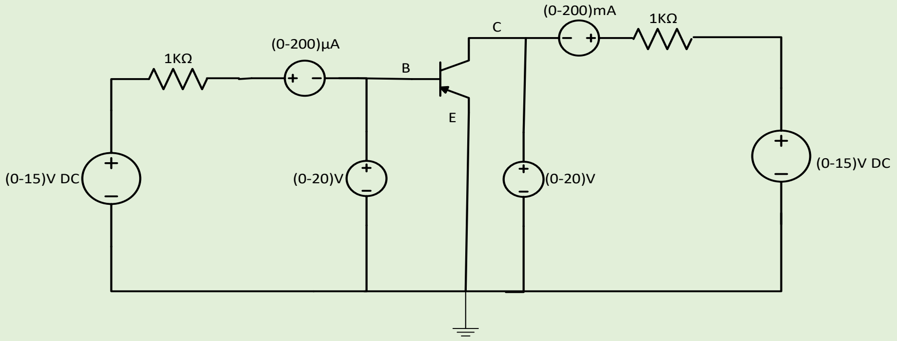


Figure 6.1: Common emitter configuration

Try

1. Design a circuit which acts as an electronic switch using common emitter configuration.
2. The output characteristics of a transistor connected in common emitter mode. Determine the value of I_C when V_{CE} = 15V. Also determine the value of I_C when V_{CE} is changed to 10 V.
3. Demonstrate the characteristics of Common Emitter using p-n-p transistor to determine the h - parameters.

7. Exercises on Frequency Response of Common Emitter Amplifier

The voltage gain of a CE amplifier varies with signal frequency. It is because the reactance of the capacitors in the circuit changes with signal frequency and hence affects the output voltage.

Obtain the frequency response curve of the below Figure 7.1 amplifier and determine the mid frequency gain, Amid, lower and higher cutoff frequency of the amplifier.

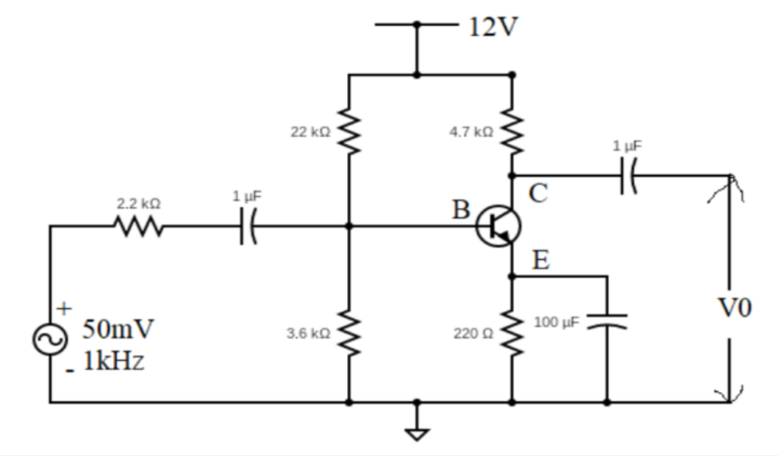


Figure 7.1: Common emitter amplifier

Try

1. In a CE germanium transistor amplifier using self-bias circuit, R_C =2.2kΩ, β= 50, VCC=9V and the operating point is required to be set at I_C= 2 mA and V_{CE}= 3V. Determine the values of R1, R2 and RE.
2. Compute the voltage gain from the Figure 7.2 by Assuming β=150.

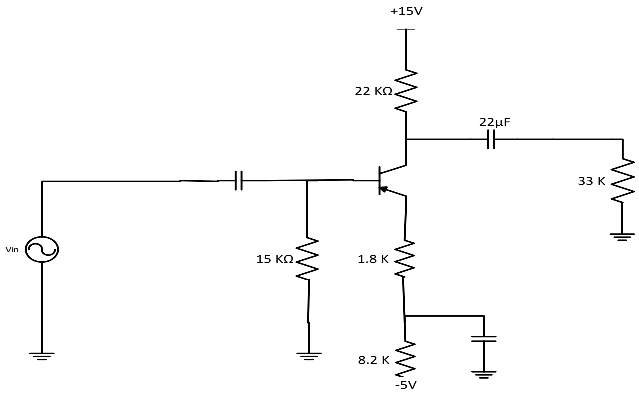


Figure 7.2

3. Measure the voltage gain for the Figure 7.2. Adjust the input signal to approximately 10mv amplitude, with a frequency of 100 kHz. What is the voltage swing?

4. Determine the input impedance and voltage gain for the circuit shown in Figure7.3. Also determine V_{load}, if V_{in} = 20 mV peak. Assume β=100.

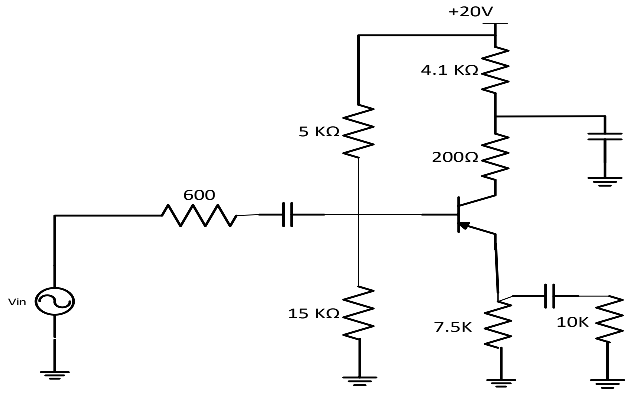


Figure 7.3

8. Exercises on Frequency Response of Common Collector Amplifier

In common collector amplifier as the collector resistance is made to zero, the collector is at AC ground that is the reason for which the circuit is also called as grounded-collector amplifier or this configuration has voltage gain close to unity and hence a change in base voltage appears as an equal change across the load at the emitter.

1. Design a common collector transistor (n-p-n) amplifier circuit as shown in Figure 8.1.
2. Obtain the frequency response curve of the amplifier and determine the mid frequency gain, Amid, lower and higher cutoff frequency of the amplifier circuit as shown in Figure 8.1.

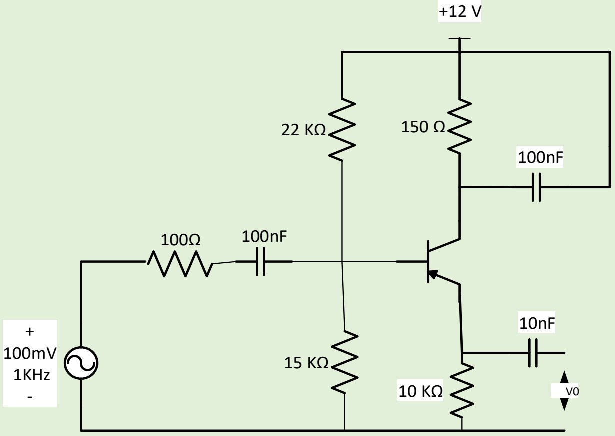


Figure 8.1: Common collector amplifier

Try

1. Design a common-collector amplifier using the 2N3904 transistor that meets the following specifications: I_C = 1mA, VCC = 20V, R_{in} = 70kΩ, R_L = 510Ω, V_{in} = 10mV @ 10kHz. Determine the value of R_E.
2. Design a Common emitter amplifier using PNP transistor.
3. For the Common Collector amplifier showed in Figure 8.2, find the voltage gain (β=120).

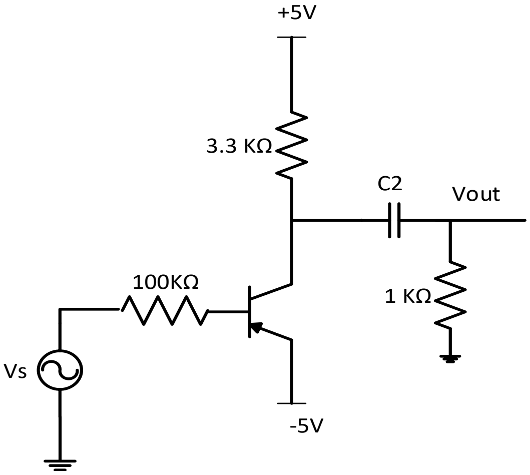


Figure 8.2

9. Exercises on UJT Characteristics

The UJT consists of an n-type silicon semiconductor bar with an electrical on each end. The terminals of these connections are called Base terminals (B1 and B2). Near to base B2, a p-n junction is formed between a p-type emitter and the n-type silicon bar. The terminal of this junction is called emitter terminal (E). Since the device has three terminals and one p-n junction, for this region this is called as a Uni junction Transistor (UJT).

Simulate and analyze the characteristics of a UJT (Uni-junction Transistor) from the given below Figure 9.1.

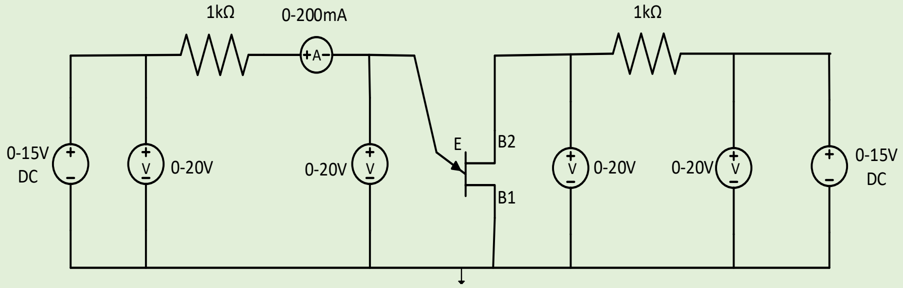


Figure 9.1: Uni-junction Transistor characteristics

Try

1. Design an UJT relaxation oscillator to generate a sawtooth waveform at a frequency of 600 Hz. Assume the supply voltage V_{BB}=18 V, V_p=2.9 V and V_V =1.118 V.
2. Plot the volt-ampere characteristics of UJT 2N46 with R_E= RB2= 1 K Ω and RB1=500 Ω.
3. Design and observe the characteristics of relaxation oscillator using Uni-Junction Transistor.

10. Exercises on SCR Characteristics

The SCR has three p-n junctions, and four layer of p and n-type semiconductor joined alternatively to get p-n-p-n device. The three terminals are taken one from outer p – type layer called anode (A), second from the outer n – type layer called cathode (K) and the third from the internal p –type layer called gate (G).

To observe and analyze the *V*-I Characteristics of Silicon Control Rectifier (SCR) and determine holding, latching current and break over voltage of given SCR in Figure 10.1.

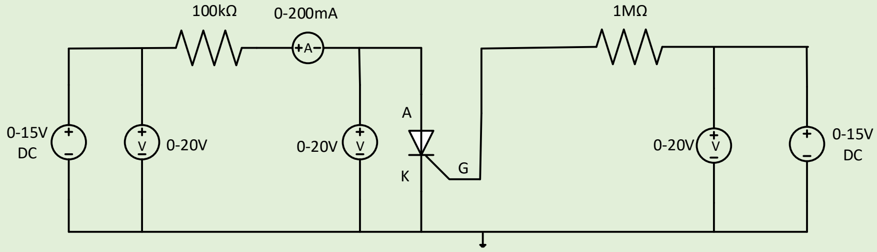


Figure 10.1: Silicon Control Rectifier characteristics

Try

1. The SCR has gate trigger voltage V_T=0.7v, gate trigger current I_T=7mA and holding current I_H=6mA.Find the input voltage that triggers the SCR?
2. If 220Ω resistor is connected in series with the gate of an SCR. The gate current required to fire the SCR is 7mA. What is the input voltage (V_{in}) required to fire the SCR?
3. A SCR full wave rectifier is connected across a sinusoidal voltage of 400 sin314t, the RMS value of the current flowing through the device is 20 A. Find the power rating of the SCR.

11. Exercises on FET Characteristics

The functioning of Junction Field Effect Transistor depends upon the flow of majority carriers (electrons or holes) only. Basically, JFETs consist of an n type or p-type silicon bar containing p-n junctions at the sides.

To study the drain and transfer characteristics of FET and find the drain resistance, trans-conductance and amplification factor for the Figure 11.1

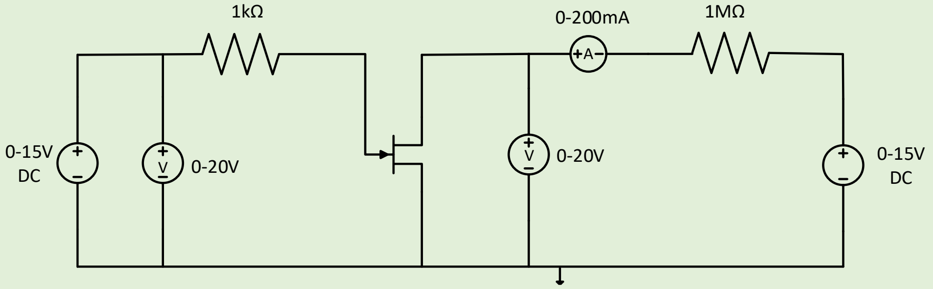


Figure 11.1: Field effect transistor characteristics

Try

1. Plot the drain and transfer characteristics of P-channel JFET BFW11/10/BF245A with RL= 50 Ω.
2. A JFET has the following parameters: IDSS = 32 mA; VGS (off) = – 8V; VGS = – 4.5V. Find the value of drain current.
3. An N-channel JFET having Vp=-4V and V_{DSS} = 10mA is used in the circuit of Figure 11.2. The parameter values are V_{DD} = 18V, Rs=2k Ω, R1=450k Ω and R2=90kΩ. Determine I_D and V_{DS}.

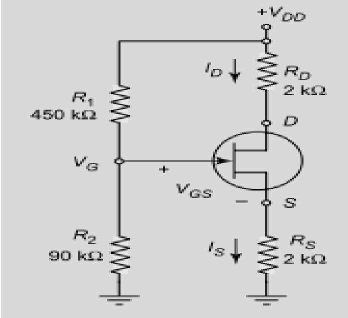


Figure 11.2

12. Exercises on frequency response of Common Source amplifier

When the input signal is applied at the gate terminal and source terminal, then the output voltage is amplified and obtained across the resistor at the load in the drain terminal. This is called a common source amplifier.

Obtain the frequency response of common source FET amplifier and also measure the voltage gain and bandwidth from the Figure 12.1.

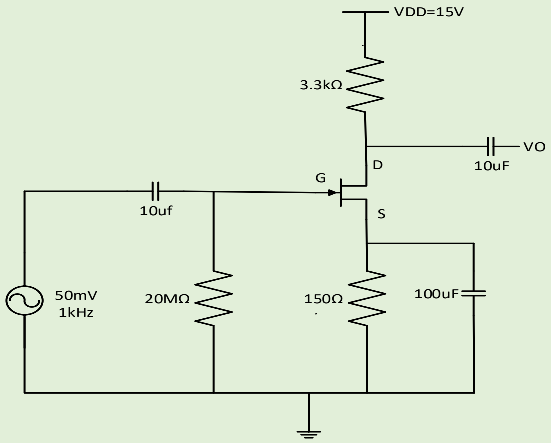


Figure 12.1: Common source amplifier

Try

1. Plot the frequency response of FET BFW11 amplifier with C2=5μF with triangular/square i/p.
2. Plot frequency response of P-Channel JFET RG1= 4.1K, RG2= 9.4K with square i/p.
3. Design a FET amplifier in the common-source configuration uses a load resistance of 500 kV. The ac drain resistance of the device is 100 kV and the transconductance is 0.8 mA/V. Calculate the voltage gain.

13. Exercises on Frequency Response of Common Drain Amplifier

A common drain amplifier is one in which the input signal is applied to the gate and the output is taken from the source, making the drain common to both. Because it is common, there is no need for a drain resistor

To obtain frequency response of common drain FET amplifier and measure the voltage gain and bandwidth of CD amplifier from the Figure 13.1.

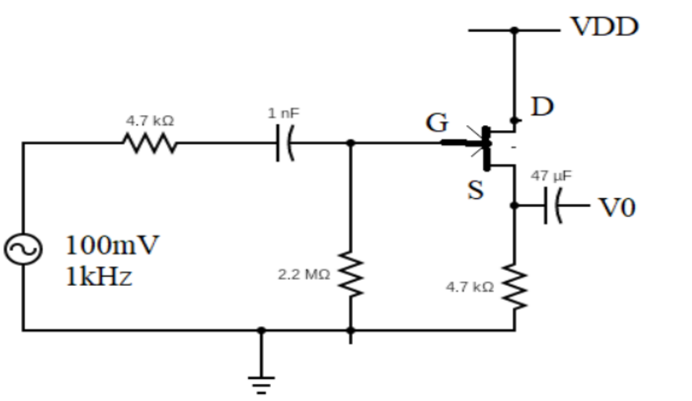


Figure 13.1: Common drain amplifier

Try

1. Design a MOSFET amplifier and plot frequency response based on the given specifications.

   Both the input and the output should be AC coupled

   Dual Supply Voltage is ± 5V

   Load Resistance, RL = 100 ohms

   0 to peak output swing is greater than or equal to 2V

   voltage gain is 50

   input resistance is 10k ohms.

2. For the source follower shown in Figure13.2. Determine the input impedance and output voltage.

   Assume V_{in}=100mV, IDSS=30 mA, V_{GS} (off)=−2 V.

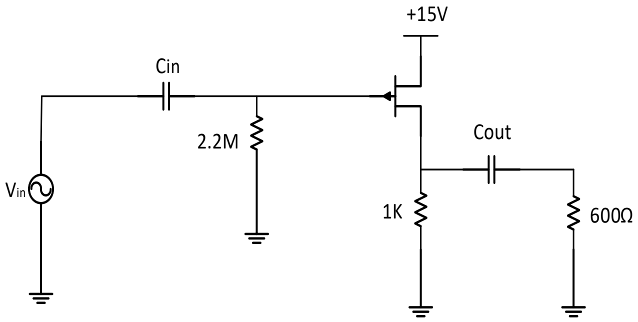


Figure13.2

3. Consider the amplifier in Figure 13.3. Find the input resistance and voltage gain of the circuit, given gm = 0.5mΩ-1 and Rds = 0.2MΩ.

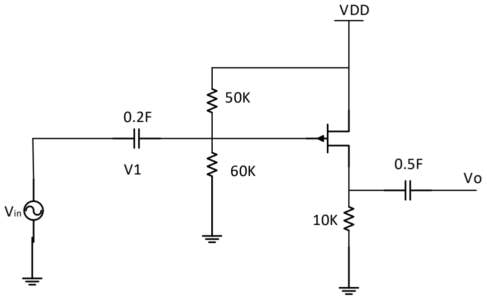


Figure13.3

14. Exercises on Clippers and Clampers

A clipper circuit in which the diode is connected in series to the input signal and biased with positive reference voltage V_r and that attenuates the positive portions of the waveform, is termed as Positive Series Clipper with positive V_r.

14.1 Positive and negative clippers with and without biasing

A biased clipper comes in handy when a small portion of positive or negative half cycles of the signal voltage is to be removed. When a small portion of the negative half cycle is to be removed, it is called a biased negative clipper.

Design non-linear wave shaping circuits as clippers shown in Figure14.1.1 and Figure14.1.2

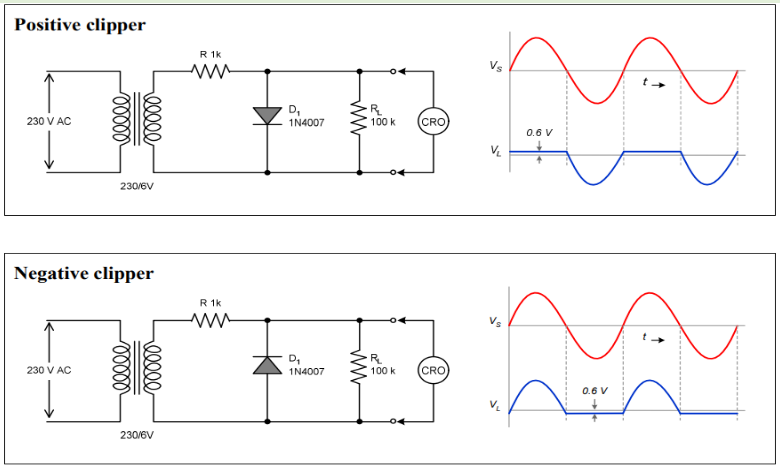


Figure14.1.1: Positive and negative clippers without biasing

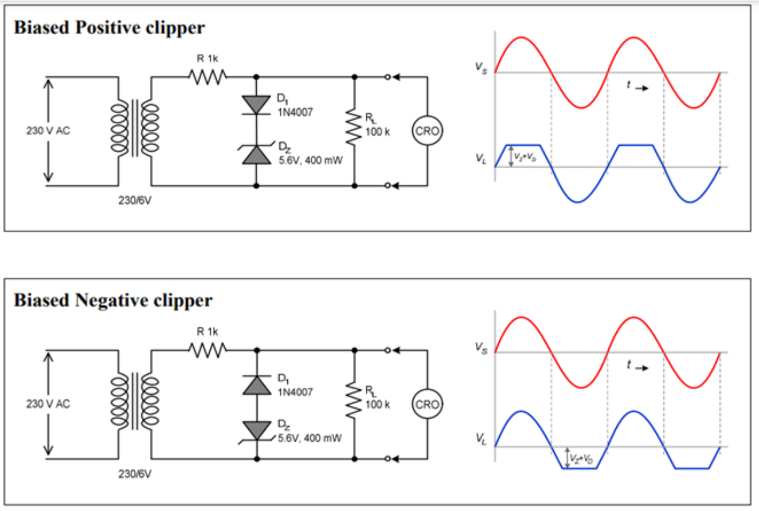


Figure 14.1.2: Positive and negative clippers with biasing

14.2 Positive clamper

A clamper circuit can be defined as the circuit that consists of a diode, a resistor and a capacitor that shifts the waveform to a desired DC level without changing the actual appearance of the applied signal. Clamping circuit restores the DC level. When a negative peak of the signal is raised above to the zero level, then the signal is said to be positively clamped.

Design non-linear wave shaping circuits as positive clampers shown in Figure 14.2.

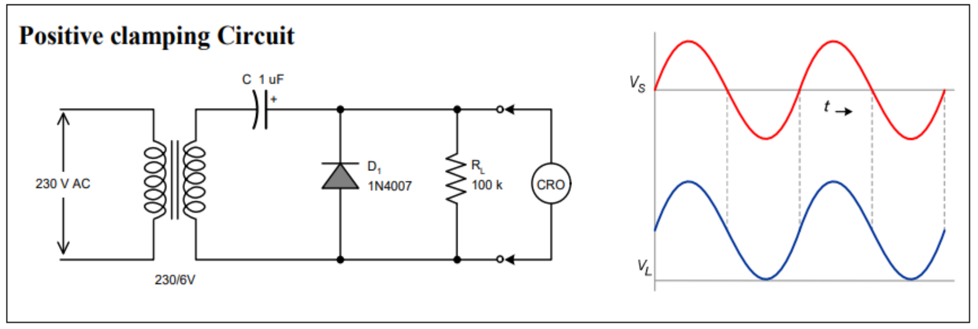


Figure 14.2: Positive clamper

14.3 Negative clamper

A negative clamper circuit is one that consists of a diode, a resistor and a capacitor and that shifts the output signal to the negative portion of the input signal. The figure below explains the construction of a negative clamper circuit.

Design non-linear wave shaping circuits as negative clampers shown in Figure 14.3.

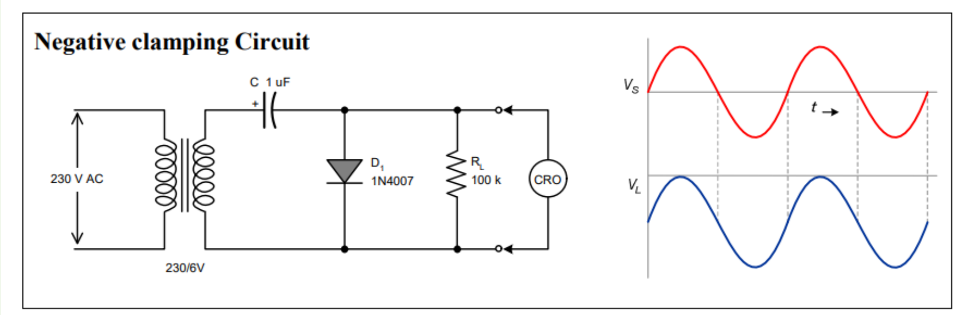


Figure 14.3: Negative clamper

Try

1. For the given circuit in Figure 14.6 for a 20 V_{peak} sinusoidal input v_i, what is the value of v_i at which the clipping begins?

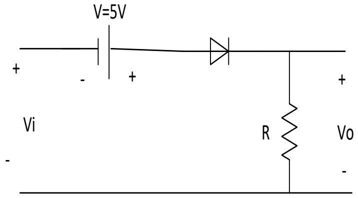


Figure14.6

2. For the given circuit in Figure 14.7, what is the minimum peak value of the output waveform if the input waveform is 10V square wave with switching time of 1 second?
Assume that the input switches between +10V and -10V DC levels.

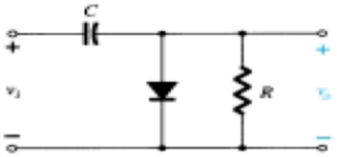


Figure14.7

3. For the given circuit in Figure 14.8 and input waveform, find the peak value of the output.

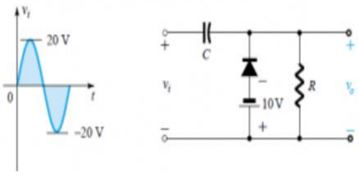


Figure14.8

15. Final Notes

Student can have any one of the following certifications:

- NPTEL – Semiconductor Devices and Circuits
- NPTEL – Basic Electronics

V. REFERENCE BOOKS:

1. J. Millman, C.C.Halkias, Millman‘s, “Integrated Electronics”, Tata McGraw Hill, 2nd edition, 2001.
2. J. Millman, C.C.Halkias and Satyabrata Jit, “Millman‘s Electronic Devices and Circuits”, Tata McGraw Hill, 2nd edition, 1998.
3. Mohammad Rashid, “Electronic Devices and Circuits”, Cengage learning, 1st edition, 2014.
4. DavidA. Bell,“ElectronicDevicesandCircuits”,OxfordUniversityPress,5th edition, 2009.

VI. WEB REFERENCES:

1. https://archive.org/details/ElectronicDevicesCircuits
2. http://www.tedpavlic.com/teaching/osu/ece327/

DIGITAL SYSTEM DESIGN LABORATORY

III Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC06	Core	L	T	P	C	CIA	SEE	Total
		0	0	2	1	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 28				Total Classes: 28		

I. COURSE OVERVIEW:

The laboratory strives in researching the logic design and related fields. Digital logic testers are used to provide students with practical training and familiarize themselves with the various functions of logic gates and using integrated components to complete circuitry functions and develop an interest in digital logic and enlighten them in the abilities of deduction. The lab allows students to conduct actual gate-level experiments to increase student interest and develop skills to design digital gates using VHDL.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Design of combinational circuits using Verilog Hardware Description Language.
II. Implementation of Sequential circuits using Verilog Hardware Description Language.
III. Demonstration of different case studies for Verilog.

III. COURSE OUTCOMES:

After the completion of the course, studens should be able to

CO 1 Utilize the concept of Boolean algebra to verify the truth table of Booleanexpressions using logic gates in Hardware Description Language.

CO 2 Make use of dataflow, structural and behavioral modelling styles of HDLfor simulating the combinational logic circuits.

CO 3 Analyze the truth tables and characteristic equations of flip flops for thefunctional simulation and timing analysis of sequential circuits.

CO 4 Construct the synchronous and asynchronous sequential circuits using theflip flops.

CO 5 Model a finite state machine with melay and moore machines for detectinga given sequence.

CO 6 Examine the functionality of real time traffic light controller, chess clockcontroller FSM, elevator operations using HDL code.

IV. COURSE CONTENT:

EXERCISES FOR DIGITAL SYSTEM DESIGN LABORATORY

Note: Students are encouraged to bring their own laptops for laboratory practice sessions.

1. Getting Started Exercises

1.1 Basic Gates

1. Install Xilinx vivado on your machine.
2. Write a VHDL program using vivado simulator for:
   - Verifying the functionality of design under test (DUT) by writing test bench to pass the stimulus
   - Synthesize the register transfer logic (RTL) using Xilinx XST synthesis tool
   - Elaborate the design and generate bit file to dump RTL code into the Zynq series and ZedBoard FPGA
   - Verify the functionality of the design under test (DUT) on FPGA board

1.2 AND-OR-INVERT AND OR-AND-INVERT LOGIC

Write VHDL code to implement the function expressed by the following logic equation

$Y = \overline{a_0 b_0 + a_1 b_1 + a_2 b_2}$

Use only simple signal assignment statements in your VHDL data flow model.

Hints

Use and, or and compliment operators for implementation of the logic.

```
Input a0, a1, a2, a3, b0, b1, b2, b3;
Output y;

/**   Data  flow model for AOI logic */
entity AOI port (
         a0, a1, a2, a3, b0, b1, b2, b3: in std_logic;
         y: out std_logic);
end AOI;

/**   architecture body  */
architecture arch_AOI of AOI is

begin
    Y = not ((a0 & b0) | (a1 & b1) | (a2 & b2) | (a3 & b3));
    . . .

End arch_AOI;

//Write the test bench for providing the stumulus
```

```
entity tb_AOI port
end tb_AOI;

architecture arch_AOI of AOI is
   signal a0, a1, a2, a3, b0, b1, b2, b3: std_logic := ‘0’;
   signal y: std_logic;

   component AOI port (
         a0, a1, a2, a3, b0, b1, b2, b3: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: AOI port map (a0, a1, a2, a3, b0, b1, b2, b3, y);

   a0 = ‘0’; b0 = ‘0’;
   a1 = ‘0’; b1 = ‘0’;
   a2 = ‘0’; b2 = ‘0’;
   a3 = ‘0’; b3 = ‘0’;
   wait for 10 ns;

   a0 = ‘0’; b0 = ‘1’;
   a1 = ‘1’; b1 = ‘0’;
   a2 = ‘0’; b2 = ‘1’;
   a3 = ‘1’; b3 = ‘0’;
   wait for 10 ns;

   . . .
   . . .

   a0 = ‘1’; b0 = ‘1’;
   a1 = ‘1’; b1 = ‘1’;
   a2 = ‘1’; b2 = ‘1’;
   a3 = ‘1’; b3 = ‘1’;
   wait for 10 ns;

End architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

Develop a model for a general or- and-invert gate, with two std_logic vector input ports a and b and a standard-logic output port y. The output of the gate is

1.3 Product of Sum Boolean expression

Write a program to perform product of sum which evaluates the given Boolean expressions and write the test-bench to verify the functionality with all possible combinations of the input.

Hints

```
/**
 * Consider the POS expression for F = π(0, 1,2,4,8,9,10,12,13)   */
Input a, b, c, d;
Output F;

Minimize the logic for the given F          // POS logic
Implement the logic using basic gates

/**   Declare the port signals */
entity POS port (
         a, b, c, d: in std_logic;
         f: out std_logic);
end POS;

/**   architecture body  */
architecture arch_POS of POS is

Begin
    Y = F(a, b, c, d);
    . . .

End arch_POS;

//Write the test bench for providing the stumulus

entity tb_POS port
end tb_POS;

architecture arch_POS of POS is
   signal a, b, c, d: std_logic := ‘0’;
   signal f: std_logic;

   component POS port (
         a, b, c, d: in std_logic;
         f: out std_logic);
   end component;

begin
   DUT: POS port map (a, b, c, d, f);

   a = ‘0’; b =’0’; c = ‘0’; d =’0’;
   wait 10 ns;

   a = ‘0’; b =’0’; c = ‘0’; d =’1’;
   wait 10 ns;

   . . .
   . . .

   a = ‘1’; b =’1’; c = ‘1’; d =’1’;
   wait 10 ns;


End architecture
```

```
Provide the stimulus for all 16 possible combinations starting from 0000 to 1111
Simulate the DUT with the given stimulus
Verify the output using waveforms
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

Modify the POS expression including the don’t care cases F = π(0, 1,2,4,9,10,12,13) + d(3, 7, 11)

1.5 Sum of Product Boolean expression

Write a program to perform sum of product which evaluates the given Boolean expressions and write the test-bench to verify the functionality with all possible combinations of the input.

Hints

```
/**
 * Consider the POS expression for F = Σ(0, 1,2,4,8,9,10,12,13)   */
Input a, b, c, d;
Output F;

Minimize the logic for the given F          // POS logic
Implement the logic using basic gates

/**   Declare the port signals */
entity SOP port (
         a, b, c, d: in std_logic;
         f: out std_logic);
end SOP;

/**   architecture body  */
architecture arch_SOP of SOP is

Begin
    Y = F(a, b, c, d);
    . . .

End arch_SOP;

//Write the test bench for providing the stumulus

entity tb_SOP port
end tb_SOP;

architecture arch_SOP of SOP is
   signal a, b, c, d: std_logic := ‘0’;
   signal f: std_logic;

   component SOP port (
         a, b, c, d: in std_logic;
         f: out std_logic);
   end component;

begin
   DUT: SOP port map (a, b, c, d, f);

   a = ‘0’; b =’0’; c = ‘0’; d =’0’;
   wait 10 ns;

   a = ‘0’; b =’0’; c = ‘0’; d =’1’;
   wait 10 ns;

   . . .
   . . .

   a = ‘1’; b =’1’; c = ‘1’; d =’1’;
   wait 10 ns;

End architecture

Simulate the DUT with the given stimulus

Verify the output using waveforms

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

Modify the POS expression including the don’t care conditions F = Σ(0, 1,2,4,9,10,12,13) + d(3, 7, 11)

1.6 Code Conversions

To familiarize students with code converters. The student should also become familiar with gray to binary conversion, binary to gray conversion.
Given the sequence of three-bit Gray code as (000, 001, 011, 010, 110, 111, 101, 100)

A given Gray code number can be converted into its binary equivalent by going through the following steps:
1. Begin with the most significant bit (MSB). The MSB of the binary number is the same as the MSB of the Gray code number.
2. The bit next to the MSB (the second MSB) in the binary number is obtained by adding the MSB in the binary number to the second MSB in the Gray code number and disregarding the carry, if any.
3. The third MSB in the binary number is obtained by adding the second MSB in the binary number to the third MSB in the Gray code number. Again, carry, if any, is to be ignored.
4. The process continues until we obtain the LSB of the binary number.

Hints

```
/**   Declare the port signals */
entity gray2binary port (
         gray_code: in std_logic_vector(3 downto 0);
         binary_code: out std_logic_vector(3 downto 0));
end gray2binary;

/**   architecturebody  */
architecture arch_gray2binary of gray2binary is

Begin
    binary_code[3] = gray_code[3];
    binary_code[2] = gray_code[3] xor gray_code[2];
    . . .
```

```
End arch_gray2binary;

//Write the test bench for providing the stumulus

entity tb_gray2binary port
end tb_gray2binary;

architecture arch_gray2binary of gray2binary is
   signal gray_code: std_logic_vector(3 downto 0) := “0000”;
   signal binary_code: std_logic_vector(3 downto 0);

   component gray2binary port (
         gray_code: in std_logic_vector(3 downto 0);
         binary_code: out std_logic_vector(3 downto 0));
   end component;

begin
   DUT: gray2binary port map (gray_code, binary_code);

   Process
   begin
        gray_code = gray_code + 1;
        Wait for 10 ns;

   End process;

End architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

1. Design and implement the binary to gray code conversion
2. Design and implement the binary to excess 3-code conversion
3. Design and implement the excess 3-code to binary conversion

2. Exercises on Gate Realization

To be proficient in programming, you need to be able to:

1 Construct basic logic gates realization using NAND gates and NOR gates

2 Utilize min no of 2 input NAND Gates to implement three input NAND gate using

3 Build a user defined logic gate for the given specifications

2.1 Basic gates realization using NAND gate

Realize the inverter gate logic using NAND gate

NAND gate is actually a combination of two logic gates i.e. AND gate followed by NOT gate. So its output is complement of the output of an AND gate. This gate can have minimum two inputs. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, Ex-OR, Ex-NOR, NOR. So this gate is also called as universal gate.

Hint

```
/**   Declare the port signals */
entity inv_nand port (
         i: in std_logic;
         y: out std_logic);
end inv_nand;

/**   architecture body  */
architecture arch_inv_nand of inv_nand is

component nand_gate port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: nand_gate port map (i, i, y);
    . . .

End arch_inv_gate;

//Write the test bench for providing the stumulus

entity tb_inv_nand port
end tb_inv_nand;

architecture arch_inv_nand of inv_nand is
   signal i: std_logic := ‘0’;
   signal y: std_logic;

   component inv_nand port (
         i: in std_logic;
         y: out std_logic);
   end component;
```

```
begin
   DUT: inv_nand port map (i, y);

   i =’0’ after 10 ns;
   i =’1’ after 10 ns;

   . . .
   . . .

end architecture
```

Try

1. Realize the AND gate logic using NAND gate
2. Realize the OR gate logic using NAND gate

2.2 Gate realization using NOR gate

Realize the inverter gate logic using NOR gate

Hint

```
/**   Gate level model for */
entity inv_nor port (
         i: in std_logic;
         y: out std_logic);
end inv_nor;

/**   architecture body   */
architecture arch_inv_nor of inv_nand is

component nor_gate port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: nor_gate port map (i, i, y);
    . . .

End arch_inv_nor;

//Write the test bench for providing the stumulus

entity tb_inv_nor port
end tb_inv_nor;

architecture arch_inv_nor of inv_nor is
   signal i: std_logic := ‘0’;
   signal y: std_logic;

   component inv_nor port (
         i: in std_logic;
         y: out std_logic);
   end component;

begin
```

```
   DUT: inv_nor port map (i, y);

   i =’0’ after 10 ns;
   i =’1’ after 10 ns;

   . . .
   . . .

End architecture
```

Try

1. Realize the AND gate logic using NOR gate
2. Realize the OR gate logic using NOR gate

2.3 XOR gate realization using minimum number of NAND gates

Realize XOR gate using minimum number of NAND gates

Hint

```
/**   Declare the port signals */
entity xor_nand port (
         a, b: in std_logic;
         y: out std_logic);
end xor_nand;

/**   architecture body  */
architecture arch_xor_nand of xor_nand is

component nand_gate port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: nand_gate port map (a, b, y);
    . . .

End arch_xor_nand;

//Write the test bench for providing the stumulus

entity tb_xor_nand port
end tb_xor_nand;

architecture arch_xor_nand of xor_nand is
   signal a, b: std_logic := ‘0’;
   signal y: std_logic;

   component xor_nand port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
```

```
   DUT: xor_nand port map (a, b, y);

   a =’0’ after 10 ns;
   b =’0’ after 10 ns;

   . . .
   . . .

End architecture
```

Try

Realize XNOR gate using minimum number of NAND gates

2.4 Three input NAND gate using min no of 2 input NAND Gate

To implement 3 input NAND gate realization using minimum number of NAND gates

```
a) A and B to the first NAND gate
b) Output of first Nand gate is given to the two inputs of the second NAND gate (this
basically realizes the inverter functionality)
c) Output of second NAND gate is given to the input of the third NAND gate, whose
otherinput is C ((A NAND B) NAND (A NAND B)) NAND C Thus, can be implemented using '3'2-
input NAND gates.
```

Hints:

Assume three inputs of the NAND gate are A, B and C and connect these inputs as

```
/**   Declare the port signals */
entity nand3 port (
          a, b, c: in std_logic;
          y: out std_logic);
end nand3;

/**   architecture body  */
architecture arch_nand3 of nand3 is

component nand2_gate port (
          a, b: in std_logic;
          y: out std_logic);
   end component;

begin
   DUT: nand2_gate port map (a, b, y);
    . . .

End arch_nand3;

//Write the test bench for providing the stumulus

entity tb_nand3 port
end tb_nand3;

architecture arch_tb_nand3 of tb_nand3 is
   signal a, b: std_logic := ‘0’;
   signal y: std_logic;
   component nand2 port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: nand2 port map (a, b, y);

   a =’0’ after 10 ns;
   b =’0’ after 10 ns;

   . . .
   . . .

   a = ‘1’ after 10 ns;
   b = ‘1’ after 10 ns;

End architecture
```

Try:
To implement XNOR gate realization using minimum number of NAND gates

2.5 User defined logic gate (Muller-C element cell)

Develop a behavioral model for a two-input Muller-C element cell, with two input ports and one output, all of type bit. The inputs and outputs are initially ‘0’. When both inputs are ‘1’, the output changes to ‘1’. It stays ‘1’ until both inputs are ‘0’, at which time it changes back to ‘0’. Your model should have a propagation delay for rising output transitions of 3.5 ns, and for falling output transitions of 2.5 ns.

Hints:
Assume three inputs of the NAND gate are A, B and C and connect these inputs as

```
Take inputs A and B
Extract the truth table and Boolean expression as per the specifications
Implement the gate using VHDL model
Write the logic for selecting the stimulus for verifying the logic

/**   Declare the port signals */
entity mc_cell port (
         a, b, c: in std_logic;
         y: out std_logic);
end mc_cell;

/**   architecture body  */
architecture arch_mc_cell of mc_cell is

component mc_cell port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: nand2_gate port map (a, b, y);
    . . .

End arch_mc_cell;
```

```
//Write the test bench for providing the stumulus

entity tb_mc_cell port
end tb_mc_cell;

architecture arch_tb_mc_cell of tb_mc_cell is
   signal a, b: std_logic := ‘0’;
   signal y: std_logic;

   component mc_cell port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: mc_cell port map (a, b, y);

   a =’0’ after 10 ns;
   b =’0’ after 10 ns;

   . . .
   . . .

   a = ‘1’ after 10 ns;
   b = ‘1’ after 10 ns;

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

Develop a behavioral model for a two-input Muller-C element cell, with two input ports and one output, all of type bit. The inputs and outputs are initially ‘1’. When both inputs are ‘0’, the output changes to ‘0’. It stays ‘0’ until both inputs are ‘1’, at which time it changes back to ‘1’.

3. Exercises on Multiplexers and Demultiplexers

To be proficient in programming, you need implement the following digital circuits:

1. Implementation of 2x1, 4x1 multiplexers, demultiplexers
2. Realization of higher order multiplexers using lower order multiplexers
3. Realization of basic gates using 2x1 multiplexer

3.1 Implementation of 2x1, 4x1 multiplexers

Develop a behavioral model for a two-input multiplexer, with ports of type bit and a propagation delay from data or select input to data output of 5 ns. You should declare a constant for the propagation delay, rather than writing it as a literal in signal assignments in the model

The inputs to the MUX are data inputs I1, I0 and a one control input SEL(s) The single output is Y

Hints

```
/**     Implementation of 2x1 multiplexer   **/

Declare the inputs I0, I1 and S
Declare the output Y.

//Write the logic for selecting the data depends on the select line and pass to
the output

entity mux_2x1 port (
         i0, i1, s: in std_logic;
         y: out std_logic);
end mux_2x1;

/**   architecture body  */
architecture arch_mux_2x1 of mux_2x1 is

   component nand port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

   component inv port (
         i: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT1: inv port map (s, sb);
   DUT2: nand2_gate port map (i0, sb, s1); . . .

   . . .
   . . .

End arch_mux2x1;

//Write the test bench for providing the stumulus
entity tb_mux2x1 port
end tb_mux2x1;

architecture arch_tb_mux2x1 of tb_mc_mux2x1 is
   signal i0, i1, s: std_logic := ‘0’;
   signal y: std_logic;

   component mux2x1 port (
         i0, i1, s: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: mux2x1 port map (i0, i1, s, y);

   s =’0’ after 10 ns, ‘1’ after 10 ns;
   process
   begin
     i0 = ~i0;
     wait for 10ns;
     i1 = ~i1;
     wait for 15ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA


Plot the wave form for all possible select lines

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

Develop a behavioral model for a four-input multiplexer, with ports of type bit and a propagation delay from data or select input to data output of 4.5 ns. You should declare a constant for the propagation delay, rather than writing it as a literal in signal assignments in the model.

3.2 Implementation of 2x1, 4x1 demultiplexers

Build a behavioral model for a two-input demultiplexers, with ports of type bit and a propagation delay from data or select input to data output of 5 ns.

Hints

```
/**
  Implementation of 2x1 demultiplexer.
 **/

Declare the inputs I and S
Declare the output Y0, Y1.

//Write the logic for selecting the data depends on the select line and pass to
the output

entity dmux_1x2 port (
         i, s: in std_logic;
         y0, y1: out std_logic);
end dmux_1x2;

/**   architecture body  */
architecture arch_dmux_1x2 of dmux_1x2 is

begin
   process(I,s)
   begin
      case S is
        when ‘0’: y0 = I; y1 = ‘z’;
        . . .
        . . .

End arch_dmux1x2;
```

```
//Write the test bench for providing the stumulus

entity tb_dmux1x2 port
end tb_dmux1x2;

architecture arch_tb_dmux1x2 of tb_dmux1x2 is
   signal i, s: std_logic := ‘0’;
   signal y0, y1: std_logic;

   component dmux1x2 port (
         i, s: in std_logic;
         y0, y1: out std_logic);
   end component;

begin
   DUT: dmux1x2 port map (i, s, y0, y1);

   s =’0’ after 10 ns, ‘1’ after 10 ns;
   process
   begin
     i = ~i;
     wait for 10ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA

Plot the wave form for all possible select lines

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

1 Modify the program to function as 1x4 demultiplexers

2 Modify the program to function as 1x8 demultiplexers

3.3 Realization of higher order multiplexers using lower order multiplexers

Realize the higher order multiplexers using lower order multiplexers. Write the VHDL model for the realized circuits. Simulate the test benches for the corresponding and verify the design under test by plotting the waveforms. Figure 1 shows realization of 4x1 mux using 2x1 mux.

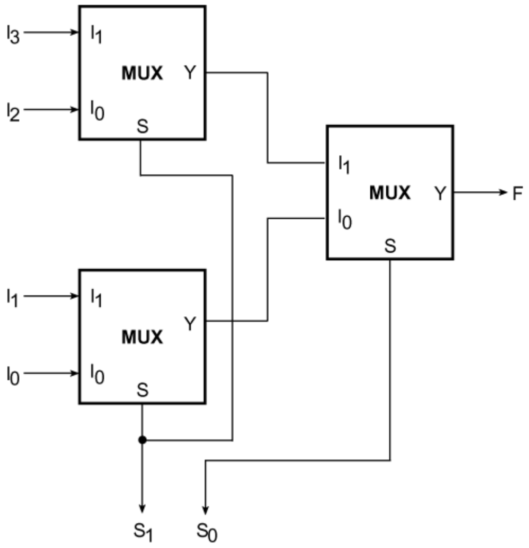


Figure 1: Realization of 4x1 mux using 2x1 mux

```
/**
 Realize the lower order multiplexers for design of higher order multiplexers.
 **/

In work library simulate the lower order multiplexer

For the realized higher order multiplexer, instance the component in the declaration part
of the architecture

By using positional or name mapping instance the component with the signals
```

Hint

```
//Write the logic for selecting the data depends on the select line and pass to
the output

entity mux_4x1 port (
         i0, i1, i2, i3: in std_logic;
         s: in std_logic_vector(1 donwto 0);
         y: out std_logic);
end mux_4x1;

/**   architecture body  */
architecture arch_mux_4x1 of mux_4x1 is

   component mux2x1 port (
         i0, i1, s: in std_logic;
         y: out std_logic);
   end component;
   // declare intermediate signals

begin
   DUT1: mux2x1 port map (i0, i1, s(0), s1);
   DUT2: mux2x1 port map (i2, i3, s(1), s2);
   . . .
   . . .
   . . .

End arch_mux4x1;
```

```
//Write the test bench for providing the stumulus

entity tb_mux4x1 port
end tb_mux4x1;

architecture arch_tb_mux4x1 of tb_mux4x1 is
   signal i0, i1, i2, i3: std_logic := ‘0’;
   signal s: std_logic_vector(1 donwto 0) := “00”;
   signal y: std_logic;

   component mux4x1 port (
         i0, i1, i2, i3: in std_logic;
         s: in std_logic_vector(1 donwto 0);
         y: out std_logic);
   end component;

begin
   DUT: mux4x1 port map (i0, i1, i2, i3, s, y);

   s =’00’ after 10 ns, ‘01’ after 10 ns . . .;
   process
   begin
     i0 = ~i0;
     wait for 10ns;
     i1 = ~i1;
     wait for 12ns;
     i2 = ~i2;
     wait for 14ns;
     i3 = ~i3;
     wait for 16ns;
   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

3.4 Realization of basic gates using 2x1 multiplexer

Realize all the basic gates like AND and inverter using 2x1 multiplexer

```
/**   Realize the multiplexers for function as basic logic gates **/

In work library simulate the 2x1 multiplexer

For the realized basic gates using 2x1 multiplexer, instance the component in the
declaration part of the architecture
```

Hints

```
//Write the logic for selecting the data depends on the select line and pass to
the output

entity mux_and port (
         a, b: in std_logic;
         y: out std_logic);
```

```
end mux_2x1;

/**   architecture body  */
architecture arch_mux_and of mux_and is

   component mux2x1 port (
         i0, i1, s: in std_logic;
         y: out std_logic);
   end component;
   // declare intermediate signals

begin
   DUT1: mux2x1 port map (‘0’, b, a, y);
   . . .
   . . .
   . . .

End arch_mux_and;

//Write the test bench for providing the stumulus

entity tb_mux_and port
end tb_mux_and;

architecture arch_tb_mux_and of tb_mux_and is
   signal a, b: std_logic := ‘0’;
   signal y: std_logic;

   component mux_and port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

begin
   DUT: mux_and port map (a, b, y);

   process
   begin
     a = ~a;
     wait for 10ns;
     b = ~b;
     wait for 12ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

1. Realize all the OR gate using 2x1 multiplexer
2. Realize all the basic gates like XOR and XNOR using 2x1 multiplexer
3. Realize all the inverter using 2x1 multiplexer

4. Exercises on decoders

To be proficient in programming, you need implement the following digital circuits

1. Implementation of 2 to 4 and 3 to 8 decoders
2. Implementation of 4 to 2 and 8 to 3 encoders
3. Realization of higher order decoders using lower order decoders
4. Realization of 8 to 3 priority encoder using 2x1 multiplexer
5. Develop a functional model of a BCD-to-seven-segment decoder for a light emitting diode (LED) display.

4.1 Implementation of 2 to 4 decoder

Write the VHDL code for the circuit contains an input bundle of two input signals and an output bundle of four decoded signals. The input bundle, i_0, i_1 represents decoder inputs. The output bus, Y0, Y1, Y2 and Y3, are used to indicate the decoded output for the two inputs. The relationship between the input and output is shown in the table below. Use a selected signal assignment statement in the solution.

```
/**
  Implementation of 2 to 4 decoder.
 **/

Declare the inputs I0, I1.
Declare the output Y0, Y1, Y2 and Y3.

I1       I0       Y3       Y2       Y1       Y0
0        0        0        0        0        1
0        1        0        0        1        0
1        0        0        1        0        0
1        1        1        0        0        0
```

Hints

```
//Write VHDL model for 2 to 4 decoder gate level model
entity dec2to4 port (
         i0, i1: in std_logic;
         y0, y1, y2, y3: out std_logic);
end dec2to4;

/**   architecture body  */
architecture arch_dec2to4 of dec2to4 is

   component nand_gate port (
         a, b: in std_logic;
         y: out std_logic);
   end component;
    // component declaration for inverter
   // declare intermediate signals

begin
   DUT1: nand_gate port map (ni1, ni0, y0);
   . . .
   . . .
   . . .
```

```
End arch_dec2to4;

//Write the test bench for providing the stumulus

entity tb_dec2to4 port
end tb_dec2to4;

architecture arch_tb_dec2to4 of tb_dec2to4 is
   signal i0, i1: std_logic := ‘0’;
   signal y0, y1, y2, y3: std_logic;

   component dec2x4 port (
         i0, i1: in std_logic;
         y0, y1, y2, y3: out std_logic);
   end component;

begin
   DUT: dec2to4 port map (i0,i1, y0, y1, y2, y3);

   process
   begin
     i0 = ~i0;
     wait for 10ns;
     i1 = ~i1;
     wait for 15ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

1. Implement gate level model for 2 to 4 decoder. Plot the waveforms
2. Implement behavioral model for 2 to 4 decoder. Plot the waveforms

4.2 Implementation of 3 to 8 decoder

Build behavioral model to function as 3 to 8 decoder.

Hints

```
/** Behavioral model implementation of 3 to 8 decoder    **/
Declare the inputs I0, I1, I2.
Declare the output Y0, Y1, Y2, Y3, Y4, Y5, Y6 and Y7.

//Write VHDL model for 2 to 4 decoder gate level model
entity dec3to8 port (
         i0, i1, i2: in std_logic;
         y0, y1, y2, y4, y5, y6, y7: out std_logic);
end dec3to8;

/**   architecture body  */
architecture arch_dec3to8 of dec3to8 is

   // declare intermediate signals
```

```
begin
   process(i0, i1, i2)
   begin
      { y0, y1, y2, y4, y5, y6, y7} = “00000000”;

      case {i2, i1, i0} is
        when “000” => Y0 <= ‘1’;
        when “001” => Y0 <= ‘1’;
        when “010” => Y0 <= ‘1’;
        . . .
        . . .
        . . .
      End case
   End process;

End arch_dec3to8;

//Write the test bench for providing the stumulus

entity tb_dec3to8 port
end tb_dec3to8;

architecture arch_tb_dec3to8 of tb_dec3to8 is
   signal i0, i1, i2: std_logic := ‘0’;
   signal y0, y1, y2, y3, y4, y5, y6, y7: std_logic;

   component dec3x8 port (
         i0, i1, i2: in std_logic;
         y0, y1, y2, y3, y4, y5, y6, y7: out std_logic);
   end component;

begin
   DUT: dec3to8 port map (i0, i1, i2, y0, y1, y2, y3, y4, y5, y6, y7);

   process
   begin
     i0 = ~i0;
     wait for 10ns;
     i1 = ~i1;
     wait for 15ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

1. Construct a 4-to-16 line decoder with two 3-to-8 line decoders having active LOW ENABLE inputs
2. Implement the three-variable Boolean function $F = \bar{a}c + a\bar{b}c + a\bar{b}c$ using (i) an 8-to-1 multiplexer and (ii) a 4-to-1 multiplexer.

4.3 Realization of higher order multiplexers using lower order multiplexers

Construct a 4-to-16 line decoder with two 3-to-8 line decoders having active LOW ENABLE inputs

Hints

```
/**
 Realize the lower order multiplexers for design of higher order multiplexers   **/

In work library simulate the lower order multiplexer

For the realized higher order multiplexer, instance the component in the declaration part
of the architecture

By using positional or name mapping instance the component with the signals

Design a test bench

Write the logic for generating stimulus for the input signals

Plot the wave form for all possible select lines

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

Construct a 3-to-8 line decoder with two 2-to-4 line decoders having active LOW ENABLE inputs

4.4 Realization of basic gates using 2x1 multiplexer

Realize all the basic gates like AND, OR, XOR, XNOR and inverter using 2x1 multiplexer

```
/**   Realize the multiplexers for function as basic logic gates **/

In work library simulate the 2x1 multiplexer

For  the  realized  basic  gates  using  2x1  multiplexer,  instance  the  component  in  the
declaration part of the architecture

By using positional or name mapping instance the component with the signals

Design a test bench

Write the logic for generating stimulus for the input signals

Plot the wave form for all possible select lines

Synthesize the design

Elaborate the design and dump the bit file into FPGA
```

Try

Realization of BCD-to-seven-segment decoder for a light emitting diode (LED) display

Develop a functional model of a BCD-to-seven-segment decoder for a light emitting diode (LED) display. The decoder has a 4-bit input that encodes a numeric digit between 0 and 9. There are seven outputs indexed from 'a' to 'g', corresponding to the seven segments of the LED display as shown in the margin. An output bit being '1' causes the corresponding segment to illuminate. For each input digit, the decoder activates the appropriate combination of segment outputs to form the displayed representation of the digit.

Hint:
For example, for the input "0010", which encodes the digit 2, the output is "1101101". Your model should use a selected signal assignment statement to describe the decoder function in truth-table form

5. Exercises on encoders and priority encoders

To be proficient in programming, you need implement the following digital circuits

1. Implementation of 4 to 2 encoders
2. Implementation of 8 to 3 encoders
3. Build 8 to 3 priority encoder
4. Realization of 8 to 3 priority encoder using 2x1 multiplexer

5.1 Implementation of 4 to 2 encoder

An encoder is a digital circuit that converts a set of binary inputs into a unique binary code. The binary code represents the position of the input and is used to identify the specific input that is active. Encoders are commonly used in digital systems to convert a parallel set of inputs into a serial code.

The 4 to 2 Encoder consists of four inputs Y3, Y2, Y1 & Y0, and two outputs A1 & A0. At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output. The figure below shows the logic symbol of the 4 to 2 encoder.

```
/**   Implementation of 4 to 2 decoder **/

Declare the inputs I0, I1, I2, I3.
Declare the output Y0 and Y1

I3      I2      I1      I0      Y1      Y0
0       0       0       1       0       0
0       0       1       0       0       1
0       1       0       0       1       0
1       0       0       0       1       1
```

Hints

```
//Write VHDL model for 2 to 4 decoder gate level model
entity enc4to2 port (
    i : in STD_LOGIC_VECTOR(3 downto 0);
    y : out STD_LOGIC_VECTOR(1 downto 0) );
end enc4to2;
```

```
architecture arch_enc4to2 of enc4to2 is
begin

    process(i)
    begin
        if (i="1000") then y <= "00";
        elsif (i="0100") then y <= "01";
        elsif (i="0010") then y <= "10";
        elsif (i="0001") then y <= "11";
        else  y <= "ZZ";
        end if;
    end process;

End arch_enc4to2;

//Write the test bench for providing the stumulus

entity tb_enc4to2 port
end tb_enc4to2;

architecture arch_tb_enc4to2 of tb_enc4to2 is
   signal i: in STD_LOGIC_VECTOR(3 downto 0) := “0000”;
   signal y: out STD_LOGIC_VECTOR(1 downto 0);

   component enc4to2 port (
         i: in STD_LOGIC_VECTOR(3 downto 0);
         y: out STD_LOGIC_VECTOR(1 downto 0));
   end component;

begin
   DUT: enc4to2 port map (i, y);

   process
   begin
     i = i + ‘1’;
     wait for 10ns;
   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

1. Modify VHDL behavioral model with gate level model for 4 to 2 encoder. Plot the waveforms

2. Decimal to BCD Encoder

   The decimal-to-binary encoder usually consists of 10 input lines and 4 output lines. Each input line corresponds to each decimal digit and 4 outputs correspond to the BCD code. This encoder accepts the decoded decimal data as an input and encodes it to the BCD output which is available on the output lines.

3. Octal to Binary Encoder (8 to 3 Encoder)

   The 8 to 3 Encoder or octal to Binary encoder consists of 8 inputs: Y7 to Y0 and 3 outputs: A2, A1 & A0. Each input line corresponds to each octal digit and three outputs generate corresponding binary code.

5.2 Implementation of 8 to 3 priority encoder

A 8 to 3 priority encoder has eight inputs Y7, Y6, Y5, Y4, Y3, Y2, Y1 & Y0 and two outputs A2, A1 and A0. Here, the input, Y7 has the highest priority, whereas the input, Y0 has the lowest priority. In this case, even if more than one input is '1' at the same time, the output will be the binary code corresponding to the input, which is having higher priority.

We considered one more output, V in order to know, whether the code available at outputs is valid or not.

- If at least one input of the encoder is '1', then the code available at outputs is a valid one. In this case, the output, V will be equal to 1.
- If all the inputs of encoder are '0', then the code available at outputs is not a valid one. In this case, the output, V will be equal to 0.

The Truth table of 4 to 2 priority encoder is shown below.

```
/**   Implementation of 4 to 2 decoder **/

Declare the inputs I0, I1, I2, I3.
Declare the output Y0 and Y1

I3      I2      I1      I0      Y1      Y0
0       0       0       1       0       0
0       0       1       0       0       1
0       1       0       0       1       0
1       0       0       0       1       1
```

Hints

```
//Write VHDL model for 2 to 4 decoder gate level model
entity penc8to3 port (
    i : in STD_LOGIC_VECTOR(7 downto 0);
    y : out STD_LOGIC_VECTOR(2 downto 0) );
end enc4to2;

architecture arch_penc8to3 of penc8to3 is
begin

   y <= "111" when i(7)='1' else
        "110" when i(6)='1' else
        "101" when i(5)='1' else
        "100" when i(4)='1' else
        "011" when i(3)='1' else
        "010" when i(2)='1' else
        "001" when i(1)='1' else
        "000";

End arch_penc8to3;

//Write the test bench for providing the stumulus

entity tb_penc8to3 port
end tb_penc8to3;

architecture arch_tb_penc8to3 of tb_penc8to3 is
   signal i : in STD_LOGIC_VECTOR(7 downto 0) := “00000000”;
   signal y : out STD_LOGIC_VECTOR(2 downto 0);
   component penc8to3 port (
         i : in STD_LOGIC_VECTOR(7 downto 0);
         y : out STD_LOGIC_VECTOR(2 downto 0));
   end component;

begin
   DUT: penc8to3 port map (i, y);

   process
   begin
     i = i + ‘1’;
     wait for 10ns;
   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

Try

1. Modify VHDL behavioral model with gate level model for 8 to 3 encoder. Plot the waveforms.
2. Realize 8 to 3 priority encoder using 2x1 mux and implement with VHDL gate level model

6. Exercises on Adders and Subtractors

To be proficient in programming, you need implement the following digital circuits

1 Implementation of half adder

2 Implementation of full adder

3 Realization of full adder using half adder

4 Design and implement 4-bit ripple carry adder

5 Implementation of half subtractor

6 Implementation of full subtractor

7 Realization of full subtractor using half subtractor

8 Realization of full subtractor using full adder

6.1 Implementation of half adder

Design a gate level circuit for half adder and verify for the following truth table and write a test bench for verifying the functionality of the ha1f adder

A half adder has two inputs for the two bits to be added and two outputs one from the sum ‘ S’ and other from the carry ’ c’ into the higher adder position. Above circuit is called as a carry signal from the addition of the less significant bits sum from the X-OR Gate the carry out from the AND gate.

Consider the inputs are A, B and outputs are Sum, Cout

A	B	Sum	Cout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Hints

The *code pattern* for implementing half adder:

```
// Declare the port signals
Input a, b;
Output sum, cout

//Write VHDL model for half adder in behavioral flow model
entity ha_df port (
         a, b: in std_logic;
         sum, cout: out std_logic);
end ha_df;

/**   architecture body  */
architecture arch_ha_df of ha_df is

begin
   process(a, b)
   begin

      case {a, b} is
        when “00” => sum <= ‘0’; cout <= ‘0’;
        when “01” => sum <= ‘1’; cout <= ‘0’;
        when “10” => sum <= ‘1’; cout <= ‘0’;
        . . .
        . . .
        . . .
      End case
   End process;

End arch_ha_df;

//Write the test bench for providing the stumulus

entity tb_ha_df port
end tb_ha_df;

architecture arch_tb_ha_df of tb_ha_df is
   signal a, b: std_logic := ‘0’;
   signal sum, cout: std_logic;

   component ha_df port (
         a, b: in std_logic;
         sum, cout: out std_logic);
   end component;

begin
   DUT: ha_df port map (a, b, sum, cout);

   process
   begin
     a = ~a;
     wait for 10ns;
     b = ~b;
     wait for 15ns;

   end process
```

```
End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

6.2 Implementation of full adder

Design a gate level circuit for full adder and verify for the following truth table and write a test bench for verifying the functionality of the full adder

Consider the inputs are A, B, C and outputs are Sum, Cout

A	B	C	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Hints

The pseudo *code* for printing full adder:

```
// Declare the port signals
Input a, b, c;
Output sum, cout
Work lib should consist of xor, and, or gates modules

//Write VHDL model for half adder in behavioral flow model
entity fa_gl port (
         a, b, cin: in std_logic;
         sum, cout: out std_logic);
end fa_gl;

/**   architecture body  */
architecture arch_fa_gl of ga_gl is
   component xor_gate port (
         a, b: in std_logic;
         y: out std_logic);
   end component;

    component declaration for or_gate, and_gate
begin

    u0: xor_gate port map(a, b, x1);
    u1: xor_gate port map(x1, cin, sum);
    . . .
    . . .

End arch_fa_gl;

//Write the test bench for providing the stumulus
```

```
entity tb_fa_gl port
end tb_fa_gl;

architecture arch_tb_fa_gl of tb_fa_gl is
   signal a, b, cin: std_logic := ‘0’;
   signal sum, cout: std_logic;

   component fa_gl port (
         a, b, cin: in std_logic;
         sum, cout: out std_logic);
   end component;

begin
   DUT: fa_gl port map (a, b, cin, sum, cout);

   process
   begin
     a = ~a;
     wait for 5ns;
     b = ~b;
     wait for 10ns;
     cin = ~cin;
     wait for 15ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

6.3 Realization of full adder using half adders

Design a gate level circuit for full adder using half adder and verify for the functionality and write a test bench for verifying the functionality of the full adder

Hints

The pseudo *code* for printing full adder:

```
// Declare the port signals
Input a, b, c;
Output sum, cout

Work lib should consist of half adder, or_gate modules

entity fa_ha port (
         a, b, cin: in std_logic;
         sum, cout: out std_logic);
end fa_ha;

/**   architecture body  */
architecture arch_fa_ha of ga_ha is
   component ha_df port (
         a, b: in std_logic;
         y: out std_logic);
   end component;
   component declaration for or_gate, and_gate
begin

    u0: xor_gate port map(a, b, x1);
    u1: xor_gate port map(x1, cin, sum);
    . . .
    . . .

End arch_fa_gl;

//Write the test bench for providing the stumulus

entity tb_fa_gl port
end tb_fa_gl;

architecture arch_tb_fa_gl of tb_fa_gl is
   signal a, b, cin: std_logic := ‘0’;
   signal sum, cout: std_logic;

   component fa_gl port (
         a, b, cin: in std_logic;
         sum, cout: out std_logic);
   end component;

begin
   DUT: fa_gl port map (a, b, cin, sum, cout);

   process
   begin
     a = ~a;
     wait for 5ns;
     b = ~b;
     wait for 10ns;
     cin = ~cin;
     wait for 15ns;

   end process

End architecture

Plot the wave form for all possible select lines
Synthesize the design
Elaborate the design and dump the bit file into FPGA
```

6.4 Design and implement 4-bit ripple carry adder

Design a gate level circuit for ripple carry adder using full adder and verify for the functionality and write a test bench for verifying the functionality of the ripple carry adder

Hints

The pseudo *code* for printing full adder:

```
// Declare the port signals
Input a, b;      //vector of 4-bit size
Input cin;
Output sum;      //vector of 4-bit size
Output cout;
```

```
// Declare xor gate, and gate
Component declaration for full adder

// instance xor gate, and gate
Port map for full adder
```

6.5 Implementation of half subtractor

Design a gate level circuit for half subtractor and verify for the following truth table and write a test bench for verifying the functionality of the ha1f subtractor

```
Consider the inputs are A, B and outputs are Diff, Bout

A        B        Diff     Bout
0        0        0        0
0        1        1        1
1        0        1        0
1        1        0        0
```

Hints

The *code pattern* for printing half subtractor:

```
// Declare the port signals
Input a, b;
Output diff, bout

// Declare xor gate, and gate
Component declaration for xor gate, and gate

// instance xor gate, and gate
Port map for xor gate
Port map for and gate
```

6.6 Implementation of full subtractor

Design a gate level circuit for full subtractor and verify for the following truth table and write a test bench for verifying the functionality of the full subtractor

```
Consider the inputs are A, B, Bin and outputs are Diff, Bout

A        B        Bin      Sum      Bout
0        0        0        0        0
0        0        1        1        1
0        1        0        1        1
0        1        1        0        1
1        0        0        1        0
1        0        1        0        0
1        1        0        0        0
1        1        1        1        1
```

Hints

The pseudo *code* for printing full subtractor:

```
// Declare the port signals
Input a, b, bin;
```

```
Output diff, bout

// Declare xor gate, and gate
Component declaration for xor gate, and gate

// instance xor gate, and gate
Port map for xor gate
Port map for and gate
```

6.7 Realization of full subtractor using half subtractor

Design a gate level circuit for full subtractor using half subtractor and verify for the functionality and write a test bench for verifying the functionality of the full subtractor

Hints

The pseudo *code* for printing full subtractor:

```
// Declare the port signals
Input a, b, bin;
Output diff, bout

// Declare xor gate, and gate
Component declaration for half subtractor, and gate, or gate

// instance xor gate, and gate
Port map for half subtractor
Port map for and gate
Port map for or gate
```

Try

Design a gate level circuit for ripple carry adder using full adder and verify for the functionality and write a test bench for verifying the functionality of the ha1f adder

Hints

The pseudo *code* for printing full adder:

```
// Declare the port signals
Input a, b;      //vector of 4-bit size
Input cin;
Output sum;       //vector of 4-bit size
Output cout;

// Declare xor gate, and gate
Component declaration for full adder

// instance xor gate, and gate
Port map for full adder
```

7. Exercises on barrel shifter and ALU

To be proficient in programming, you need implement the following digital circuits

1 Design a 4- bit barrel shifter with four select lines for supporting 16- functionalities

2 Implementation 8-bit ALU to perform the arithmetic, logical operations

7.1. 4- bit barrel shifter

A barrel shifter is a digital circuit that can shift a data word by a specified number of bits without use of any sequential logic, only pure combinational logic. There are 3 type of bitwise shift operation: logical shift, arithmetic shift, and circular shift (rotate). The data can be shifted to the left as well as to the right
circular shift.

Hints

Consider the port specification as
Input datain   // 8 bit wide
Input sel   // 3 bit wide to support 16 operations
Input dataout   // 8 bit wide

The circuit allows rotating the input data word right, where the amount of rotating is selected by the control inputs. The circuit can design by three stages of 2:1 multiplexer.

When all multiplexer select inputs are active (low), the input data passes straight through the cascade of the multiplexers and the output data (q_7.....q_0) is equal to the input data (d_7..... d_0). When S_2control signal is selected, the first stage of multiplexers performs a rotate-right by one bit operation, due to their inter-connection to the next lower input.

The second stage of multiplexers performs a rotate-right by two bits when S_1 control signal is selected.
Here the corresponding multiplexer inputs are connected to their second next-lower input.

Finally, the third stage of multiplexers performs a rotate-right by four bits, when S_0 control signal is selected. The design uses a case statement to exhaustively list all combinations of the amt signal and the corresponding rotated results.

Try:

1. Modify the barrel shifter by changing the number of select lines to 4 bit wide to support 16 different functionalities
2. Modify the barrel shifter by changing the number of data lines to 16 bit wide to support 16 different functionalities

7.2. 8- bit ALU

Arithmetic Logic Unit (ALU) is one of the most important digital logic components in CPUs. It normally executes arithmetic operations such as addition, subtraction, multiplication, division, etc. and logic operations such as and, or, xor, xnor, nand, nor, not, buffer, rotate and shift operations.

Hints

Consider the port specification as
Input A, B   // 8 bit wide
Input sel   // 4 bit wide to support 16 operations
Input dataout   // 8 bit wide

// The logic and arithmetic operations being implemented in the ALU are as follows:

1. Arithmetic Addition:	ALU_Out = A + B;
2. Arithmetic Subtraction	ALU_Out = A - B;
3. Arithmetic Multiplication	ALU_Out = A * B;
4. Arithmetic Division	ALU_Out = A / B;
5. Logical Shift Left	ALU_Out = A logical shifted left by 1;
6. Logical Shift Right	ALU_Out = A logical shifted right by 1;
7. Rotate Left	ALU_Out = A rotated left by 1;
8. Rotate Right	ALU_Out = A rotated right by 1;
9. Logical AND	ALU_Out = A AND B;
10. Logical OR	ALU_Out = A OR B;
11. Logical XOR	ALU_Out = A XOR B;
12. Logical NOR	ALU_Out = A NOR B;
13. Logical NAND	ALU_Out = A NAND B;
14. Logical XNOR	ALU_Out = A XNOR B;
15. Greater comparison	ALU_Out = 1 if A > B else 0;
16. Equal comparison	

ALU_Out = 1 A = B else 0;

The second stage of multiplexers performs a rotate-right by two bits when S_1 control signal is selected. Here the corresponding multiplexer inputs are connected to their second next-lower input.

Finally, the third stage of multiplexers performs a rotate-right by four bits, when S_0 control signal is selected. The design uses a case statement to exhaustively list all combinations of the amt signal and the corresponding rotated results.

Try:

1. Modify the ALU by changing the number of select lines to 4 bit wide to support 16 different functionalities
2. Modify the ALU by changing the number of data lines to 16 bit wide to support 16 different functionalities

8. Exercises on Latches and Flip-flops

To be proficient in programming, you need implement the following sequential logic circuits

1 Implementation of SR latch, JK latch, D latch and T latch

2 Implementation of JK flip-flop, D flip-flop and T flip-flop

3 Realization of D flip-flop using D latch

4 Realization of D flip-flop using JK flip-flop

5 Realization of T flip-flop using JK flip-flop

6 Realization of T flip-flop using D flip-flop

8.1 SR latch, JK latch, D latch and T latch

Construct an SR latch using NOR gates. Verify its operation and demonstrate the circuit.
Write an entity declaration for a positive level-triggered SR-latch with asynchronous active-low preset and clear inputs, and Q and outputs. Include concurrent assertion statements and passive processes as necessary in the entity declaration to verify that

- The preset and clear inputs are not activated simultaneously,
- The setup time of 6 ns from the J and K inputs to the rising clock edge is observed,
- The hold time of 2 ns for the J and K inputs after the rising clock edge is observed and
- The minimum pulse width of 5 ns on each of the clock, preset and clear inputs is observed.

Write a gate level architecture body for the SR latch and a test bench that exercises the statements in the entity declaration.

Hints

```
// Declare port signal

Input rst_l, clk;
Input s, r;

Output q, qb;

// Component declaration of NAND gates
Component NAND_gate (input a, b; output y);

// component instance in the architecture body of VHDL program
NAND_gate port map(s, qb, q);
NAND_gate port map(r, q, qb);

// Test bench for SR latch
Design a test bench to provide the stimulus for the inputs s, r
Generate a clk signal with 5MHz frequency
Generate the reset logic to reset the latch at initial
```

Try

1. Construct SR latch using NAND gates. Verify its operation and demonstrate the circuit

2. Construct JK latch using NAND gates. Verify its operation and demonstrate the circuit

3. Construct D latch using NAND gates. Verify its operation and demonstrate the circuit

4. Construct T latch using NAND gates. Verify its operation and demonstrate the circuit

8.2 JK flip-flop, D flip-flop and T flip-flop

Construct flip-flops using latches and verify its operation and demonstrate the circuit.

Write an entity declaration for a positive edge-triggered JK-flipflop with asynchronous active-low preset and clear inputs, and Q and outputs. Include concurrent assertion statements and passive processes as necessary in the entity declaration to verify that

- The preset and clear inputs are not activated simultaneously,
- The setup time of 6 ns from the J and K inputs to the rising clock edge is observed,
- The hold time of 2 ns for the J and K inputs after the rising clock edge is observed and
- The minimum pulse width of 5 ns on each of the clock, preset and clear inputs is observed.

Write a structural architecture body for the flipflop and a test bench that exercises the statements in the entity declaration.

Hints

The pseudo code for SR latch

```
// Declare port signal

Input rst_l, clk;
Input s, r;

Output q, qb;

// Component declaration of NAND gates
Component NAND_gate (input a, b; output y);

// component instance in the architecture body of VHDL program
NAND_gate port map(s, qb, q);
NAND_gate port map(r, q, qb);

// Test bench for SR latch
Design a test bench to provide the stimulus for the inputs s, r
Generate a clk signal with 5MHz frequency
Generate the reset logic to reset the latch at initial
```

Try

Design 2-bit register

Write component instantiation statements to model the structure shown by the schematic diagram in Figure. Assume that the entity ttl_74x74 and the corresponding architecture basic have been analyzed into the library work. Figure 2 shows the 2-bit register.

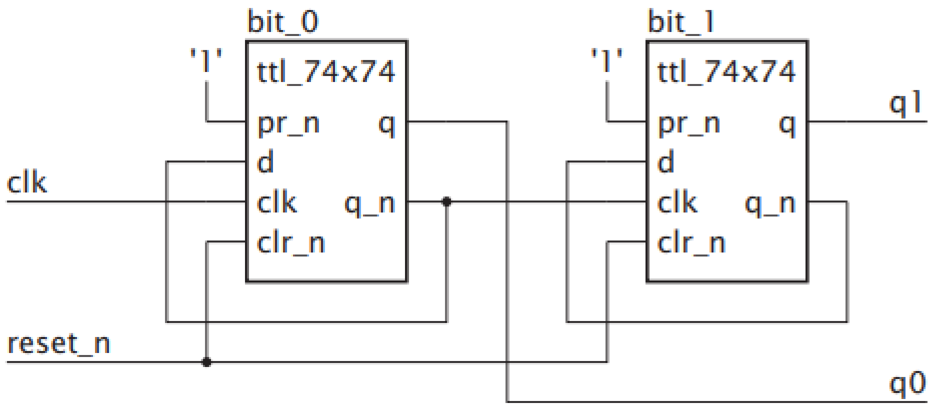


Figure 2: 2-bit register.

9. Exercises on counters and shift registers

To be proficient in programming, you need implement the following sequential logic circuits

1. 4-bit synchronous counter with synchronous reset
2. Decade counter
3. 4-bit serial in serial out shift register (SISO)

9.1: 4-bit synchronous counter with synchronous reset

Build an entity for a 4-bit counter with synchronous reset input. Include a process in the entity declaration that measures the duration of each reset pulse and reports the duration at the end of each pulse.

Hints

```
/**   Declare the port signals */
entity counter_synrst port (
         clk, rst: in std_logic;
         q: out std_logic_vector(3 downto 0));
end counter_synrst;

/**   architecturebody  */
architecture arch_counter_synrst of counter_synrst is

Begin
    Process (clk, rst)
    Begin

       If clk’event and clk = ‘1’ then
          If rst then
                 q = “0000”;
          else
                 q = q + 1;
       end if;
    end process

End arch_counter_synrst;

//Write the test bench for providing the stumulus

entity tb_counter_synrst port
end tb_counter_synrst;

architecture arch_tb_counter_synrst of tb_counter_synrst is

   signal clk, rst: std_logic := ‘0’;
   signal q: std_logic_vector(3 downto 0);

   component counter_synrst port (
         clk, rst: in std_logic;
         q: out std_logic_vector(3 downto 0));
   end component;

begin

   DUT: counter_synrst port map (gray_code, binary_code);

   Process
   begin
        clk = ~clk;
        Wait for 10 ns;
   end process;

   Process
   begin
        rst = ‘0’;
        Wait for 10 ns;
        rst = ‘1’;
        wait;
   end process;
end architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design

Synthesize the design
```

Try

1 Design 4-bit synchronous counter with asynchronous reset

2 Design 4-bit asynchronous counter with synchronous reset

3 Design 4-bit asynchronous counter with asynchronous reset

9.2 Decade counter with asynchronous reset

Construct an entity for a decade counter with asynchronous reset input. Include a process in the entity declaration that measures the duration of each reset pulse and reports the duration at the end of each pulse.

Hints

```
/**   Declare the port signals */
entity dec_counter_asynrst port (
         clk, rst: in std_logic;
         q: out std_logic_vector(3 downto 0));
end dec_counter_asynrst;

/**   architecturebody  */
architecture arch_dec_counter_asynrst of dec_counter_asynrst is

Begin
    Process (clk, rst)
    Begin

       If rst then
                q = “0000”;
       elif clk’event and clk = ‘1’ then
          if (q = “1010”) then
                q = “0000”;
          else
                 q = q + 1;
          end if;

       end if;
    end process

End arch_dec_counter_asynrst;

//Write the test bench for providing the stumulus

entity tb_dec_counter_asynrst port
end tb_dec_counter_asynrst;

architecture arch_tb_dec_counter_asynrst of tb_dec_counter_asynrst is
```

```
   signal clk, rst: std_logic := ‘0’;
   signal q: std_logic_vector(3 downto 0);

   component dec_counter_asynrst port (
         clk, rst: in std_logic;
         q: out std_logic_vector(3 downto 0));
   end component;

begin

   DUT: dec_counter_asynrst port map (clk, rst, q);

   Process
   begin
        clk = ~clk;
        Wait for 10 ns;
   end process;

   Process
   begin
        rst = ‘0’;
        Wait for 10 ns;
        rst = ‘1’;
        wait;
   end process;

end architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design

Synthesize the design
Elaborate the design and create bit file
Dump the bit file in zybo fpga
```

Try

1 Design decade synchronous counter with synchronous reset

2 Design counter to count the events from 3 to 12

9.3 4-bit serial in serial out shift register (SISO)

In digital systems it is often necessary to have circuits that can shift the bits of a vector by one or more bit positions to the left or right. Design a circuit that can shift a four-bit vector W = w3w2w1w0 one bit position to the right when a control signal Shift is equal to 1. Let the outputs of the circuit be a four-bit vector Y = y3y2y1y0 and a signal k, such that if Shift = 1 then y3 = 0, y2 = w3, y1 = w2, y0 = w1, and k = w0. If Shift = 0 then Y = W and k = 0.
Build an entity for a shift register to drive the resister serially and output the data serially. Write a test bench architecture to simulate and verify the design.

Hints

```
/**   Declare the port signals */
entity siso port (
        clk, rst: in std_logic;
        sin : in std_logic;
        q: out std_logic_vector(3 downto 0));
end siso;

/**   architecturebody  */
architecture arch_siso of siso is

Begin
    Process (clk, rst)
    Begin

       If rst then
                 q = “0000”;
       elif clk’event and clk = ‘1’ then
          q[3] = sin;
          q[2] = q[3];
          q[1] = q[2];
          q[0] = q[1];

       end if;
    end process

End arch_siso;

//Write the test bench for providing the stumulus

entity tb_siso port
end tb_siso;

architecture arch_tb_siso of tb_siso is

   signal clk, rst: std_logic := ‘0’;
   signal sin: std_logic := ‘0’;
   signal q: std_logic_vector(3 downto 0);

   component siso port (
         clk, rst: in std_logic;
         sin : in std_logic;
         q: out std_logic_vector(3 downto 0));
   end component;

begin

   DUT: siso port map (clk, rst, sin, q);

   Process
   begin
        clk = ~clk;
        Wait for 10 ns;
   end process;

   Process
   begin
        rst = ‘0’;
```

```
          Wait for 10 ns;
          rst = ‘1’;
         wait;
     end process;

     Process
     begin
          sin = ~sin;
          Wait for 25 ns;
     end process;


end architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design

Synthesize the design
Elaborate the design and create bit file
Dump the bit file in zybo FPGA
```

Try

1 Design 4-bit serial in parallel out shift register (SIPO)

2 Design 4-bit parallel in serial out shift register (PISO)

3 Design 4-bit parallel in parallel out shift register (PIPO)

10. Exercises on case study: Pseudo random generator

To be proficient in programming, you need implement the following finite state machines

1 Pseudo random number generator using LFSR

2 Pseudo random number generator for CRC logic

10.1 Pseudo random number generator using LFSR

Build Pseudo random number generator using LFSRAn LFSR is a shift register that, when clocked, advances the signal through the register from one bit to the next most-significant bit. Some of the outputs are combined in exclusive-OR configuration to form a feedback mechanism. A linear feedback shift register can be formed by performing exclusive-OR on the outputs of two or more of the flip-flops together and feeding those outputs back into the input of one of the flip-flops.

Linear feedback shift registers make extremely good pseudorandom pattern generators. When the outputs of the flip-flops are loaded with a seed value (anything except all 0s, which would cause the LFSR to produce all 0 patterns) and when the LFSR is clocked, it will generate a pseudorandom pattern of 1s and 0s. Note that the only signal necessary to generate the test patterns is the clock.

Hints

Step 1:
The linear feedback shift register is implemented as a series of Flip-Flops inside of an FPGA that are wired together as a shift register.
Several taps off of the shift register chain are used as inputs to either an XOR or XNOR gate.
The output of this gate is then used as feedback to the beginning of the shift register chain, hence the Feedback in LFSR.

Step 2:
When an LFSR is running, the pattern that is being generated by the individual Flip-Flops is pseudo-random, meaning that it's close to random. It's not completely random because from any state of the LFSR pattern and can predict the next state.

Step 3:
Longer LFSRs will take longer to run through all iterations. The longest possible number of iterations for an LFSR of N-bits is 2N-1.

Step 4:
That pattern is all 0's when using XOR gates, or all 1's when using XNOR gates as your feedback gate.

Try:

1. Pseudo random number generator for 8-bit CRC logic
2. Pseudo random number generator for 12-bit CRC logic
3. Pseudo random number generator for 16-bit CRC logic

11. Exercises on CARRY-LOOK AHEAD ADDER

11.1 Carry look ahead adder

Build 4- bit carry look ahead adder (CLA) and justify the speed of operation CLA is more than ripple carry adder

Develop a functional model of a 3-bit carry-look-ahead adder. The adder has two 3-bit data inputs, a (2 downto 0) and b(3 downto 0); a 3-bit data output, s (2 downto 0); a carry input, c_in; a carry output, c_out; a carry generate output, g; and a carry propagate output, p. The adder is described by the logic equations and associated propagation delays: where the Gi are the intermediate carry generate signals, the Pi are the intermediate carry propagate signals and the Ci are the intermediate carry signals. C–1 is c_in and C3 is cout. Your model should use the expanded equation to calculate the intermediate carries, which are then used to calculate the sums.

$$s_i = a_i \oplus b_i \oplus c_{i-1}$$

$$g_i = a_i \, . b_i$$

$$p_i = a_i + b_i$$

$$c_i = g_i + p_i \, c_{i-1}$$

Hints

```
/**   Declare the port signals */
entity cla port (
        a, b: in std_logic-vector(2 downto 0);
        sum: out std_logic_vector(2 downto 0);
        cout: out std_logic);
end cla;

/**   architecturebody  */
architecture arch_cla of cla is

Begin
    g0 = a[0] & b[0];
    p0 = a[0] | b[0];
    c1 = g0 + p0 & c0;
     . . . . . . .
     . . . . . . .
     . . . . . . .

End arch_cla;

//Write the test bench for providing the stumulus

entity tb_cla port
end tb_cla;

architecture arch_tb_cla of tb_cla is

   signal a, b: std_logic_vector(2 downto 0) := “000”;
   signal cin: std_logic := ‘0’;
   signal sum: std_logic_vector(s downto 0);

   component cla port (
         a, b: in std_logic-vector(2 downto 0);
         sum: out std_logic_vector(2 downto 0);
         cout: out std_logic);
   end component;

begin

   DUT: cla port map (clk, rst, sin, q);

   Process
   begin
        a = a + “0110”
        wait for 10 ns;
        b = b + “1010”
   end process;

end architecture

// After post simulation

Simulate the design using Xilinx software

Plot the wave forms and verify the functionality of the design
```

```
Synthesize the design
Elaborate the design and create bit file
Dump the bit file in zybo FPGA
```

Try:

1. Design and implement 4-bit carry look ahead adder
2. Design and implement 4-bit ripple carry adder

12. Exercises on VENDING MACHINE CONTROLLER

Vending-Machine Controller sells candy bars for 25 cents. The inputs are nickel_in, dime_in, and quarter_in, indicating the type of coin that was deposited, plus clock (clk) and reset (rst), to which the circuit responds with the outputs candy_out, to dispense a candy bar, plus nickel_out or dime_out, asserted when change is due. Design this circuit using the FSM approach. Also, estimate the number of flip-flops that will be required. Figure 3. Shows the block level diagram representation of vending machine controller and Figure 4 shows the state diagram of the vending machine.

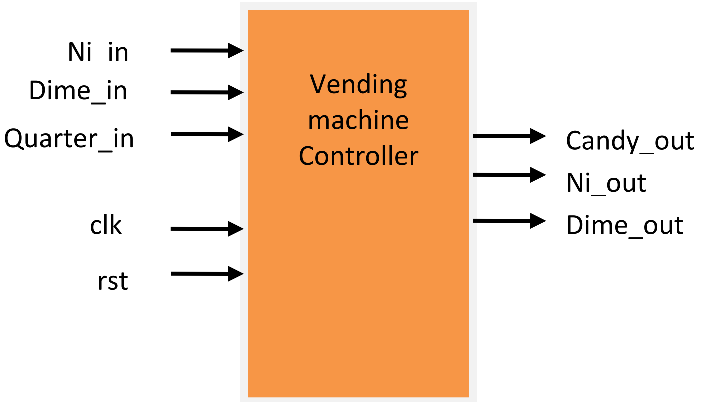


Figure 3: The vending machine controller block diagram

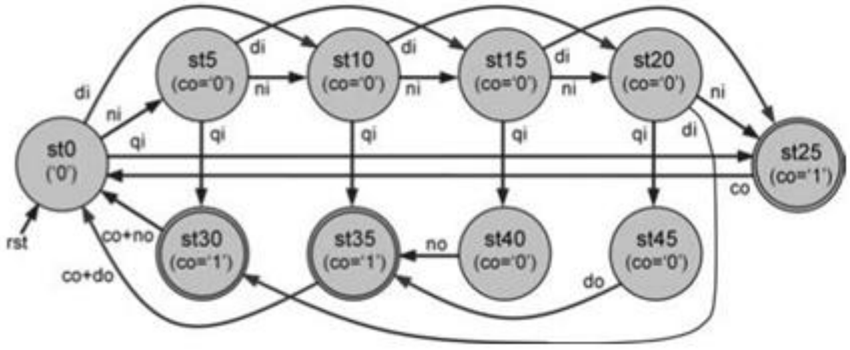


Figure 4: State diagram of the vending machine controller

```
/**   VHDL model for gray counter */

entity vend_mach is
port(   Clk, rst : in std_logic;
        Ni_in, Dime_in, Quarter_in : in std_logic;
        Candy_out, Ni_out, Dime_out : out std_logic
    );
end vend_mach;

architecture Behavioral of vend_mach is

--type of state machine and signal declaration.
type state_type is (st0, st5, st10, st15, st20, st25, st30, st35, st40, st45);
signal next_state : state_type;

begin

process(Clk, rst)
begin
    if(rising_edge(Clk)) then
        case next_s is
            when st0 =>
                if(Ni_in) then
                    next_state <= st5;
                elsif(Dime_in) then
                    next_state <= st10;
                elsif(Quarter_in) then
                    next_state <= st25;
                end if;
            when st5 =>
                 . . .
                 . . .
                 . . .
        end case;
    end if;
end process;

end Behavioral;
```

Try

1. Implement the vending machine using Melay finite state machine

13. Exercises on Gray-Encoded Counter

Design a 0-to-8 counter with Gray-encoded outputs.

a) Draw the state transition diagram.
b) Estimate the number of flip-flops that will be needed.
c) Write the VHDL code, then compile and simulate it.
d) Check whether the number of DFFs inferred by the compiler matches your prediction.

```
/**   VHDL model for gray counter */
entity graycounter is
  generic (n: integer := 6);
  port (clk, rst, en: in std_logic;
        output: out std_logic_vector (n-1 downto 0));
end graycounter;

architecture graycounter_beh of graycounter is
```

```
  signal currstate, nextstate, hold, next_hold: std_logic_vector (n-1 downto 0);
begin

  statereg: process (clk)
  begin
    if (clk = '1' and clk'event) then
      if (rst = '1') then
        currstate <= (others =>'0');
      elsif (en = '1') then
        currstate <= nextstate;
      end if;
    end if;
  end process;

  hold <= currstate xor ('0' & hold(n-1 downto 1));
  next_hold <= std_logic_vector(unsigned(hold) + 1);
  nextstate <= next_hold xor ('0' & next_hold(n-1 downto 1));
  output <= currstate;

end graycounter_beh;

/**   VHDL test bench  */
entity testbench is
end testbench;

architecture arch of testbench is
  component graycounter is
    generic (n: integer := 6);
    port (clk, rst, en: in std_logic;
          output: out std_logic_vector (n-1 downto 0));
  end component;

  signal clk_s, rst_s, en_s: std_logic;
  signal output_s: std_logic_vector(5 downto 0);

begin
  comptotest: graycounter generic map (6) port map (clk_s, rst_s, en_s, output_s);

  clk_proc: process
  begin
    clk_s <= '1';
    wait for 10 ns;
    clk_s <= '0';
    wait for 10 ns;
  end process clk_proc;

  vector_proc: process
  begin
    rst_s <= '1';
    wait until clk_s='1' and clk_s'event;
    wait for 5 ns;
    rst_s <= '0';
    en_s <= '1';
    for index in 0 to 3 loop
      wait until clk_s='1' and clk_s'event;
    end loop;
    wait for 5 ns;
    wait;
  end process vector_proc;
end arch;
```

Try

1. Design a counter with Johnson-encoded output instead of Gray-encoded.
2. Design a counter with one-hot output instead of Gray output.
3. Modify the counter designed, such that the circuit stays in each state during T ¼ 1 s, with the output displayed on a seven segment display. Assume that the clock frequency is 50 MHz.

14. Exercises on RAM design

Write an entity declaration for a lookup table RAM modeled at an abstract level. The RAM has an address input of type look_up_index, which is an integer range from 0 to 31, and a data output of type real. Include declarations within the declarative part of the entity to define the RAM contents, initialized to numbers of your choice.

```
/**   Declare the port signals */
entity single_port_RAM is
  generic (
    addr_width : integer := 2;
    data_width : integer := 3
  );

  port(
    clk: in std_logic;
    we : in std_logic;
    addr : in std_logic_vector(addr_width-1 downto 0);
    din : in std_logic_vector(data_width-1 downto 0);
    dout : out std_logic_vector(data_width-1 downto 0)
    );
end single_port_RAM;

architecture arch of single_port_RAM is
 type ram_type is array (2**addr_width-1 downto 0) of std_logic_vector (data_width-1
downto 0);
 signal ram_single_port : ram_type;

begin
  process(clk)
  begin
    if (clk'event and clk='1') then
      if (we='1') then -- write data to address 'addr'
        --convert 'addr' type to integer from std_logic_vector
        ram_single_port(to_integer(unsigned(addr))) <= din;
      end if;
  end if;
  end process;

  -- read data from address 'addr'
  -- convert 'addr' type to integer from std_logic_vector
  dout<=ram_single_port(to_integer(unsigned(addr)));
end arch;
```

Try

1. Extend the functionality of RAM memory block by allowing write and read operations to construct RAM block of size 16 x 8 RAM.

15. Final Notes

The only way to learn programming is program, program and program on challenging problems. The problems in this tutorial are certainly NOT challenging. There are tens of thousands of challenging problems available – used in training for various programming contests (such as International Collegiate Programming Contest (ICPC), International Olympiad in Informatics (IOI)). Check out these sites:

- Cadence certifications (https://www.cadence.com/en_US/home/training/become-cadence-certified.html#dds0
- National Institute of Electronics and Information Technology (https://reg.nielitchennai.edu.in)

Student can have any one of the following certifications:

- NPTEL – Digital design
- NPTEL – HDL programming

V. TEXT BOOKS:

1 Wen-Long Chin, "Principles of Verilog Digital Design", CRC Press, 1st edition, 2022

2 Charles Roth, "Digital System Design using VHDL", Tata McGraw Hill, 2nd edition, 2012.

3 M. Morris Mano and Michael D. Ciletti, "Digital Design", Pearson Education, 6th edition, 2018.

4 John F Wakerly, "Digital Design Principles and practices", Pearson Education, 4th edition, 2008

VI. REFERENCES

1 Mohammad Karim, Xinghao Chen, "Digital Design: Basic Concepts and Principles", CRC Press, 2nd edition, 2007.

2 Samir Palnitkar, "Verilog HDL: A Guide to Digital Design and Synthesis", Pearson Education, 2nd edition, 2003.

DATA STRUCTURES LABORATORY

III Semester: Common for all branches								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC10	Core	L	T	P	C	CIA	SEE	Total
		0	0	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 45				Total Classes: 45		
Prerequisite: Programming for Problem Solving using C and Python Programming								

I. COURSE OVERVIEW:

The course covers some of the general-purpose data structures and algorithms, and software development. Topics covered include managing complexity, analysis, static data structures, dynamic data structures and hashing mechanisms. The main objective of the course is to teach the students how to select and design data structures and algorithms that are appropriate for problems that they might encounter in real life. This course reaches to student by power point presentations, lecture notes, and lab which involve the problem solving in mathematical and engineering areas.

II. COURSES OBJECTIVES:

The students will try to learn

I. The skills needed to understand and analyze performance trade-offs of different algorithms / implementations and asymptotic analysis of their running time and memory usage.

II. The basic abstract data types (ADT) and associated algorithms: stacks, queues, lists, tree, graphs, hashing and sorting, selection and searching.

III. The fundamentals of how to store, retrieve, and process data efficiently.

III. COURSE OUTCOMES:

At the end of the course students should be able to:

CO 1	Interpret the complexity of algorithm using the asymptotic notations.
CO 2	Select appropriate searching and sorting technique for a given problem.
CO 3	Construct programs on performing operations on linear and nonlinear data structures for organization of a data
CO 4	Make use of linear data structures and nonlinear data structures solving real time applications.
CO 5	Describe hashing techniques and collision resolution methods for efficiently accessing data with respect to performance.
CO 6	Compare various types of data structures; in terms of implementation, operations and performance.

IV. COURSE CONTENT:

EXERCISES FOR DATA STRUCTURES LABORATORY

Note: Students are encouraged to bring their own laptops for laboratory practice sessions.

1. Getting Started Exercises

1.1 Implicit Recursion

A specific type of recursion called **implicit recursion** occurs when a function calls itself without making an explicit recursive call. This can occur when a function calls another function, which then calls the original code once again and starts a recursive execution of the original function.

Using implicit recursion find the second-largest elements from the array.

In this case, the **find_second_largest** method calls the **find_largest()** function via implicit recursion to locate the second-largest number in a provided list of numbers. Implicit recursion can be used in this way to get the second-largest integer without having to write any more code

Input: nums = [1, 2, 3, 4, 5]

Output: 4

```
def find_largest(numbers):
     # Write code here
     …

def find_second_largest(numbers):
    # Write code here
     …

# Driver code
numbers = [1, 2, 3, 4, 5]

# Function call
second_largest = find_second_largest(numbers)
print(second_largest)
```

1.2 Towers of Hanoi

Tower of Hanoi is a mathematical puzzle where we have three rods (A, B, and C) and N disks. Initially, all the disks are stacked in decreasing value of diameter i.e., the smallest disk is placed on the top and they are on rod A. The objective of the puzzle is to move the entire stack to another rod (here considered C), obeying the following simple rules:

- Only one disk can be moved at a time.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
- No disk may be placed on top of a smaller disk.

Input: 2

Output: Disk 1 moved from A to B

Disk 2 moved from A to C
Disk 1 moved from B to C

Input: 3

Output: Disk 1 moved from A to C

Disk 2 moved from A to B
Disk 1 moved from C to B

Disk 3 moved from A to C

Disk 1 moved from B to A

Disk 2 moved from B to C

Disk 1 moved from A to C

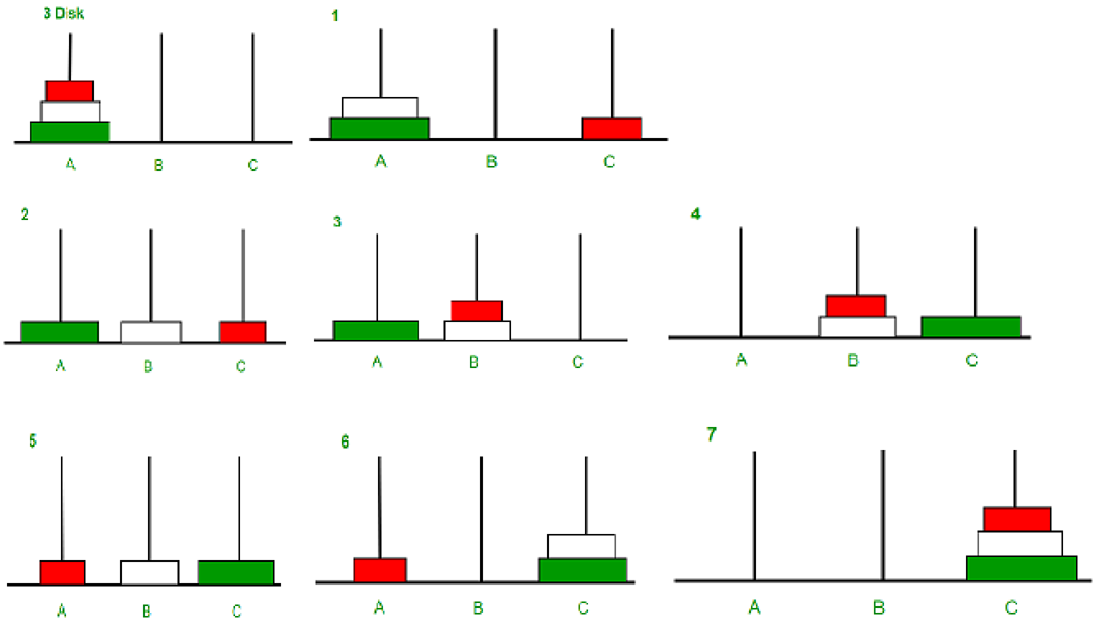
Tower of Hanoi using Recursion:

The idea is to use the helper node to reach the destination using recursion. Below is the pattern for this problem:

- Shift 'N-1' disks from 'A' to 'B', using C.
- Shift last disk from 'A' to 'C'.
- Shift 'N-1' disks from 'B' to 'C', using A.

Follow the steps below to solve the problem:

- Create a function towerOfHanoi where pass the N (current number of disk), from_rod, to_rod, aux_rod.
- Make a function call for N – 1 th disk.
- Then print the current the disk along with from_rod and to_rod
- Again make a function call for N – 1 th disk.



```
# Recursive Python function to solve Tower of Hanoi
def TowerOfHanoi(n, from_rod, to_rod, aux_rod):
    if n == 0:
        return
    # Write code here
    …

# Driver code
N = 3

# A, C, B are the name of rods
TowerOfHanoi(N, 'A', 'C', 'B')
```

1.3 Recursively Remove all Adjacent Duplicates

Given a string, recursively remove adjacent duplicate characters from the string. The output string should not have any adjacent duplicates.

Input: s = "azxxzy"

Output: "ay"

Explanation:

- First "azxxzy" is reduced to "azzy".
- The string "azzy" contains duplicates
- So it is further reduced to "ay"

Input: "caaabbbaacdddd"

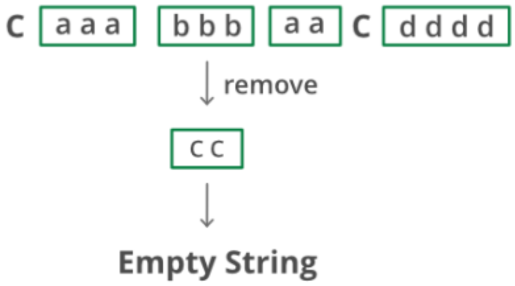
Output: Empty String

Input: "acaaabbbacdddd"

Output: "acac"

Procedure to remove duplicates:

- Start from the leftmost character and remove duplicates at left corner if there are any.
- The first character must be different from its adjacent now. Recur for string of length n-1 (string without first character).
- Let the string obtained after reducing right substring of length n-1 be rem_str. There are three possible cases
    - ➢ If first character of rem_str matches with the first character of original string, remove the first character from rem_str.
    - ➢ If remaining string becomes empty and last removed character is same as first character of original string. Return empty string.
    - ➢ Else, append the first character of the original string at the beginning of rem_str.
- Return rem_str.



```
# Program to remove all adjacent duplicates from a string

# Recursively removes adjacent duplicates from str and returns
# new string. last_removed is a pointer to last_removed character

def removeUtil(string, last_removed):

      # Write code here
       …
def remove(string):
    # Write code here
    …
# Utility functions
def toList(string):
    x = []
    for i in string:
        x.append(i)
    return x

def toString(x):
    return ''.join(x)

# Driver program
string1 = "azxxxzy"
print remove(string1)

string2 = "caaabbbaac"
print remove(string2)

string3 = "gghhg"
print remove(string3)

string4 = "aaaacddddcappp"
print remove(string4)

string5 = "aaaaaaaaaa"
print remove(string5)
```

1.4 Product of Two Numbers using Recursion

Given two numbers x and y find the product using recursion.

Input: x = 5, y = 2

Output: 10

Input: x = 100, y = 5

Output: 500

Procedure

1. If x is less than y, swap the two variables value
2. Recursively find y times the sum of x
3. If any of them become zero, return 0

```
# Find Product of two Numbers using Recursion

# recursive function to calculate multiplication of two numbers
def product( x , y ):
    # Write code here
    …
# Driver code
x = 5
y = 2
print( product(x, y))
```

1.5 Binary to Gray Code using Recursion

Given the Binary code of a number as a decimal number, we need to convert this into its equivalent Gray Code. Assume that the binary number is in the range of integers. For the larger value, we can take a binary number as string.

In gray code, only one bit is changed in 2 consecutive numbers.

Input: 1001

Output: 1101

Explanation: 1001 -> 1101 -> 1101 -> 1101

Input: 11

Output: 10

Explanation: 11 -> 10

Procedure:

The idea is to check whether the last bit and second last bit are same or not, if it is same then move ahead otherwise add 1.

Follow the steps to solve the given problem:

```
binary_to_grey(n)
if n == 0
    grey = 0;
else if last two bits are opposite to each other
    grey = 1 + 10 * binary_to_gray(n/10))
else if last two bits are same
    grey = 10 * binary_to_gray(n/10))
```

```
# Convert Binary to Gray code using recursion
# Function to change Binary to Gray using recursion
def binary_to_gray(n):
    # write code here
    …
# Driver Code
binary_number = 1011101
print(binary_to_gray(binary_number), end='')
```

1.6 Count Set-bits of a number using Recursion

Given a number N. The task is to find the number of set bits in its binary representation using recursion.

Input: 21

Output: 3

Explanation: 21 represented as 10101 in binary representation

Input: 16

Output: 1

Explanation: 16 represented as 10000 in binary representation

Procedure:

1. First, check the LSB of the number.
2. If the LSB is 1, then we add 1 to our answer and divide the number by 2.
3. If the LSB is 0, we add 0 to our answer and divide the number by 2.
4. Then we recursively follow step 1 until the number is greater than 0.

```
# Find number of set bits in a number
# Recursive function to find number of set bits in a number
def CountSetBits(n):
    # write code here
    …
# Driver code
n = 21;
# Function call
print(CountSetBits(n));
```

1.7 Fibonacci Series in Reverse Order using Recursion

Given an integer N, the task is to print the first N terms of the Fibonacci series in reverse order using Recursion.

Input: N = 5

Output: 3 2 1 1 0

Explanation: First five terms are – 0 1 1 2 3

Input: N = 10

Output: 34 21 13 8 5 3 2 1 1 0

The idea is to use recursion in a way that keeps calling the same function again till N is greater than 0 and keeps on adding the terms and after that starts printing the terms.

Follow the steps below to solve the problem:

1. Define a function fibo (int N, int a, int b) where
   i. N is the number of terms and
   ii. a and b are the initial terms with values 0 and 1.
2. If N is greater than 0, then call the function again with values N-1, b, a+b.
3. After the function call, print a as the answer.

```
# Function to print the Fibonacci series in reverse order.
def fibo(n, a, b):
    # write code here
    …
# Driver Code
N = 10
fibo(N, 0, 1)
```

1.8 Length of Longest Palindromic Sub-string using Recursion

Given a string S, the task is to find the length longest sub-string which is a palindrome.

Input: S = “aaaabbaa”

Output: 6

Explanation: Sub-string “aabbaa” is the longest palindromic sub-string.

Input: S = “banana”

Output: 5

Explanation: Sub-string “anana” is the longest palindromic sub-string.

The idea is to use recursion to break the problem into smaller sub-problems. In order to break the problem into two smaller sub-problems, compare the start and end characters of the string and recursively call the function for the middle substring.

```
# Find the length of longest palindromic sub-string using Recursion
# Function to find maximum of the two variables
def maxi(x, y):
    if x > y:
        return x
    else:
        return y
 # Function to find the longest palindromic substring: Recursion
def longestPalindromic(strn, i, j, count):
    # write code here
    …
# Function to find the longest palindromic sub-string
def longest_palindromic_substr(strn):
    # write code here
    …
strn = "aaaabbaa"
# Function Call
print(longest_palindromic_substr(strn))
```

1.9 Find the Value of a Number Raised to its Reverse

Given a number N and its reverse R. The task is to find the number obtained when the number is raised to the power of its own reverse

Input : N = 2, R = 2

Output: 4

Explanation: Number 2 raised to the power of its reverse 2 gives 4 which gives 4 as a result after performing modulo 10^9+7

Input: N = 57, R = 75

Output: 262042770

Explanation: 57^{75} modulo 10^9+7 gives us the result as 262042770

```
# Function to return ans with modulo
def PowerOfNum(N, R):
    # write code here
     …
# Driver code
N = 57
R = 75
# Function call
print(int(PowerOfNum(N, R)))
```

1.10 Mean of Array using Recursion

Find the mean of the elements of the array.

Mean = (Sum of elements of the Array) / (Total no of elements in Array)

Input: 1 2 3 4 5

Output: 3

Input: 1 2 3

Output: 2

To find the mean using recursion assume that the problem is already solved for N-1 i.e. you have to find for n

Sum of first N-1 elements = (Mean of N-1 elements) * (N-1)

Mean of N elements = (Sum of first N-1 elements + N-th elements) / (N)

```
# Program to find mean of array
# Function definition of findMean function
def findMean(A, N):
    # write code here
    …
# Driver Code
Mean = 0
```

```
A = [1, 2, 3, 4, 5]
N = len(A)
print(findMean(A, N))
```

Try:

1. Given two numbers **N** and **r**, find the value of $^{N}C_{r}$ using recursion.

$$C(n,r) = C(n-1,r-1) + C(n-1,r)$$

Input: N = 5, r = 2

Output: 10

Explanation: The value of 5C2 is 10

2. Predict the output of the following program. What does the following fun() do in general?

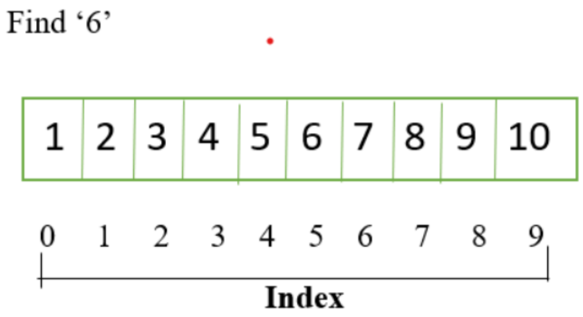
```
fp = 15
def fun(n):
    global fp
    if (n <= 2):
        fp = 1
        return 1

    t = fun(n - 1)
    f = t + fp
    fp = t
    return f

# Driver code
print(fun(5))
```

3. **Tail recursion:** Calculate factorial of a number using a Tail-Recursive function.

2. Searching

2.1 Linear / Sequential Search

Linear search is defined as the searching algorithm where the list or data set is traversed from one end to find the desired value. Given an array arr[] of n elements, write a recursive function to search a given element x in arr[].



Note : We find '6' at index '5' through linear search

Linear search procedure:

1. Start from the leftmost element of arr[] and one by one compare x with each element of arr[]
2. If x matches with an element, return the index.
3. If x doesn't match with any of the elements, return -1.

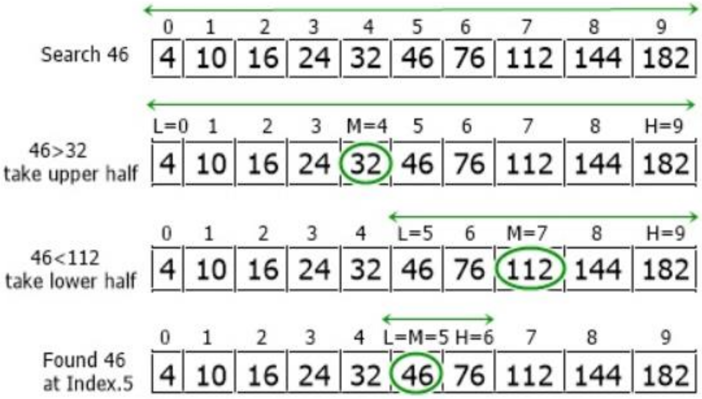
Input: arr[] = {10, 20, 80, 30, 60, 50, 110, 100, 130, 170}
x = 110;
Output: 6
Element x is present at index 6

Input: arr[] = {10, 20, 80, 30, 60, 50, 110, 100, 130, 170}
x = 175;
Output: -1
Element x is not present in arr[].

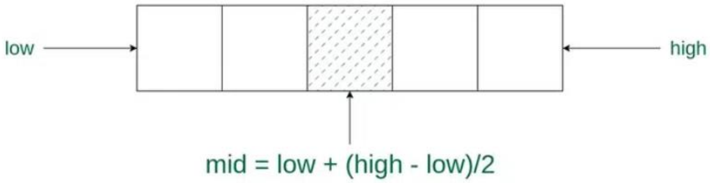
```
# Recursive linear search
def linear_search(arr, curr_index, key):
    # write code here
    …

# Driver code
arr = [10, 20, 80, 30, 60, 50, 110, 100, 130, 170]
x = 110
linear_search(arr, 0, x)
```

2.2 Binary Search

Binary Search is defined as a searching algorithm used in a sorted array by repeatedly dividing the search interval in half. The idea of binary search is to use the information that the array is sorted and reduce the time complexity to O(log N).



Conditions for Binary Search algorithm:
1. The data structure must be sorted.
2. Access to any element of the data structure takes constant time.



Binary Search Procedure:

1. Divide the search space into two halves by finding the middle index "mid".
2. Compare the middle element of the search space with the key.
3. If the key is found at middle element, the process is terminated.
4. If the key is not found at middle element, choose which half will be used as the next search space.
    a. If the key is smaller than the middle element, then the left side is used for next search.
    b. If the key is larger than the middle element, then the right side is used for next search.
5. This process is continued until the key is found or the total search space is exhausted.

Input: arr = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]
Output: target = 23
Element 23 is present at index 5

```
# Program for recursive binary search.

# Returns index of x in arr if present, else -1
def binarySearch(arr, l, r, x):
    # write code here
    …

# Driver Code
arr = [2, 3, 4, 10, 40]
x = 10
result = binarySearch(arr, 0, len(arr)-1, x)

if result != -1:
    print("Element is present at index", result)
else:
    print("Element is not present in array")
```

2.3 Uniform Binary Search

Uniform Binary Search is an optimization of Binary Search algorithm when many searches are made on same array or many arrays of same size. In normal binary search, we do arithmetic operations to find the mid points. Here we precompute mid points and fills them in lookup table. The array look-up generally works faster than arithmetic done (addition and shift) to find the mid-point.

Input: array = {1, 3, 5, 6, 7, 8, 9}, v=3

Output: Position of 3 in array = 2

Input: array = {1, 3, 5, 6, 7, 8, 9}, v=7

Output: Position of 7 in array = 5

The algorithm is very similar to Binary Search algorithm, the only difference is a lookup table is created for an array and the lookup table is used to modify the index of the pointer in the array which makes the search faster. Instead of maintaining lower and upper bound the algorithm maintains an index and the index is modified using the lookup table.

```
# Implementation of above approach

MAX_SIZE = 1000

# lookup table
lookup_table = [0] * MAX_SIZE
```

```
# create the lookup table for an array of length n
def create_table(n):
    # write code here
    …
# binary search

def binary(arr, v):
    # write code here
    …

# Driver code
arr = [1, 3, 5, 6, 7, 8, 9]
n = len(arr)

# create the lookup table
create_table(n)

# print the position of the array
print("Position of 3 in array = ", binary(arr, 3))
```

2.4 Interpolation Search

Interpolation search works better than Binary Search for a Sorted and Uniformly Distributed array. Binary search goes to the middle element to check irrespective of search-key. On the other hand, Interpolation search may go to different locations according to search-key. If the value of the search-key is close to the last element, Interpolation Search is likely to start search toward the end side. Interpolation search is more efficient than binary search when the elements in the list are uniformly distributed, while binary search is more efficient when the elements in the list are not uniformly distributed.

Interpolation search can take longer to implement than binary search, as it requires the use of additional calculations to estimate the position of the target element.

Input: arr = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Output: target = 5

```
# Interpolation search
def interpolation_search(arr, target):
    # write code here
    …

# Driver code
arr = [1, 2, 3, 4, 5, 6, 7, 8, 9]
target = 5

index = interpolation_search(arr, target)
if index == -1:
    print(f"{target} not found in the list")
else:
    print(f"{target} found at index {index}")
```

2.5 Fibonacci Search

Given a sorted array arr[] of size n and an element x to be searched in it. Return index of x if it is present in array else return -1.

Input: arr[] = {2, 3, 4, 10, 40}, x = 10
Output: 3
Element x is present at index 3.

Input: arr[] = {2, 3, 4, 10, 40}, x = 11
Output: -1
Element x is not present.

Fibonacci Search is a comparison-based technique that uses Fibonacci numbers to search an element in a sorted array.
Fibonacci Numbers are recursively defined as F(n) = F(n-1) + F(n-2), F(0) = 0, F(1) = 1. First few Fibonacci Numbers are 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …

Fibonacci Search Procedure:

Let the searched element be x. The idea is to first find the smallest Fibonacci number that is greater than or equal to the length of the given array. Let the found Fibonacci number be fib (m'th Fibonacci number). We use (m-2)'th Fibonacci number as the index (If it is a valid index). Let (m-2)'th Fibonacci Number be i, we compare arr[i] with x, if x is same, we return i. Else if x is greater, we recur for subarray after i, else we recur for subarray before i.

Let arr[0..n-1] be the input array and the element to be searched be x.

1. Find the smallest Fibonacci number greater than or equal to n. Let this number be fibM [m'th Fibonacci number]. Let the two Fibonacci numbers preceding it be fibMm1 [(m-1)'th Fibonacci Number] and fibMm2 [(m-2)'th Fibonacci Number].
2. While the array has elements to be inspected:
   i. Compare x with the last element of the range covered by fibMm2

   ii. If x matches, return index

   iii. Else If x is less than the element, move the three Fibonacci variables two Fibonacci down, indicating elimination of approximately rear two-third of the remaining array.

   iv. Else x is greater than the element, move the three Fibonacci variables one Fibonacci down. Reset offset to index. Together these indicate the elimination of approximately front one-third of the remaining array.
3. Since there might be a single element remaining for comparison, check if fibMm1 is 1. If Yes, compare x with that remaining element. If match, return index.

```
# Fibonacci search
from bisect import bisect_left

# Returns index of x if present, else returns -1
 def fibMonaccianSearch(arr, x, n):
    # write code here
    …

# Driver Code
arr = [10, 22, 35, 40, 45, 50, 80, 82, 85, 90, 100,235]
n = len(arr)
```

```
x = 235
ind = fibMonaccianSearch(arr, x, n)
if ind>=0:
  print("Found at index:",ind)
else:
  print(x,"isn't present in the array");
```

3. Sorting

3.1 Bubble Sort

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order. This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

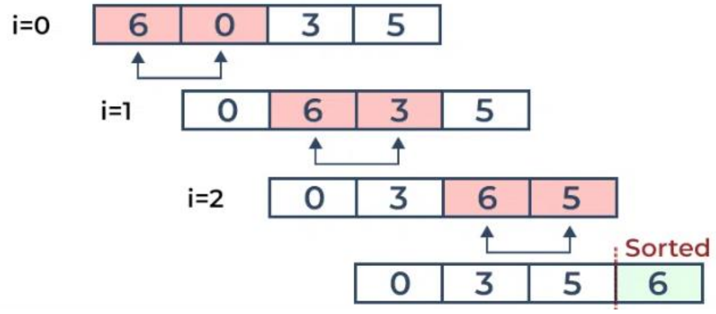
Bubble Sort Procedure:

1. Traverse from left and compare adjacent elements and the higher one is placed at right side.
2. In this way, the largest element is moved to the rightmost end at first.
3. This process is then continued to find the second largest and place it and so on until the data is sorted.

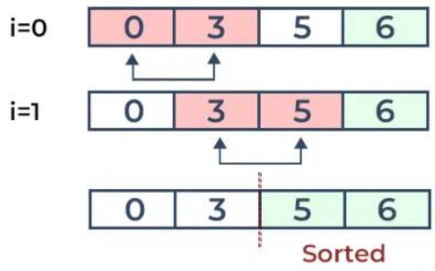
Input: arr = [6, 3, 0, 5]

Output:

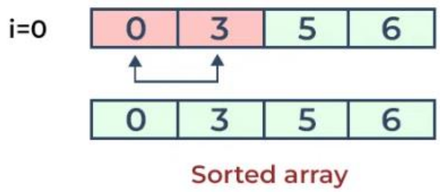
First Pass:



Second Pass:



Third Pass:



```
# Implementation of Bubble Sort

def bubbleSort(arr):
       # write code here
       …

# Driver code to test above
arr = [64, 34, 25, 12, 22, 11, 90]

bubbleSort(arr)

print("Sorted array:")
for i in range(len(arr)):
    print("%d" % arr[i], end=" ")
```

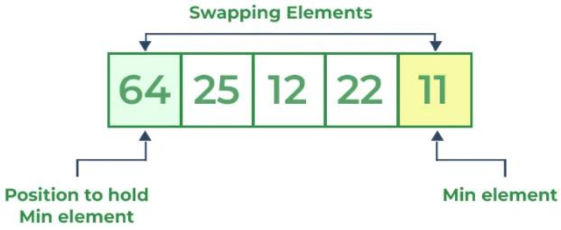
3.2 Selection Sort

Selection sort is a simple and efficient sorting algorithm that works by repeatedly selecting the smallest (or largest) element from the unsorted portion of the list and moving it to the sorted portion of the list. The algorithm repeatedly selects the smallest (or largest) element from the unsorted portion of the list and swaps it with the first element of the unsorted part. This process is repeated for the remaining unsorted portion until the entire list is sorted.

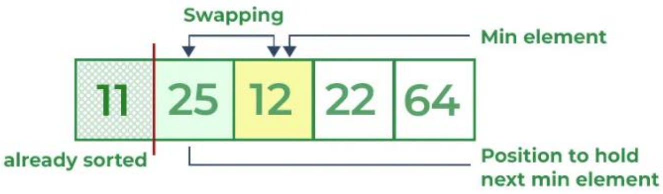
Input: arr = [64, 25, 12, 22, 11]

Output: arr = [11, 12, 22, 25, 64]

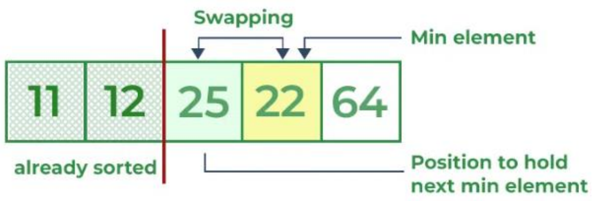
First Pass: For the first position in the sorted array, the whole array is traversed from index 0 to 4 sequentially. The first position where 64 is stored presently, after traversing whole array it is clear that 11 is the lowest value. Thus, replace 64 with 11. After one iteration 11, which happens to be the least value in the array, tends to appear in the first position of the sorted list.



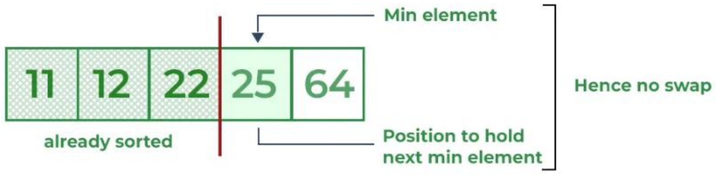
Second Pass: For the second position, where 25 is present, again traverse the rest of the array in a sequential manner. After traversing, we found that 12 is the second lowest value in the array and it should appear at the second place in the array, thus swap these values.



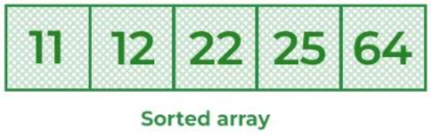
Third Pass: Now, for third place, where 25 is present again traverse the rest of the array and find the third least value present in the array. While traversing, 22 came out to be the third least value and it should appear at the third place in the array, thus swap 22 with element present at third position.



Fourth Pass: Similarly, for fourth position traverse the rest of the array and find the fourth least element in the array. As 25 is the 4th lowest value hence, it will place at the fourth position.



Fifth Pass: At last the largest value present in the array automatically get placed at the last position in the array. The resulted array is the sorted array.



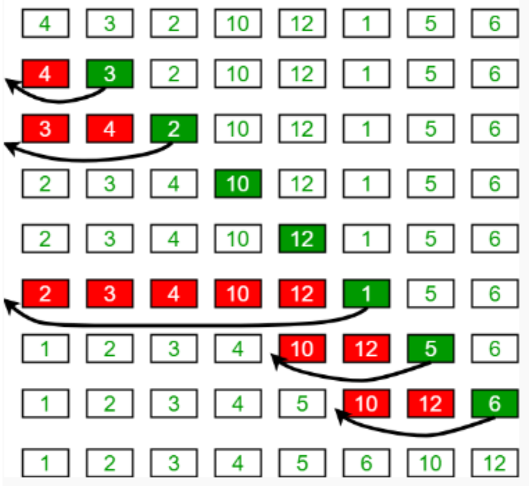
```
# Implementation of selection sort
import sys
A = [64, 25, 12, 22, 11]

# Traverse through all array elements
for i in range(len(A)):
      # write code here
       …
# Driver code
print ("Sorted array")
for i in range(len(A)):
    print("%d" %A[i],end=" , ")
```

3.3 Insertion Sort

Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.

Insertion Sort Procedure:

1. To sort an array of size N in ascending order iterate over the array and compare the current element (key) to its predecessor, if the key element is smaller than its predecessor, compare it to the elements before.
2. Move the greater elements one position up to make space for the swapped element.



```
4  3  2  10 12 1  5  6
4  3  2  10 12 1  5  6
3  4  2  10 12 1  5  6
2  3  4  10 12 1  5  6
2  3  4  10 12 1  5  6
2  3  4  10 12 1  5  6
1  2  3  4  10 12 5  6
1  2  3  4  5  10 12 6
1  2  3  4  5  6  10 12
```

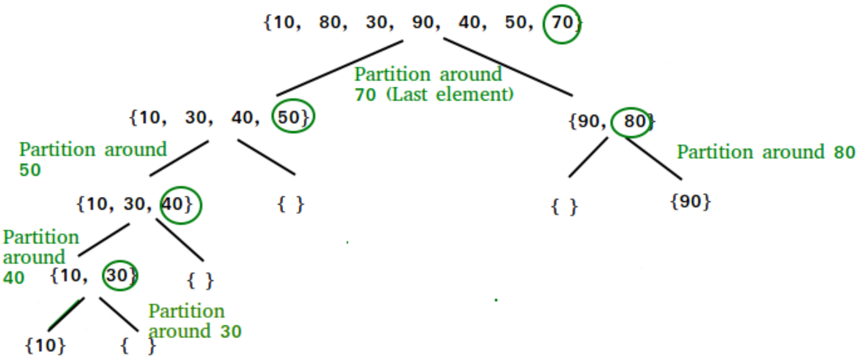
Input: arr = [4, 3, 2, 10, 12, 1, 5, 6]
Output: arr = [1, 2, 3, 4, 5, 6, 10, 12]

```
# Implementation of Insertion Sort

# Function to do insertion sort
def insertionSort(arr):
    # write code here
    …

# Driver code
arr = [12, 11, 13, 5, 6]
insertionSort(arr)
for i in range(len(arr)):
    print ("% d" % arr[i])
```

4. Divide and Conquer

4.1 Quick Sort

QuickSort is a sorting algorithm based on the Divide and Conquer algorithm that picks an element as a pivot and partitions the given array around the picked pivot by placing the pivot in its correct position in the sorted array. The key process in quickSort is a partition(). The target of partitions is to place the pivot (any element can be chosen to be a pivot) at its correct position in the sorted array and put all smaller elements to the left of the pivot, and all greater elements to the right of the pivot. Partition is done recursively on each side of the pivot after the pivot is placed in its correct position and this finally sorts the array.



The quick sort method can be summarized in three steps:
1. **Pick:** Select a pivot element.
2. **Divide:** Split the problem set, move smaller parts to the left of the pivot and larger items to the right.
3. **Repeat and combine:** Repeat the steps and combine the arrays that have previously been sorted.

Algorithm for Quick Sort Function:

```
//start –> Starting index,  end --> Ending index
Quicksort(array, start, end)
{
    if (start < end)
    {
        pIndex = Partition(A, start, end)
        Quicksort(A,start,pIndex-1)
        Quicksort(A,pIndex+1, end)
    }
}
```

Algorithm for Partition Function:

```
partition (array, start, end)
{
    // Setting rightmost Index as pivot
    pivot = arr[end];

    i = (start - 1)  // Index of smaller element and indicates the
        // right position of pivot found so far
    for (j = start; j <= end- 1; j++)
    {
        // If current element is smaller than the pivot
        if (arr[j] < pivot)
        {
            i++;    // increment index of smaller element
            swap arr[i] and arr[j]
        }
    }
    swap arr[i + 1] and arr[end])
    return (i + 1)
}
```

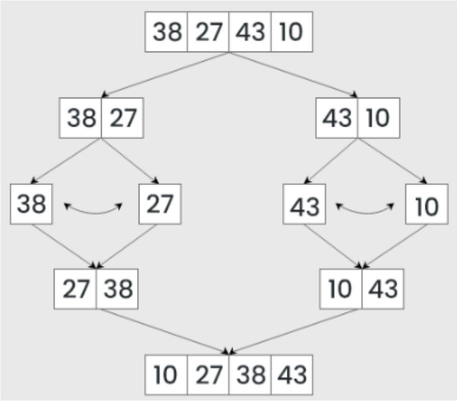
Input: arr = [10, 80, 30, 90, 40, 50, 70]
Output: arr = [10, 30, 40, 50, 70, 80, 90]

```
# Implementation of QuickSort

# Function to find the partition position
def partition(array, low, high):
    # write code here
    …

# Function to perform quicksort
def quicksort(array, low, high):
    # write code here
    …
```

```
# Driver code
array = [10, 7, 8, 9, 1, 5]
N = len(array)

# Function call
quicksort(array, 0, N - 1)
print('Sorted array:')
for x in array:
    print(x, end=" ")
```

4.2 Merge Sort

Merge sort is defined as a sorting algorithm that works by dividing an array into smaller subarrays, sorting each subarray, and then merging the sorted subarrays back together to form the final sorted array. In simple terms, we can say that the process of merge sort is to divide the array into two halves, sort each half, and then merge the sorted halves back together. This process is repeated until the entire array is sorted.



Input: arr = [12, 11, 13, 5, 6, 7]
Output: arr = [5, 6, 7, 11, 12, 13]

```
# Implementation of MergeSort

def mergeSort(arr):
    # write code here
    …

# print the list
def printList(arr):
    for i in range(len(arr)):
        print(arr[i], end=" ")
    print()


# Driver Code
arr = [12, 11, 13, 5, 6, 7]
print("Given array is")
printList(arr)
mergeSort(arr)
print("\nSorted array is ")
printList(arr)
```

4.3 Heap Sort

Heap sort is a comparison-based sorting technique based on Binary Heap data structure. It is similar to the selection sort where we first find the minimum element and place the minimum element at the beginning. Repeat the same process for the remaining elements.

Heap Sort Procedure:

First convert the array into heap data structure using heapify, then one by one delete the root node of the Max-heap and replace it with the last node in the heap and then heapify the root of the heap. Repeat this process until size of heap is greater than 1.

- Build a heap from the given input array.
- Repeat the following steps until the heap contains only one element:
    - Swap the root element of the heap (which is the largest element) with the last element of the heap.
    - Remove the last element of the heap (which is now in the correct position).
    - Heapify the remaining elements of the heap.
    - The sorted array is obtained by reversing the order of the elements in the input array.

Input: arr = [12, 11, 13, 5, 6, 7]
Output: Sorted array is 5 6 7 11 12 13

```
# Implementation of heap Sort
# To heapify subtree rooted at index i.
# n is size of heap

def heapify(arr, N, i):
    # write code here
    …

# The main function to sort an array of given size
def heapSort(arr):
    # write code here
    …
 # Driver code
arr = [12, 11, 13, 5, 6, 7]

# Function call
heapSort(arr)
N = len(arr)
print("Sorted array is")
for i in range(N):
    print("%d" % arr[i], end=" ")
```

4.4 Radix Sort

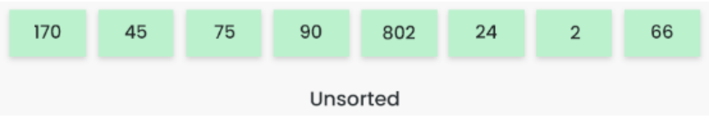
Radix Sort is a linear sorting algorithm that sorts elements by processing them digit by digit. It is an efficient sorting algorithm for integers or strings with fixed-size keys. Rather than comparing elements directly, Radix Sort distributes the elements into buckets based on each digit's value. By repeatedly sorting the elements by their significant digits, from the least significant to the most significant, Radix Sort achieves the final sorted order.

Radix Sort Procedure:

The key idea behind Radix Sort is to exploit the concept of place value.

1. It assumes that sorting numbers digit by digit will eventually result in a fully sorted list.
2. Radix Sort can be performed using different variations, such as Least Significant Digit (LSD) Radix Sort or Most Significant Digit (MSD) Radix Sort.

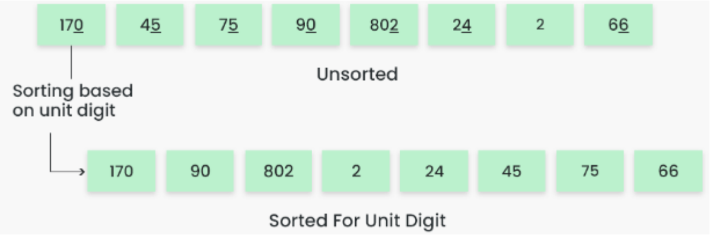
To perform radix sort on the array [170, 45, 75, 90, 802, 24, 2, 66], we follow these steps:



Step 1: Find the largest element in the array, which is 802. It has three digits, so we will iterate three times, once for each significant place.

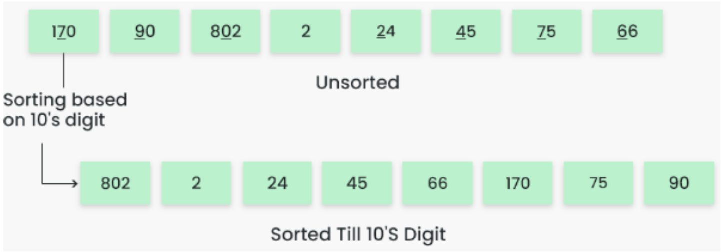
Step 2: Sort the elements based on the unit place digits (X=0). We use a stable sorting technique, such as counting sort, to sort the digits at each significant place.

Sorting based on the unit place:
Perform counting sort on the array based on the unit place digits.
The sorted array based on the unit place is [170, 90, 802, 2, 24, 45, 75, 66]



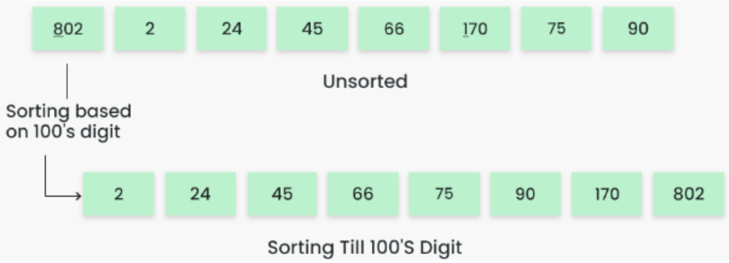
Step 3: Sort the elements based on the tens place digits.

Sorting based on the tens place:
Perform counting sort on the array based on the tens place digits.
The sorted array based on the tens place is [802, 2, 24, 45, 66, 170, 75, 90]



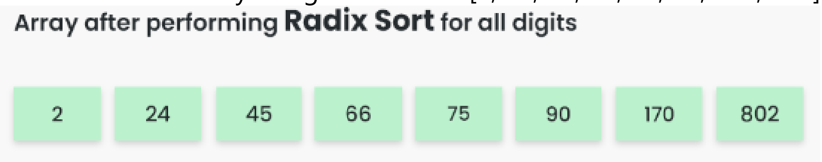
Step 4: Sort the elements based on the hundreds place digits.

Sorting based on the hundreds place:
Perform counting sort on the array based on the hundreds place digits.
The sorted array based on the hundreds place is [2, 24, 45, 66, 75, 90, 170, 802]



Step 5: The array is now sorted in ascending order.

The final sorted array using radix sort is [2, 24, 45, 66, 75, 90, 170, 802]



```
# Implementation of Radix Sort
# A function to do counting sort of arr[] according to the digit represented by exp.

def countingSort(arr, exp1):
    # write code here
    …

# Method to do Radix Sort
def radixSort(arr):
    # write code here
    …
# Driver code
arr = [170, 45, 75, 90, 802, 24, 2, 66]

# Function Call
radixSort(arr)

for i in range(len(arr)):
    print(arr[i],end=" ")
```

4.5 Shell Sort

Shell sort is mainly a variation of Insertion Sort. In insertion sort, we move elements only one position ahead. When an element has to be moved far ahead, many movements are involved. The idea of ShellSort is to allow the exchange of far items. In Shell sort, we make the array h-sorted for a large value of h. We keep reducing the value of h until it becomes 1. An array is said to be h-sorted if all sublists of every h'th element are sorted.

Shell Sort Procedure:

1. Initialize the value of gap size h
2. Divide the list into smaller sub-part. Each must have equal intervals to h
3. Sort these sub-lists using insertion sort
4. Repeat this step 1 until the list is sorted.
5. Print a sorted list.

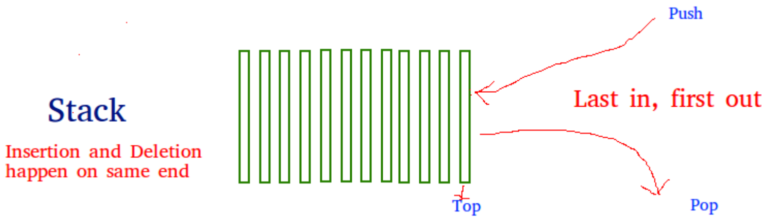
```
Procedure Shell_Sort(Array, N)
  While Gap < Length(Array) /3 :
            Gap = ( Interval * 3 ) + 1
  End While Loop
  While Gap > 0 :
    For (Outer = Gap; Outer < Length(Array); Outer++):
        Insertion_Value = Array[Outer]
            Inner = Outer;
        While Inner > Gap-1 And Array[Inner – Gap] >= Insertion_Value:
            Array[Inner] = Array[Inner – Gap]
            Inner = Inner – Gap
         End While Loop
           Array[Inner] = Insertion_Value
    End For Loop
    Gap = (Gap -1) /3;
  End While Loop
End Shell_Sort
```

```
# Implementation of Shell Sort

def shellSort(arr, n):
    # write code here
    …

# Driver code
arr = [12, 34, 54, 2, 3]
print("input array:",arr)

shellSort(arr,len(arr))
print("sorted array",arr)
```

5. Stack

5.1 Stack implementation using List

A stack is a linear data structure that stores items in a Last-In/First-Out (LIFO) or First-In/Last-Out (FILO) manner. In stack, a new element is added at one end and an element is removed from that end only. The insert and delete operations are often called push and pop.



The functions associated with stack are:

- **empty()** – Returns whether the stack is empty
- **size()** – Returns the size of the stack
- **top() / peek()** – Returns a reference to the topmost element of the stack
- **push(a)** – Inserts the element 'a' at the top of the stack
- **pop()** – Deletes the topmost element of the stack

```
# Stack implementation using list
top=0
```

```
mymax=5
def createStack():
    stack=[]
    return stack
def isEmpty(stack):
    # write code here
    …
def Push(stack,item):
    # write code here
    …
def Pop(stack):
    # write code here
    …
# create a stack object
stack = createStack()
while True:
    print("1.Push")
    print("2.Pop")
    print("3.Display")
    print("4.Quit")
    # write code here
    …
```

5.2 Balanced Parenthesis Checking

Given an expression string, write a python program to find whether a given string has balanced parentheses or not.

Input: {[]{()}}

Output: Balanced

Input: [{}{}(]

Output: Unbalanced

Using stack One approach to check balanced parentheses is to use stack. Each time, when an open parentheses is encountered push it in the stack, and when closed parenthesis is encountered, match it with the top of stack and pop it. If stack is empty at the end, return Balanced otherwise, Unbalanced.

```
# Check for balanced parentheses in an expression
open_list = ["[","{","("]
close_list = ["]","}",")"]
 # Function to check parentheses
def check(myStr):
    # write code here
    …
```

5.3 Evaluation of Postfix Expression

Given a postfix expression, the task is to evaluate the postfix expression. Postfix expression: The expression of the form “a b operator” (ab+) i.e., when a pair of operands is followed by an operator.

Input: str = “2 3 1 * + 9 -“

Output: -4

Explanation: If the expression is converted into an infix expression, it will be 2 + (3 * 1) – 9 = 5 – 9 = -4.

Input: str = “100 200 + 2 / 5 * 7 +”

Output: 757

Procedure for evaluation postfix expression using stack:

- Create a stack to store operands (or values).
- Scan the given expression from left to right and do the following for every scanned element.
    - If the element is a number, push it into the stack.
    - If the element is an operator, pop operands for the operator from the stack. Evaluate the operator and push the result back to the stack.
- When the expression is ended, the number in the stack is the final answer.

```
# Evaluate value of a postfix expression

# Class to convert the expression
class Evaluate:

    # Constructor to initialize the class variables
    def __init__(self, capacity):
        self.top = -1
        self.capacity = capacity

        # This array is used a stack
        self.array = []

    # Check if the stack is empty
    def isEmpty(self):
        # write code here

        …

    def peek(self):
        # write code here

        …

    def pop(self):
        # write code here

        …

    def push(self, op):
        # write code here

        …

    def evaluatePostfix(self, exp):
        # write code here

        …

# Driver code
exp = "231*+9-"
```

```
obj = Evaluate(len(exp))

# Function call
print("postfix evaluation: %d" % (obj.evaluatePostfix(exp)))
```

5.4 Infix to Postfix Expression Conversion

For a given Infix expression, convert it into Postfix form.
Infix expression: The expression of the form "a operator b" (a + b) i.e., when an operator is in-between every pair of operands.
Postfix expression: The expression of the form "a b operator" (ab+) i.e., When every pair of operands is followed by an operator.

Infix to postfix expression conversion procedure:

1. Scan the infix expression from left to right.
2. If the scanned character is an operand, put it in the postfix expression.
3. Otherwise, do the following
    - If the precedence and associativity of the scanned operator are greater than the precedence and associativity of the operator in the stack [or the stack is empty or the stack contains a '('], then push it in the stack. ['^' operator is right associative and other operators like '+','–','*' and '/' are left-associative].
    - Check especially for a condition when the operator at the top of the stack and the scanned operator both are '^'. In this condition, the precedence of the scanned operator is higher due to its right associativity. So it will be pushed into the operator stack.
    - In all the other cases when the top of the operator stack is the same as the scanned operator, then pop the operator from the stack because of left associativity due to which the scanned operator has less precedence.
    - Else, Pop all the operators from the stack which are greater than or equal to in precedence than that of the scanned operator.
    - After doing that Push the scanned operator to the stack. (If you encounter parenthesis while popping then stop there and push the scanned operator in the stack.)
4. If the scanned character is a '(', push it to the stack.
5. If the scanned character is a ')', pop the stack and output it until a '(' is encountered, and discard both the parenthesis.
6. Repeat steps 2-5 until the infix expression is scanned.
7. Once the scanning is over, Pop the stack and add the operators in the postfix expression until it is not empty.
8. Finally, print the postfix expression.

Input: A + B * C + D
Output: A B C * + D +

Input: ((A + B) – C * (D / E)) + F
Output: A B + C D E / * - F +

```
# Convert infix expression to postfix
# Class to convert the expression

class Conversion:

    # Constructor to initialize the class variables
    def __init__(self, capacity):
        self.top = -1
        self.capacity = capacity

        # This array is used a stack
        self.array = []


        # Precedence setting
        self.output = []
        self.precedence = {'+': 1, '-': 1, '*': 2, '/': 2, '^': 3}

    # Check if the stack is empty
    def isEmpty(self):
        # write code here
        …

    # Return the value of the top of the stack
    def peek(self):
        # write code here
        …

    # Pop the element from the stack
    def pop(self):
        # write code here
        …

    # Push the element to the stack
    def push(self, op):
        # write code here
        …

    # A utility function to check is the given character is operand
    def isOperand(self, ch):
        # write code here
        …

    # Check if the precedence of operator is strictly less than top of stack or not
    def notGreater(self, i):
        # write code here
        …
    # The main function that converts given infix expression
    # to postfix expression
    def infixToPostfix(self, exp):
       # write code here
        …
```

```
# Driver code
exp = "a+b*(c^d-e)^(f+g*h)-i"
obj = Conversion(len(exp))

# Function call
obj.infixToPostfix(exp)
```

5.5 Reverse a Stack

The stack is a linear data structure which works on the LIFO concept. LIFO stands for last in first out. In the stack, the insertion and deletion are possible at one end the end is called the top of the stack. Define two recursive functions BottomInsertion() and Reverse() to reverse a stack using Python. Define some basic function of the stack like push(), pop(), show(), empty(), for basic operation like respectively append an item in stack, remove an item in stack, display the stack, check the given stack is empty or not.

BottomInsertion(): this method append element at the bottom of the stack and BottomInsertion accept two values as an argument first is stack and the second is elements, this is a recursive method.

Reverse(): the method is reverse elements of the stack, this method accept stack as an argument Reverse() is also a Recursive() function. Reverse() is invoked BottomInsertion() method for completing the reverse operation on the stack.

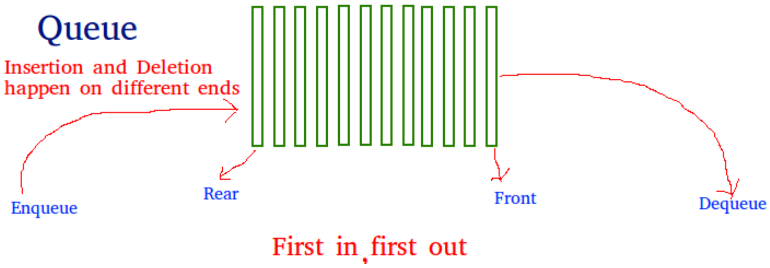
Input: Elements = [1, 2, 3, 4, 5]
Output: Original Stack
5
4
3
2
1
Stack after Reversing
1
2
3
4
5

```
# create class for stack
class Stack:

    # create empty list
    def __init__(self):
        self.Elements = []

    # push() for insert an element
    def push(self, value):
        self.Elements.append(value)

    # pop() for remove an element
    def pop(self):
        return self.Elements.pop()

    # empty() check the stack is empty of not
    def empty(self):
        return self.Elements == []
    # show() display stack
    def show(self):
        for value in reversed(self.Elements):
            print(value)
 # Insert_Bottom() insert value at bottom
def BottomInsert(s, value):
   # write code here

   …
# Reverse() reverse the stack
def Reverse(s):
   # write code here

   …
# create object of stack class
stk = Stack()

stk.push(1)
stk.push(2)
stk.push(3)
stk.push(4)
stk.push(5)

print("Original Stack")
stk.show()

print("\nStack after Reversing")
Reverse(stk)
stk.show()
```

6. Queue

6.1 Linear Queue

Linear queue is a linear data structure that stores items in First in First out (FIFO) manner. With a queue the least recently added item is removed first. A good example of queue is any queue of consumers for a resource where the consumer that came first is served first.



```
# Static implementation of linear queue
front=0
rear=0
mymax=5
def createQueue():
    queue=[]   #empty list
    return queue
def isEmpty(queue):
    # write code here
    …
def enqueue(queue,item):  # insert an element into the queue
    # write code here
    …
def dequeue(queue):  #remove an element from the queue
    # write code here
    …
# Driver code
queue = createQueue()
while True:
    print("1.Enqueue")
    print("2.Dequeue")
    print("3.Display")
    print("4.Quit")
    # write code here
    …
```

6.2 Stack using Queues

Implement a last-in-first-out (LIFO) stack using only two queues. The implemented stack should support all the functions of a normal stack (push, top, pop, and empty).

- void push(int x) Pushes element x to the top of the stack.
- int pop() Removes the element on the top of the stack and returns it.
- int top() Returns the element on the top of the stack.
- boolean empty() Returns true if the stack is empty, false otherwise.

Input:

["MyStack", "push", "push", "top", "pop", "empty"]

[[], [1], [2], [], [], []]

Output:

[null, null, null, 2, 2, false]

```
class MyStack:

    def __init__(self):
       # write code here
       …

    def push(self, x: int) -> None:
       # write code here
       …

    def pop(self) -> int:
       # write code here
       …
    def top(self) -> int:
       # write code here
       …
    def empty(self) -> bool:
       # write code here
       …
# Your MyStack object will be instantiated and called as such:
# obj = MyStack()
# obj.push(x)
# param_2 = obj.pop()
# param_3 = obj.top()
# param_4 = obj.empty()
```

6.3 Queue using Stacks

Implement a first in first out (FIFO) queue using only two stacks. The implemented queue should support all the functions of a normal queue (push, peek, pop, and empty).

- void push(int x) Pushes element x to the back of the queue.
- int pop() Removes the element from the front of the queue and returns it.
- int peek() Returns the element at the front of the queue.
- boolean empty() Returns true if the queue is empty, false otherwise.

Input:

["MyQueue", "push", "push", "peek", "pop", "empty"]

[[], [1], [2], [], [], []]

Output:

[null, null, null, 1, 1, false]

```
class MyQueue:

    def __init__(self):
       # write code here
       …
    def push(self, x: int) -> None:
       # write code here
       …
    def pop(self) -> int:
       # write code here
       …
    def peek(self) -> int:
       # write code here
       …
    def empty(self) -> bool:
       # write code here
```

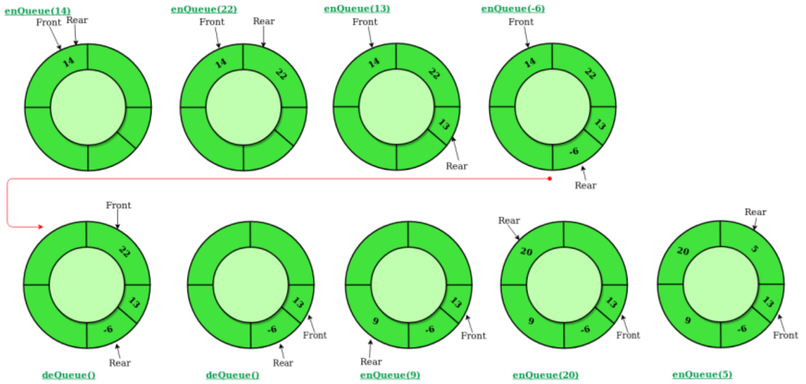
```
        …
# Your MyQueue object will be instantiated and called as such:
# obj = MyQueue()
# obj.push(x)
# param_2 = obj.pop()
# param_3 = obj.peek()
# param_4 = obj.empty()
```

6.4 Circular Queue

A Circular Queue is an extended version of a normal queue where the last element of the queue is connected to the first element of the queue forming a circle. The operations are performed based on FIFO (First In First Out) principle. It is also called 'Ring Buffer'.

Operations on Circular Queue:

- **Front:** Get the front item from the queue.
- **Rear:** Get the last item from the queue.
- **enQueue(value)** This function is used to insert an element into the circular queue. In a circular queue, the new element is always inserted at the rear position.
    - Check whether the queue is full – [i.e., the rear end is in just before the front end in a circular manner].
    - If it is full then display Queue is full.
        - If the queue is not full then, insert an element at the end of the queue.
- **deQueue()** This function is used to delete an element from the circular queue. In a circular queue, the element is always deleted from the front position.
    - Check whether the queue is Empty.
    - If it is empty then display Queue is empty.
        - If the queue is not empty, then get the last element and remove it from the queue.



Implement Circular Queue using Array:

1. Initialize an array queue of size **n**, where n is the maximum number of elements that the queue can hold.
2. Initialize two variables front and rear to -1.
3. **Enqueue:** To enqueue an element **x** into the queue, do the following:
    - Increment rear by 1.

- If **rear** is equal to n, set **rear** to 0.
- If **front** is -1, set **front** to 0.
- Set queue[rear] to x.

4. **Dequeue:** To dequeue an element from the queue, do the following:
   - Check if the queue is empty by checking if **front** is -1.
   - If it is, return an error message indicating that the queue is empty.
   - Set **x** to queue [front].
   - If **front** is equal to **rear**, set **front** and **rear** to -1.
   - Otherwise, increment **front** by 1 and if **front** is equal to n, set **front** to 0.
   - Return x.

```
class CircularQueue():

    # constructor
    def __init__(self, size): # initializing the class
        self.size = size

        # initializing queue with none
        self.queue = [None for i in range(size)]
        self.front = self.rear = -1

    def enqueue(self, data):
         # Write code here
         …

    def dequeue(self):
         # Write code here
         …

    def display(self):
        # Write code here
        …
# Driver Code
ob = CircularQueue(5)
ob.enqueue(14)
ob.enqueue(22)
ob.enqueue(13)
ob.enqueue(-6)
ob.display()
print ("Deleted value = ", ob.dequeue())
print ("Deleted value = ", ob.dequeue())
ob.display()
ob.enqueue(9)
ob.enqueue(20)
ob.enqueue(5)
ob.display()
```

6.5 Deque (Doubly Ended Queue)

In a Deque (Doubly Ended Queue), one can perform insert (append) and delete (pop) operations from both the ends of the container. There are two types of Deque:

1. **Input Restricted Deque:** Input is limited at one end while deletion is permitted at both ends.
2. **Output Restricted Deque:** Output is limited at one end but insertion is permitted at both ends.

Operations on Deque:

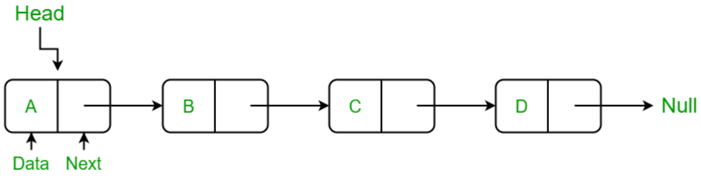
1. **append():** This function is used to insert the value in its argument to the right end of the deque.
2. **appendleft():** This function is used to insert the value in its argument to the left end of the deque.
3. **pop():** This function is used to delete an argument from the right end of the deque.
4. **popleft():** This function is used to delete an argument from the left end of the deque.
5. **index(ele, beg, end):** This function returns the first index of the value mentioned in arguments, starting searching from beg till end index.
6. **insert(i, a):** This function inserts the value mentioned in arguments(a) at index(i) specified in arguments.
7. **remove():** This function removes the first occurrence of the value mentioned in arguments.
8. **count():** This function counts the number of occurrences of value mentioned in arguments.
9. **len(dequeue):** Return the current size of the dequeue.
10. **Deque[0]:** We can access the front element of the deque using indexing with de[0].
11. **Deque[-1]:** We can access the back element of the deque using indexing with de[-1].
12. **extend(iterable):** This function is used to add multiple values at the right end of the deque. The argument passed is iterable.
13. **extendleft(iterable):** This function is used to add multiple values at the left end of the deque. The argument passed is iterable. Order is reversed as a result of left appends.
14. **reverse():** This function is used to reverse the order of deque elements.
15. **rotate():** This function rotates the deque by the number specified in arguments. If the number specified is negative, rotation occurs to the left. Else rotation is to right.

```
# importing "collections" for deque operations
import collections

# initializing deque
de = collections.deque([1, 2, 3])
print("deque: ", de)

# using append() to insert 4 at the end of deque
# Write code here

# Printing modified deque
# Write code here

# using appendleft() to insert 6 at the beginning of deque
# Write code here

# Printing modified deque
# Write code here

# using pop() to delete 4 from the right end of deque
# Write code here

# Printing modified deque
# Write code here

# using popleft() to delete 6 from the left end of deque
# Write code here

# Printing modified deque
# Write code here

# using insert() to insert the value 3 at 5th position
# Write code here

# printing modified deque
# Write code here

# using count() to count the occurrences of 3
# Write code here

# using remove() to remove the first occurrence of 3
# Write code here

# Printing modified deque
# Write code here

# Printing current size of deque
# Write code here

# using pop() to delete 6 from the right end of deque
# Write code here

# Printing modified deque
# Write code here

# Printing current size of deque
# Write code here

# Accessing the front element of the deque
# Write code here

# Accessing the back element of the deque
# Write code here

# using extend() to add 4,5,6 to right end
# Write code here

# Printing modified deque
# Write code here

# using extendleft() to add 7,8,9 to left end
# Write code here

# Printing modified deque
# Write code here

# using rotate() to rotate the deque rotates by 3 to left
# Write code here

# Printing modified deque
# Write code here

# using reverse() to reverse the deque
# Write code here

# Printing modified deque
# Write code here
```

7. Linked List

7.1 Singly Linked List

A singly linked list is a linear data structure in which the elements are not stored in contiguous memory locations and each element is connected only to its next element using a pointer.



Creating a linked list involves the following operations:

1. Creating a Node class:
2. Insertion at beginning:
3. Insertion at end
4. Insertion at middle
5. Update the node
6. Deletion at beginning
7. Deletion at end
8. Deletion at middle
9. Remove last node
10. Linked list traversal
11. Get length

```
# Create a Node class to create a node
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None


# Create a LinkedList class
class LinkedList:
    def __init__(self):
        self.head = None
```

```
     # Method to add a node at begin of LL
    def insertAtBegin(self, data):
        # Write code here
        …
    # Method to add a node at any index, Indexing starts from 0.
    def insertAtIndex(self, data, index):
        # Write code here
        …
     # Method to add a node at the end of LL
    def insertAtEnd(self, data):
        # Write code here
        …
     # Update node of a linked list at given position
    def updateNode(self, val, index):
        # Write code here
        …
    # Method to remove first node of linked list
    def remove_first_node(self):
        # Write code here
        …
    # Method to remove last node of linked list
    def remove_last_node(self):
        # Write code here
        …
    # Method to remove at given index
    def remove_at_index(self, index):
        # Write code here
        …
     # Method to remove a node from linked list
    def remove_node(self, data):
        # Write code here
        …
     # Print the size of linked list
    def sizeOfLL(self):
        # Write code here
        …
    # print method for the linked list
    def printLL(self):
        # Write code here
        …
# create a new linked list
```

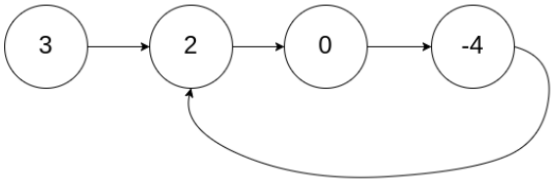
```
llist = LinkedList()
# add nodes to the linked list
llist.insertAtEnd('a')
llist.insertAtEnd('b')
llist.insertAtBegin('c')
llist.insertAtEnd('d')
llist.insertAtIndex('g', 2)
# print the linked list
print("Node Data")
llist.printLL()
# remove a nodes from the linked list
print("\nRemove First Node")
llist.remove_first_node()
print("Remove Last Node")
llist.remove_last_node()
print("Remove Node at Index 1")
llist.remove_at_index(1)
# print the linked list again
print("\nLinked list after removing a node:")
llist.printLL()
print("\nUpdate node Value")
llist.updateNode('z', 0)
llist.printLL()
print("\nSize of linked list :", end=" ")
print(llist.sizeOfLL())
```

7.2 Linked List Cycle

Given head, the head of a linked list, determine if the linked list has a cycle in it. There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Internally, pos is used to denote the index of the node that tail's next pointer is connected to.

Note that pos is not passed as a parameter.

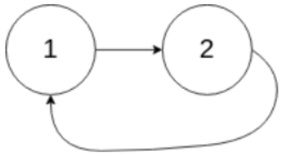
Return true if there is a cycle in the linked list. Otherwise, return false.



Input: head = [3, 2, 0, -4], pos = 1

Output: true

Explanation: There is a cycle in the linked list, where the tail connects to the 1st node (0-indexed).



Input: head = [1, 2], pos = 0

Output: true

Explanation: There is a cycle in the linked list, where the tail connects to the 0th node.

Input: head = [1], pos = -1

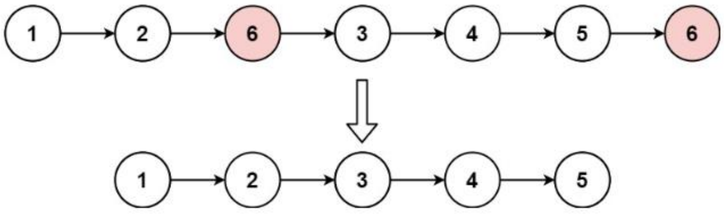
Output: false

Explanation: There is no cycle in the linked list.

```
# Definition for singly-linked list.
class ListNode:
    def __init__(self, x):
        self.val = x
        self.next = None

class Solution:
    def hasCycle(self, head):
        # Write code here
        …
```

7.3 Remove Linked List Elements

Given the head of a linked list and an integer val, remove all the nodes of the linked list that has Node.val == val, and return the new head.



Input: head = [1, 2, 6, 3, 4, 5, 6], val = 6

Output: [1, 2, 3, 4, 5]

Input: head = [ ], val = 1

Output: [ ]

Input: head = [7, 7, 7, 7], val = 7

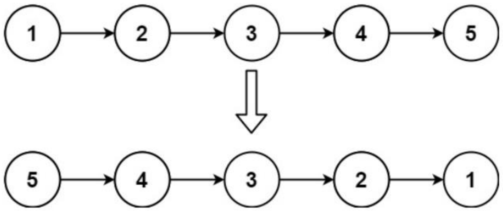
Output: [ ]

```
# Definition for singly-linked list.
class ListNode:
   def __init__(self, val=0, next=None):
       self.val = val
       self.next = next

class Solution:
    def removeElements(self, head, val):
        # Write code here
        …
```

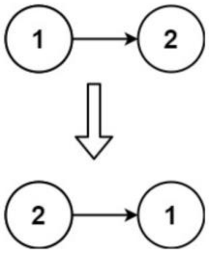
7.4 Reverse Linked List

Given the head of a singly linked list, reverse the list, and return the reversed list.

Input: head = [1, 2, 3, 4, 5]
Output: [5, 4, 3, 2, 1]



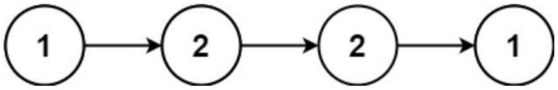
Input: head = [1, 2]
Output: [2, 1]



```
# Definition for singly-linked list.
class ListNode:
   def __init__(self, val=0, next=None):
       self.val = val
       self.next = next

class Solution:
    def reverseList(self, head):
        # Write code here
        …
```

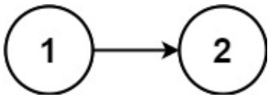
7.5 Palindrome Linked List

Given the head of a singly linked list, return true if it is a palindrome or false otherwise.



Input: head = [1, 2, 2, 1]

Output: true



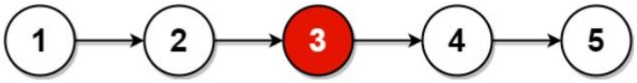
Input: head = [1, 2]

Output: false

```
# Definition for singly-linked list.
class ListNode:
   def __init__(self, val=0, next=None):
       self.val = val
       self.next = next

class Solution:
    def isPalindrome(self, head):
        # Write code here
         …
```

7.6 Middle of the Linked List

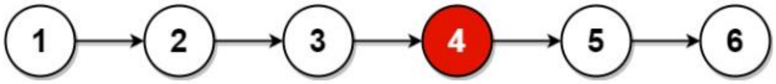
Given the head of a singly linked list, return the middle node of the linked list. If there are two middle nodes, return the second middle node.



Input: head = [1, 2, 3, 4, 5]

Output: [3, 4, 5]

Explanation: The middle node of the list is node 3.



Input: head = [1, 2, 3, 4, 5, 6]

Output: [4, 5, 6]

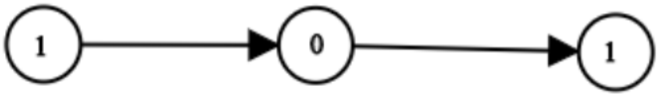
Explanation: Since the list has two middle nodes with values 3 and 4, we return the second one.

```
# Definition for singly-linked list.
class ListNode:
   def __init__(self, val=0, next=None):
       self.val = val
       self.next = next

class Solution:
    def middleNode(self, head):
        # Write code here
         …
```

7.7 Convert Binary Number in a Linked List to Integer

Given head which is a reference node to a singly-linked list. The value of each node in the linked list is either 0 or 1. The linked list holds the binary representation of a number.

Return the decimal value of the number in the linked list. The most significant bit is at the head of the linked list.



Input: head = [1, 0, 1]

Output: 5

Explanation: (101) in base 2 = (5) in base 10

Input: head = [0]

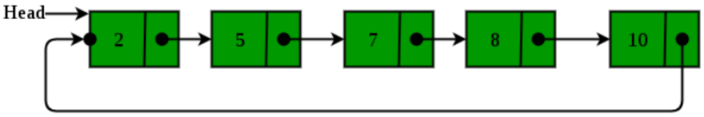
Output: 0

```
# Definition for singly-linked list.
class ListNode:
   def __init__(self, val=0, next=None):
       self.val = val
       self.next = next

class Solution:
    def getDecimalValue(self, head):
        # Write code here
         …
```

8. Circular Single Linked List and Doubly Linked List

8.1 Circular Linked List

The circular linked list is a linked list where all nodes are connected to form a circle. In a circular linked list, the first node and the last node are connected to each other which forms a circle. There is no NULL at the end.



Operations on the circular linked list:

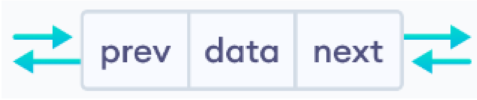
1. Insertion at the beginning
2. Insertion at the end
3. Insertion in between the nodes
4. Deletion at the beginning
5. Deletion at the end
6. Deletion in between the nodes
7. Traversal

```
# Circular linked list operations

class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class CircularLinkedList:
    def __init__(self):
        self.last = None
    def addToEmpty(self, data):
        # Write code here
        …

    # add node to the front
    def addFront(self, data):
        # Write code here
        …

    # add node to the end
    def addEnd(self, data):
        # Write code here
        …

    # insert node after a specific node
    def addAfter(self, data, item):
        # Write code here
        …

    # delete a node
    def deleteNode(self, last, key):
        # Write code here
        …

    def traverse(self):
        # Write code here
        …
# Driver Code
cll = CircularLinkedList()

last = cll.addToEmpty(6)
last = cll.addEnd(8)
last = cll.addFront(2)
last = cll.addAfter(10, 2)

cll.traverse()
last = cll.deleteNode(last, 8)
print()
cll.traverse()
```

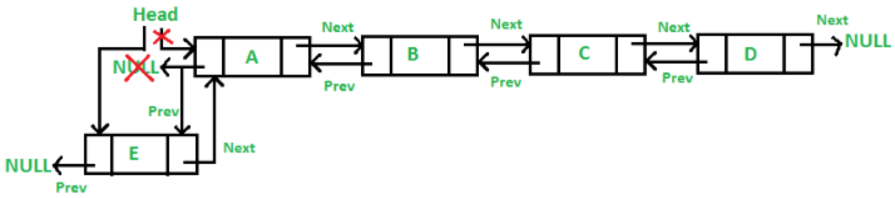
8.2 Doubly Linked List

The A doubly linked list is a type of linked list in which each node consists of 3 components:

1. *prev - address of the previous node
2. data - data item
3. *next - address of next node.



Double Linked List Node



Operations on the Double Linked List:

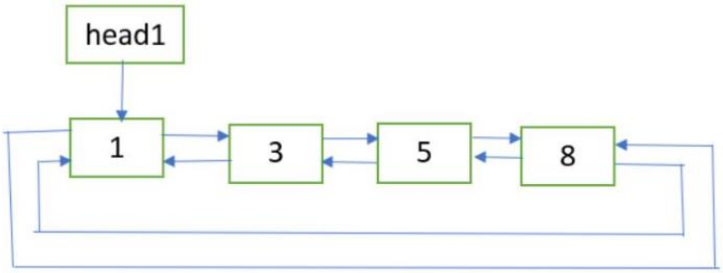
1. Insertion at the beginning
2. Insertion at the end
3. Insertion in between the nodes
4. Deletion at the beginning
5. Deletion at the end
6. Deletion in between the nodes
7. Traversal

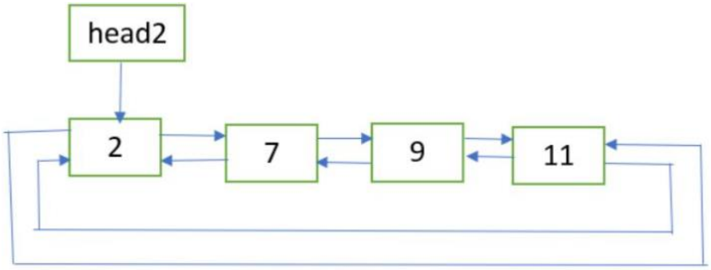
```
# Implementation of doubly linked list
class Node:
    def __init__(self,data):
        self.data=data
        self.next=self.prev=None

class DLinkedList:
    def __init__(self):
        self.head=None
        self.ctr=0
    def insert_beg(self,data):
        # Write code here
        …
    def insert_end(self,data):
        # Write code here
        …
    def delete_beg(self):
        # Write code here
        …
    def delete_end(self):
        # Write code here
        …
    def insert_pos(self,pos,data):
        # Write code here
        …
    def delete_pos(self,pos):
        # Write code here
        …
    def traverse_f(self):
        # Write code here
        …
    def traverse_r(self):
        # Write code here
        …

def menu():
    print("1.Insert at beginning")
    print("2.Insert at position")
    print("3.Insert at end")
    print("4.Delete at beginning")
    print("5.Delete at position")
    print("6.Delete at end")
    print("7.Count no of nodes")
    print("8.Traverse forward")
    print("9.Traverse reverse")
    print("10.Quit")
    ch=eval(input("Enter choice:"))
    return ch

print("******************Double linked list**************")
d=DLinkedList()
while True :
    ch=menu()
    if ch==1:
        data=eval(input("Enter data:"))
        d.insert_beg(data)

    elif ch==2:
        data=eval(input("Enter data:"))
        pos=int(input("Enter position:"))
        d.insert_pos(pos,data)

    elif ch==3:
        data=eval(input("Enter data:"))
        d.insert_end(data)

    elif ch==4:
        d.delete_beg()

    elif ch==5:
        pos=int(input("Enter position:"))
        d.delete_pos(pos)

    elif ch==6:
        d.delete_end()

    elif ch==7:
        print("Number of nodes",d.ctr)

    elif ch==8:
        d.traverse_f()
    elif ch==9:
        d.traverse_r()

    else:
        print("Exit")
        break
```

8.3 Sorted Merge of Two Sorted Doubly Circular Linked Lists

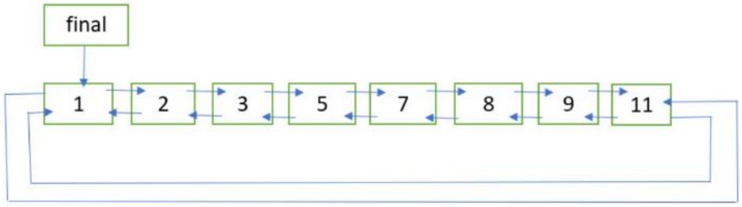
Given two sorted Doubly circular Linked List containing n1 and n2 nodes respectively. The problem is to merge the two lists such that resultant list is also in sorted order.

Input: List 1 and List 2





Output: Merged List



Procedure for Merging Doubly Linked List:

1. If head1 == NULL, return head2.
2. If head2 == NULL, return head1.
3. Let **last1** and **last2** be the last nodes of the two lists respectively. They can be obtained with the help of the previous links of the first nodes.
4. Get pointer to the node which will be the last node of the final list. If last1.data < last2.data, then **last_node** = last2, Else **last_node** = last1.
5. Update last1.next = last2.next = NULL.
6. Now merge the two lists as two sorted doubly linked list are being merged. Refer **merge** procedure of this post. Let the first node of the final list be **finalHead**.
7. Update finalHead.prev = last_node and last_node.next = finalHead.
8. Return **finalHead**.

```
# Implementation for Sorted merge of two sorted doubly circular linked list

import math

class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
        self.prev = None
```

```
# A utility function to insert a new node at the beginning
# of doubly circular linked list
def insert(head_ref, data):
     # Write code here
     …
# function for Sorted merge of two sorted doubly linked list
def merge(first, second):
    # Write code here
    …
# function for Sorted merge of two sorted doubly circular linked list
def mergeUtil(head1, head2):
    # Write code here
    …
# function to print the list
def printList(head):
    # Write code here
    …
# Driver Code
head1 = None
head2 = None
# list 1:
head1 = insert(head1, 8)
head1 = insert(head1, 5)
head1 = insert(head1, 3)
head1 = insert(head1, 1)
# list 2:
head2 = insert(head2, 11)
head2 = insert(head2, 9)
head2 = insert(head2, 7)
head2 = insert(head2, 2)

newHead = mergeUtil(head1, head2)

print("Final Sorted List: ", end = "")
printList(newHead)
```

8.4 Delete all occurrences of a given key in a Doubly Linked List

Given a doubly linked list and a key x. The problem is to delete all occurrences of the given key x from the doubly linked list.

Input: 2 <-> 2 <-> 10 <-> 8 <-> 4 <-> 2 <-> 5 <-> 2
x = 2
Output: 10 <-> 8 <-> 4 <-> 5

Algorithm:

```
delAllOccurOfGivenKey (head_ref, x)
   if head_ref == NULL
      return
   Initialize current = head_ref
   Declare next
   while current != NULL
      if current->data == x
         next = current->next
         deleteNode(head_ref, current)
         current = next
```

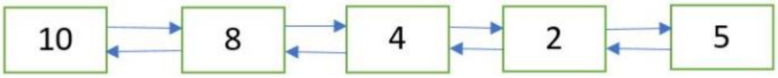
```
        else
            current = current->next
```

```python
# Implementation to delete all occurrences of a given key in a doubly linked list
import math

# a node of the doubly linked list
class Node:
    def __init__(self,data):
        self.data = data
        self.next = None
        self.prev = None

# Function to delete a node in a Doubly Linked List.
# head_ref --> pointer to head node pointer.
# del --> pointer to node to be deleted.
def deleteNode(head, delete):
    # Write code here
    …

# function to delete all occurrences of the given key 'x'
def deleteAllOccurOfX(head, x):
    # Write code here
    …

# Function to insert a node at the beginning of the Doubly Linked List
def push(head,new_data):
    # Write code here
    …

# Function to print nodes in a given doubly linked list
def printList(head):
    # Write code here
    …

# Driver Code
# Start with the empty list
head = None
# Create the doubly linked list:
head = push(head, 2)
head = push(head, 5)
head = push(head, 2)
head = push(head, 4)
head = push(head, 8)
head = push(head, 10)
head = push(head, 2)
head = push(head, 2)
print("Original Doubly linked list:")
printList(head)
x = 2
# delete all occurrences of 'x'
head = deleteAllOccurOfX(head, x)
print("\nDoubly linked list after deletion of ",x,":")
printList(head)
```

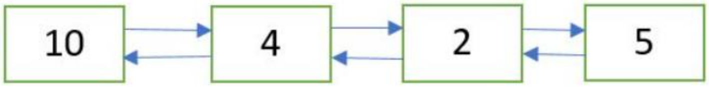
8.5 Delete a Doubly Linked List Node at a Given Position

Given a doubly linked list and a position n. The task is to delete the node at the given position n from the beginning.

Input: Initial doubly linked list



Output: Doubly Linked List after deletion of node at position n = 2



Procedure:

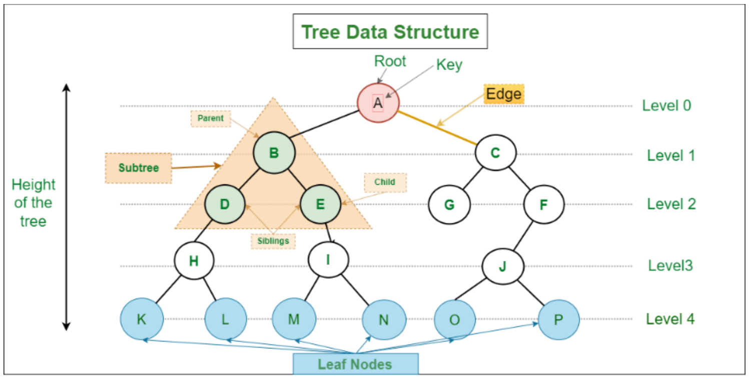
1. Get the pointer to the node at position n by traversing the doubly linked list up to the nth node from the beginning.
2. Delete the node using the pointer obtained in Step 1.

```
# Python implementation to delete a doubly Linked List node
# at the given position

# A node of the doubly linked list
class Node:

    # Constructor to create a new node
    def __init__(self, data):
        self.data = data
        self.next = None
        self.prev = None

# Function to delete a node in a Doubly Linked List.
# head_ref -. pointer to head node pointer.
# del -. pointer to node to be deleted.
def deleteNode(head_ref, del_):
    # Write code here
    …

# Function to delete the node at the given position
# in the doubly linked list
def deleteNodeAtGivenPos(head_ref, n):
    # Write code here
    …

# Function to insert a node at the beginning of the Doubly Linked List
def push(head_ref, new_data):
    # Write code here
    …

# Function to print nodes in a given doubly linked list
def printList(head):
    # Write code here
    …
# Driver Code
# Start with the empty list
head = None
head = push(head, 5)
head = push(head, 2)
head = push(head, 4)
head = push(head, 8)
head = push(head, 10)
print("Doubly linked list before deletion:")
printList(head)

n = 2

# delete node at the given position 'n'
head = deleteNodeAtGivenPos(head, n)
print("\nDoubly linked list after deletion:")
printList(head)
```

9. Trees

9.1 Tree Creation and Basic Tree Terminologies

A tree data structure is a hierarchical structure that is used to represent and organize data in a way that is easy to navigate and search. It is a collection of nodes that are connected by edges and has a hierarchical relationship between the nodes.



Basic Terminologies in Tree:

1. **Parent Node:** The node which is a predecessor of a node is called the parent node of that node. {B} is the parent node of {D, E}.

2. **Child Node:** The node which is the immediate successor of a node is called the child node of that node. Examples: {D, E} are the child nodes of {B}.

3. **Root Node:** The topmost node of a tree or the node which does not have any parent node is called the root node. {A} is the root node of the tree. A non-empty tree must contain exactly one root node and exactly one path from the root to all other nodes of the tree.

4. **Leaf Node or External Node:** The nodes which do not have any child nodes are called leaf nodes. {K, L, M, N, O, P} are the leaf nodes of the tree.

5. **Ancestor of a Node:** Any predecessor nodes on the path of the root to that node are called Ancestors of that node. {A, B} are the ancestor nodes of the node {E}

6. **Descendant:** Any successor node on the path from the leaf node to that node. {E, I} are the descendants of the node {B}.

8. **Sibling:** Children of the same parent node are called siblings. {D, E} are called siblings.

9. **Level of a node:** The count of edges on the path from the root node to that node. The root node has level 0.

10. **Internal node:** A node with at least one child is called Internal Node.
11. **Neighbour of a Node:** Parent or child nodes of that node are called neighbors of that node.
12. **Subtree:** Any node of the tree along with its descendant.

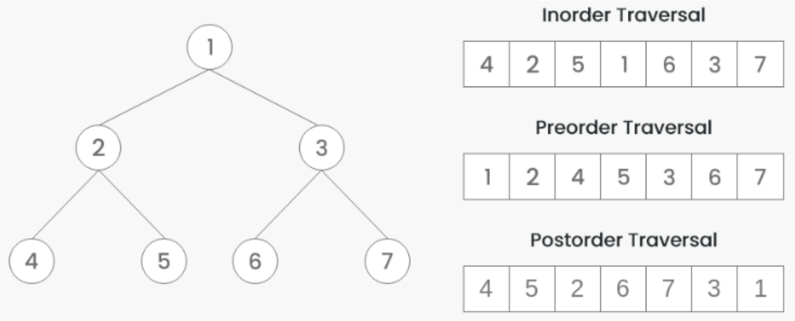
```
# Demonstration of Tree Basic Terminologies
# Function to add an edge between vertices x and y

# Function to print the parent of each node
def printParents(node, adj, parent):
    # Write code here
    …
# Function to print the children of each node
def printChildren(Root, adj):
    # Write code here
    …

# Function to print the leaf nodes
def printLeafNodes(Root, adj):
    # Write code here
    …

# Function to print the degrees of each node
def printDegrees(Root, adj):
    # Write code here
    …

# Driver code
# Number of nodes
N = 7
Root = 1

# Adjacency list to store the tree
adj = []
for i in range(0, N+1):
    adj.append([])

# Creating the tree
adj[1].append(2)
adj[2].append(1)

adj[1].append(3)
adj[3].append(1)

adj[1].append(4)
adj[4].append(1)

adj[2].append(5)
adj[5].append(2)

adj[2].append(6)
adj[6].append(2)

adj[4].append(7)
adj[7].append(4)

# Printing the parents of each node
print("The parents of each node are:")
```

```
printParents(Root, adj, 0)
# Printing the children of each node
print("The children of each node are:")
printChildren(Root, adj)

# Printing the leaf nodes in the tree
print("The leaf nodes of the tree are:")
printLeafNodes(Root, adj)

# Printing the degrees of each node
print("The degrees of each node are:")
printDegrees(Root, adj)
```

9.2 Binary Tree Traversal Techniques

A binary tree data structure can be traversed in following ways:

1. Inorder Traversal
2. Preorder Traversal
3. Postorder Traversal
4. Level Order Traversal



Inorder Traversal

4	2	5	1	6	3	7

Preorder Traversal

1	2	4	5	3	6	7

Postorder Traversal

4	5	2	6	7	3	1

Algorithm Inorder (tree)

1. Traverse the left subtree, i.e., call Inorder(left->subtree)
2. Visit the root.
3. Traverse the right subtree, i.e., call Inorder(right->subtree)

Algorithm Preorder (tree)

1. Visit the root.
2. Traverse the left subtree, i.e., call Preorder(left->subtree)
3. Traverse the right subtree, i.e., call Preorder(right->subtree)

Algorithm Postorder (tree)

1. Traverse the left subtree, i.e., call Postorder(left->subtree)
2. Traverse the right subtree, i.e., call Postorder(right->subtree)
3. Visit the root.

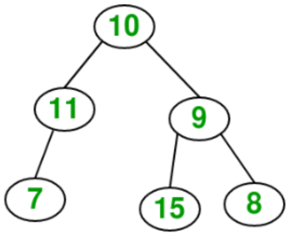
```
# Program to create a binary tree and print traversal orders
class Node:
    def __init__(self,data):
        self.data=data
        self.l=None
        self.r=None
```

```
class BT:
    def __init__(self):
        self.root=None

    def insert(self,n):
        # Write code here
        …

    def postorder(self,root):
        # Write code here
        …

    def preorder(self,root):
        # Write code here
        …

    def inorder(self,root):
        # Write code here
        …

# Driver code
b=BT()
print("******************TREE USING DOUBLE LINKED LIST************************")
while True:
    print("1.Insert data to tree")
    print("2.Post Order Traversal")
    print("3.Pre Order Traversal")
    print("4.In Order Traversal")
    print("5.Exit")
    ch=int(input("Enter choice:"))
    if ch==1:
        n=int(input("Enter number of nodes:"))
        b.insert(n)
    elif ch==2:
        b.postorder(b.root)
    elif ch==3:
        b.preorder(b.root)
    elif ch==4:
        b.inorder(b.root)
    else:
        print("Exit")
        break
```

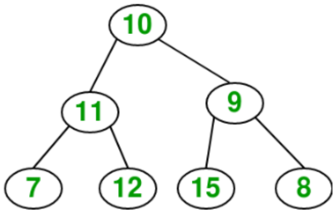
9.3 Insertion in a Binary Tree in Level Order

Given a binary tree and a key, insert the key into the binary tree at the first position available in level order.

Input: Consider the tree given below



Output:



After inserting 12

The idea is to do an iterative level order traversal of the given tree using queue. If we find a node whose left child is empty, we make a new key as the left child of the node. Else if we find a node whose right child is empty, we make the new key as the right child. We keep traversing the tree until we find a node whose either left or right child is empty.

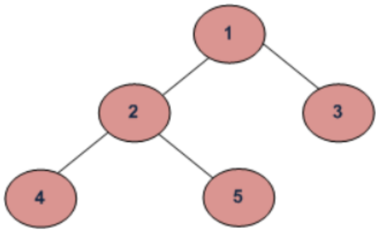
```
# Insert element in binary tree
class newNode():
    def __init__(self, data):
        self.key = data
        self.left = None
        self.right = None

# Inorder traversal of a binary tree
def inorder(temp):
    # Write code here
    …

# function to insert element in binary tree
def insert(temp,key):
   # Write code here
   …
# Driver code
root = newNode(10)
root.left = newNode(11)
root.left.left = newNode(7)
root.right = newNode(9)
root.right.left = newNode(15)
root.right.right = newNode(8)
print("Inorder traversal before insertion:", end = " ")
inorder(root)

key = 12
insert(root, key)

print()
print("Inorder traversal after insertion:", end = " ")
inorder(root)
```

9.4 Finding the Maximum Height or Depth of a Binary Tree

Given a binary tree, the task is to find the height of the tree. The height of the tree is the number of edges in the tree from the root to the deepest node.

Note: The height of an empty tree is 0.

Input: Consider the tree below



Recursively calculate the height of the left and the right subtrees of a node and assign height to the node as max of the heights of two children plus 1.

maxDepth('1') = max(maxDepth('2'), maxDepth('3')) + 1 = 2 + 1
because recursively
maxDepth('2') = max (maxDepth('4'), maxDepth('5')) + 1 = 1 + 1 and (as height of both '4' and '5' are 1)
maxDepth('3') = 1

Procedure:

- Recursively do a Depth-first search.
- If the tree is empty then return 0
- Otherwise, do the following
  - Get the max depth of the left subtree recursively i.e. call maxDepth(tree->left-subtree)
  - Get the max depth of the right subtree recursively i.e. call maxDepth(tree->right-subtree)
  - Get the max of max depths of left and right subtrees and add 1 to it for the current node.

    $$max_depth = max(maxdeptofleftsubtree, maxdepthofrightsubtree) + 1$$
  - Return max_depth.

```
# Find the maximum depth of tree
# A binary tree node
class Node:
    # Constructor to create a new node
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

# Compute the "maxDepth" of a tree -- the number of nodes
# along the longest path from the root node down to the farthest leaf node

def maxDepth(node):
    # Write code here
    …

# Driver program to test above function
root = Node(1)
```

```
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
print("Height of tree is %d" % (maxDepth(root)))
```

9.5 Deletion in a Binary Tree

Given a binary tree, delete a node from it by making sure that the tree shrinks from the bottom (i.e. the deleted node is replaced by the bottom-most and rightmost node).

Input: Delete 10 in below tree

```
      10
    /   \
 20     30
```

Output:

```
      30
    /
 20
```

Input: Delete 20 in below tree

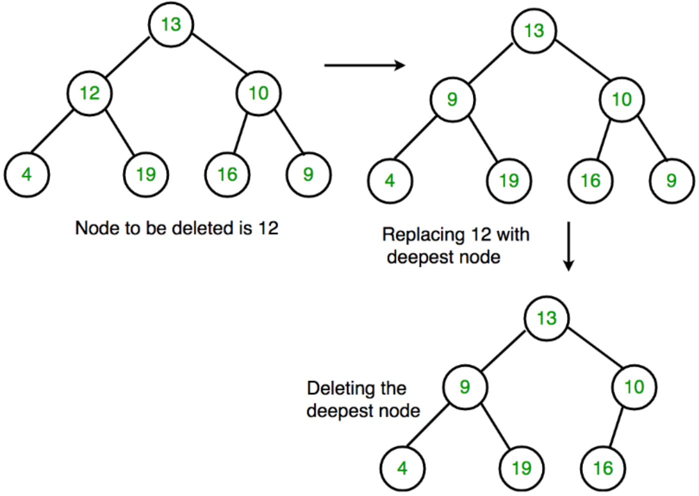
```
      10
    /   \
 20     30
          \
           40
```

Output:

```
      10
  /     \
40       30
```

Algorithm:

1. Starting at the root, find the deepest and rightmost node in the binary tree and the node which we want to delete.
2. Replace the deepest rightmost node's data with the node to be deleted.
3. Then delete the deepest rightmost node.



```
# Deletion in a Binary Tree

# Create a node with data, left child and right child.

class Node:
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

# Inorder traversal of a binary tree
def inorder(temp):
    # Write code here
    …

# function to delete the given deepest node (d_node) in binary tree
def deleteDeepest(root, d_node):
    # Write code here
    …

# function to delete element in binary tree
def deletion(root, key):
    # Write code here
    …

# Driver code
root = Node(10)
root.left = Node(11)
root.left.left = Node(7)
root.left.right = Node(12)
root.right = Node(9)
root.right.left = Node(15)
root.right.right = Node(8)
print("The tree before the deletion: ", end = "")
inorder(root)
key = 11
root = deletion(root, key)
print();
print("The tree after the deletion: ", end = "")
inorder(root)
```

10. Binary Search Tree (BST)

10.1 Searching in Binary Search Tree

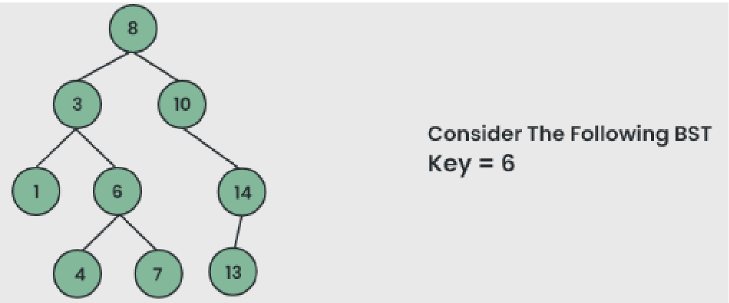
Given a BST, the task is to delete a node in this BST. For searching a value in BST, consider it as a sorted array. Perform search operation in BST using Binary Search Algorithm.

Algorithm to search for a key in a given Binary Search Tree:

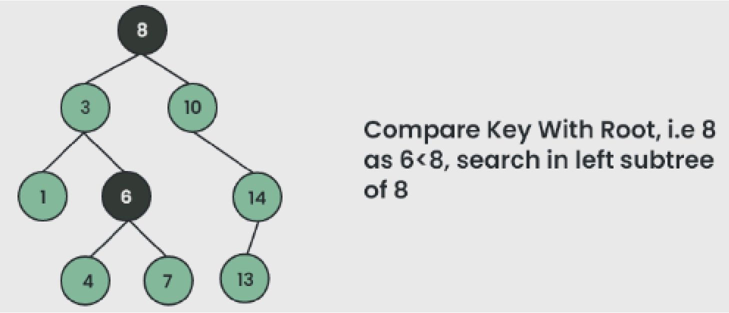
Let's say we want to search for the number **X,** We start at the root. Then:

- We compare the value to be searched with the value of the root.
- If it's equal we are done with the search if it's smaller we know that we need to go to the left subtree because in a binary search tree all the elements in the left subtree are smaller and all the elements in the right subtree are larger.
- Repeat the above step till no more traversal is possible

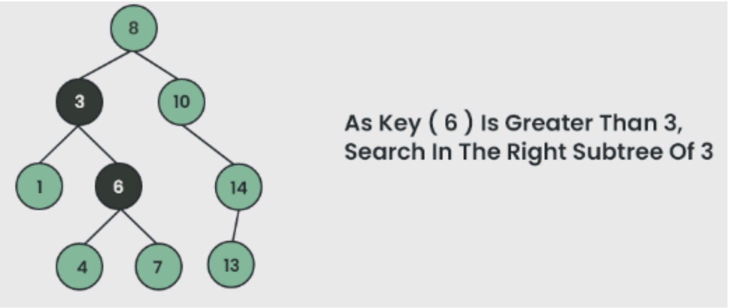
- If at any iteration, key is found, return True. Else False.



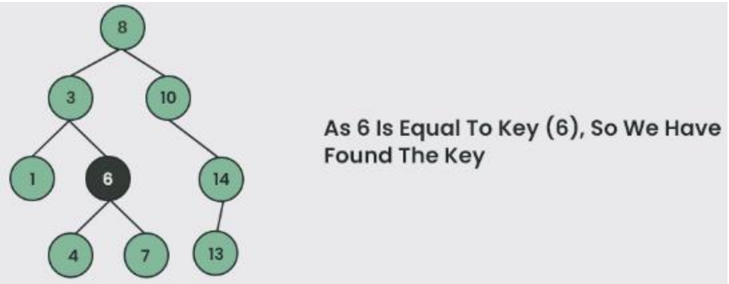
Consider The Following BST
Key = 6



Compare Key With Root, i.e 8
as 6<8, search in left subtree
of 8



As Key (6) Is Greater Than 3,
Search In The Right Subtree Of 3



As 6 Is Equal To Key (6), So We Have
Found The Key

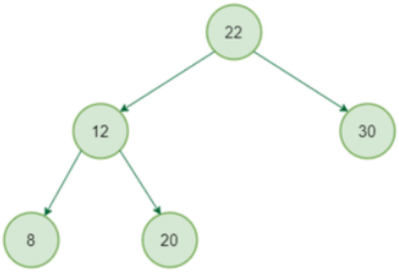
```
# Search a given key in a given BST

class Node:
    # Constructor to create a new node
    def __init__(self, key):
        self.key = key
        self.left = None
        self.right = None

# A utility function to insert
# a new node with the given key in BST
```

```
def insert(node, key):
    # Write code here
    …

# Utility function to search a key in a BST
def search(root, key):
    # Write code here
    …

# Driver Code
root = None
root = insert(root, 50)
insert(root, 30)
insert(root, 20)
insert(root, 40)
insert(root, 70)
insert(root, 60)
insert(root, 80)
# Key to be found
key = 6

# Searching in a BST
if search(root, key) is None:
    print(key, "not found")

else:
    print(key, "found")

key = 60

# Searching in a BST
if search(root, key) is None:
    print(key, "not found")
else:
    print(key, "found")
```

10.2 Find the node with Minimum Value in a BST

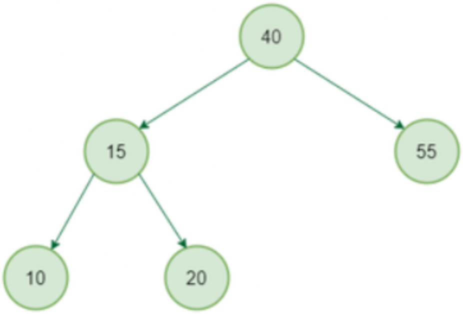
Write a function to find the node with minimum value in a Binary Search Tree.

Input: Consider the tree given below



Output: 8

Input: Consider the tree given below



Output: 10

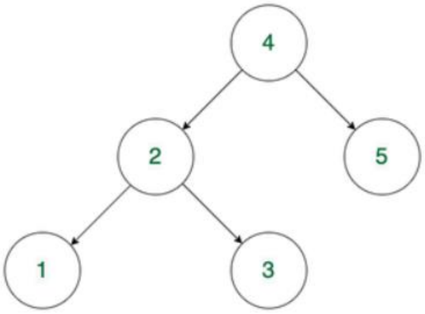
```
from typing import List
class Node:
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

# Give a binary search tree and a number, inserts a new node with the given number
# in the correct place in the tree. Returns the new root pointer
def insert(node: Node, data: int) -> Node:
    # Write code here
    …
# Given a non-empty binary search tree, inorder traversal for
# the tree is stored in the list sorted_inorder. Inorder is LEFT, ROOT, RIGHT.

def inorder(node: Node, sorted_inorder: List[int]) -> None:
    # Write code here
    …
# Driver Code
root = None
root = insert(root, 4)
insert(root, 2)
insert(root, 1)
insert(root, 3)
insert(root, 6)
insert(root, 4)
insert(root, 5)
sorted_inorder = []
inorder(root, sorted_inorder)  # calling the recursive function

# Values of all nodes will appear in sorted order in the list sorted_inorder
print(f"Minimum value in BST is {sorted_inorder[0]}")
```

10.3 Check if a Binary Tree is BST or not

A binary search tree (BST) is a node-based binary tree data structure that has the following properties.

1. The left subtree of a node contains only nodes with keys less than the node’s key.
2. The right subtree of a node contains only nodes with keys greater than the node’s key.
3. Both the left and right subtrees must also be binary search trees.
4. Each node (item in the tree) has a distinct key.

Input: Consider the tree given below



Output: Check if max value in left subtree is smaller than the node and min value in right subtree greater than the node, then print it "Is BST" otherwise "Not a BST"

Procedure:

1. If the current node is null then return true
2. If the value of the left child of the node is greater than or equal to the current node then return false
3. If the value of the right child of the node is less than or equal to the current node then return false
4. If the left subtree or the right subtree is not a BST then return false
5. Else return true

```
# Program to check if a binary tree is BST or not
# A binary tree node has data, pointer to left child and a pointer to right child

class Node:
     def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

def maxValue(node):
    # Write code here
    …

def minValue(node):
    # Write code here
    …
# Returns true if a binary tree is a binary search tree
def isBST(node):
    # Write code here
    …
# Driver code
root = Node(4)
root.left = Node(2)
root.right = Node(5)
# root.right.left = Node(7)
root.left.left = Node(1)
root.left.right = Node(3)

# Function call
if isBST(root) is True:
    print("Is BST")
else:
    print("Not a BST")
```

10.4 Second Largest Element in BST

Given a Binary search tree (BST), find the second largest element.

Input: Root of below BST

```
          10
         /
       5
```

Output: 5

Input: Root of below BST

```
         10

        /  \

     5     20

             \

              30
```

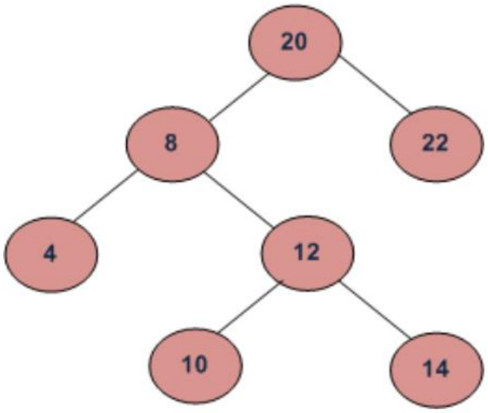
Output: 20

Procedure: The second largest element is second last element in inorder traversal and second element in reverse inorder traversal. We traverse given Binary Search Tree in reverse inorder and keep track of counts of nodes visited. Once the count becomes 2, we print the node.

```
# Find the second largest element in
class Node:

    # Constructor to create a new node
    def __init__(self, data):
        self.key = data
        self.left = None
        self.right = None

# A function to find 2nd largest element in a given tree.
def secondLargestUtil(root, c):
    # Write code here
    …
# Function to find 2nd largest element
def secondLargest(root):
    # Write code here
    …

# A utility function to insert a new node with given key in BST
def insert(node, key):

# Driver Code
# Let us create following BST
#       50
#     /    \
#    30     70
#   / \     / \
#  20  40  60  80

root = None
root = insert(root, 50)
insert(root, 30)
insert(root, 20)
insert(root, 40)
insert(root, 70)
insert(root, 60)
insert(root, 80)
secondLargest(root)
```

Try:

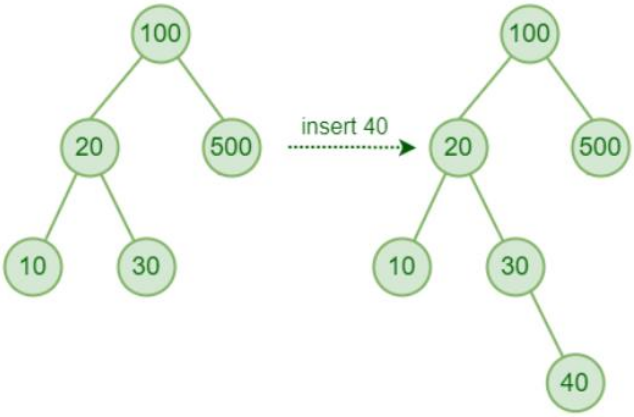
1. **Kth largest element in BST when modification to BST is not allowed:** Given a Binary Search Tree (BST) and a positive integer k, find the k'th largest element in the Binary Search Tree. For a given BST, if k = 3, then output should be 14, and if k = 5, then output should be 10.



10.5 Insertion in Binary Search Tree (BST)

Given a Binary search tree (BST), the task is to insert a new node in this BST.

Input: Consider a BST and insert the element 40 into it.



Procedure for inserting a value in a BST:

A new key is always inserted at the leaf by maintaining the property of the binary search tree. We start searching for a key from the root until we hit a leaf node. Once a leaf node is found, the new node is added as a child of the leaf node. The below steps are followed while we try to insert a node into a binary search tree:

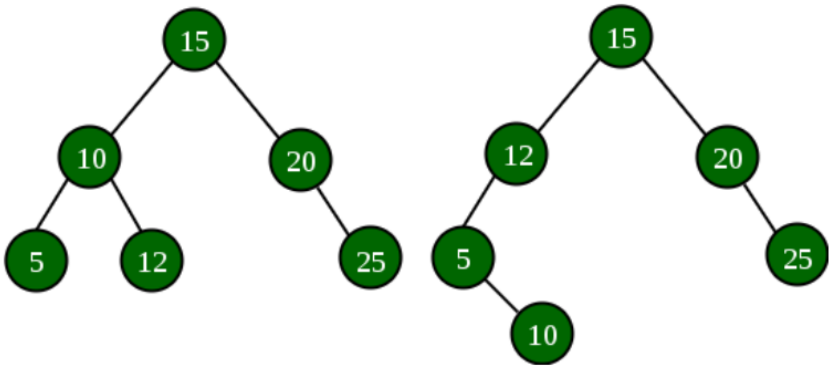
- Check the value to be inserted (say X) with the value of the current node (say val) we are in:
  - If X is less than val move to the left subtree.
  - Otherwise, move to the right subtree.
  - Once the leaf node is reached, insert X to its right or left based on the relation between X and the leaf node's value.

```
# insert operation in binary search tree
# A utility class that represents an individual node in a BST
class Node:
    def __init__(self, key):
        self.left = None
        self.right = None
        self.val = key

 # A utility function to insert a new node with the given key
def insert(root, key):
    # Write code here
    …

 # A utility function to do inorder tree traversal
def inorder(root):
    # Write code here
    …

# Driver code
# Let us create the following BST
#       50
#     /     \
#    30     70
#   / \     / \
#  20 40   60 80
r = Node(50)
r = insert(r, 30)
r = insert(r, 20)
r = insert(r, 40)
r = insert(r, 70)
r = insert(r, 60)
r = insert(r, 80)

# Print inorder traversal of the BST
inorder(r)
```

Try:

1. **Check if two BSTs contain same set of elements:** Given two Binary Search Trees consisting of unique positive elements, we have to check whether the two BSTs contain the same set of elements or not.

Input: Consider two BSTs which contains same set of elements {5, 10, 12, 15, 20, 25}, but the structure of the two given BSTs can be different.



11. AVL Tree

11.1 Insertion in an AVL Tree

AVL tree is a self-balancing Binary Search Tree (BST) where the difference between heights of left and right subtrees cannot be more than one for all nodes. To make sure that the given tree remains AVL after every insertion, we must augment the standard BST insert operation to perform some re-balancing. Following are two basic operations that can be performed to balance a BST without violating the BST property (keys(left) < key(root) < keys(right)).

- Left Rotation
- Right Rotation

T1, T2 and T3 are subtrees of the tree, rooted with y (on the left side) or x (on the right side)

```
    y                               x
   / \     Right Rotation          /  \
  x   T3   - - - - - - - >        T1   y
 / \       < - - - - - - -            / \
T1  T2     Left Rotation            T2  T3
```

Keys in both of the above trees follow the following order

keys(T1) < key(x) < keys(T2) < key(y) < keys(T3)

So BST property is not violated anywhere.

Procedure for inserting a node into an AVL tree
Let the newly inserted node be w

- Perform standard BST insert for w.
- Starting from w, travel up and find the first unbalanced node. Let z be the first unbalanced node, y be the child of z that comes on the path from w to z and x be the grandchild of z that comes on the path from w to z.

- Re-balance the tree by performing appropriate rotations on the subtree rooted with z. There can be 4 possible cases that need to be handled as x, y and z can be arranged in 4 ways.
- Following are the possible 4 arrangements:
    - y is the left child of z and x is the left child of y (Left Left Case)
    - y is the left child of z and x is the right child of y (Left Right Case)
    - y is the right child of z and x is the right child of y (Right Right Case)
    - y is the right child of z and x is the left child of y (Right Left Case)

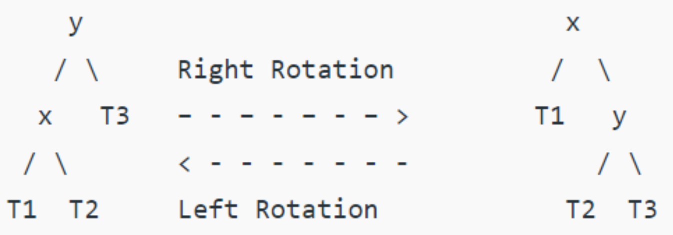
```
# Insert a node in AVL tree

# Generic tree node class
class TreeNode(object):
    def __init__(self, val):
        self.val = val
        self.left = None
        self.right = None
        self.height = 1

# AVL tree class which supports the insert operation
class AVL_Tree(object):

    # Recursive function to insert key in subtree rooted with node and returns
    # new root of subtree.
    def insert(self, root, key):
        # Write code here
        …

    def leftRotate(self, z):
        # Write code here
        …

    def rightRotate(self, z):
        # Write code here
        …

    def getHeight(self, root):
        # Write code here
        …

    def getBalance(self, root):
        # Write code here
        …

    def preOrder(self, root):
        # Write code here
        …
# Driver code
myTree = AVL_Tree()
root = None

root = myTree.insert(root, 10)
root = myTree.insert(root, 20)
root = myTree.insert(root, 30)
root = myTree.insert(root, 40)
root = myTree.insert(root, 50)
root = myTree.insert(root, 25)

"""The constructed AVL Tree would be
            30
           /  \
         20    40
        /  \     \
       10   25    50"""

# Preorder Traversal
print("Preorder traversal of the",
      "constructed AVL tree is")
myTree.preOrder(root)
print()
```

11.2 Deletion in an AVL Tree

Given an AVL tree, make sure that the given tree remains AVL after every deletion, we must augment the standard BST delete operation to perform some re-balancing. Following are two basic operations that can be performed to re-balance a BST without violating the BST property (keys(left) < key(root) < keys(right)).

1. Left Rotation
2. Right Rotation

T1, T2 and T3 are subtrees of the tree rooted with y (on left side)
or x (on right side)



```
     y                               x
    / \     Right Rotation          /  \
   x   T3   - - - - - - - >        T1   y
  / \       < - - - - - - -            / \
 T1  T2     Left Rotation            T2  T3
```

Keys in both of the above trees follow the following order
    keys(T1) < key(x) < keys(T2) < key(y) < keys(T3)
So BST property is not violated anywhere.

Procedure to delete a node from AVL tree:

Let w be the node to be deleted

1. Perform standard BST delete for w.
2. Starting from w, travel up and find the first unbalanced node. Let z be the first unbalanced node, y be the larger height child of z, and x be the larger height child of y. Note that the definitions of x and y are different from insertion here.
3. Re-balance the tree by performing appropriate rotations on the subtree rooted with z. There can be 4 possible cases that needs to be handled as x, y and z can be arranged in 4 ways. Following are the possible 4 arrangements:
   i. y is left child of z and x is left child of y (Left Left Case)
   ii. y is left child of z and x is right child of y (Left Right Case)
   iii. y is right child of z and x is right child of y (Right Right Case)
   iv. y is right child of z and x is left child of y (Right Left Case)

```
# delete a node in AVL tree

class TreeNode(object):
    def __init__(self, val):
        self.val = val
        self.left = None
        self.right = None
        self.height = 1

# AVL tree class which supports insertion, deletion operations
class AVL_Tree(object):

    def insert(self, root, key):
        # Write code here
        …

    # Recursive function to delete a node with given key from subtree
    # with given root. It returns root of the modified subtree.
    def delete(self, root, key):
        # Write code here
        …

    def leftRotate(self, z):
        # Write code here
        …

    def rightRotate(self, z):
        # Write code here
        …

    def getHeight(self, root):
        # Write code here
        …

    def getBalance(self, root):
        # Write code here
        …

    def getMinValueNode(self, root):
        # Write code here
        …

    def preOrder(self, root):
        # Write code here
        …

myTree = AVL_Tree()
root = None
nums = [9, 5, 10, 0, 6, 11, -1, 1, 2]

for num in nums:
    root = myTree.insert(root, num)


# Preorder Traversal
print("Preorder Traversal after insertion -")
myTree.preOrder(root)
print()
```

```
# Delete
key = 10
root = myTree.delete(root, key)

# Preorder Traversal
print("Preorder Traversal after deletion -")
myTree.preOrder(root)
print()
```

11.3 Count Greater Nodes in AVL Tree

Given an AVL tree, calculate number of elements which are greater than given value in AVL tree.

Input: x = 5

Root of below AVL tree

```
        9
       /  \
      1     10
     /  \     \
     0    5     11
    /   /   \
-1      2     6
```

Output: 4

Explanation: There are 4 values which are greater than 5 in AVL tree which are 6, 9, 10 and 11.

```
# Count greater nodes in an AVL tree

class Node:
    def __init__(self, key):
        self.key = key
        self.left = None
        self.right = None
        self.height = 1
        self.desc = 0

 def height(N):
    if N is None:
        return 0
    return N.height

# A utility function to get maximum of two integers

def max(a, b):
    if a > b:
        return a
    return b

def newNode(key):
    # Write code here
    …
```

```
# A utility function to right rotate subtree rooted with y

def rightRotate(y):
    # Write code here
    …

def leftRotate(x):
    # Write code here
    …
def getBalance(N):
    # Write code here
    …

def insert(root, key):
    # Write code here
    …

def minValueNode(node):
    # Write code here
    …
# Recursive function to delete a node with given key # from subtree with given root. It
returns root of the modified subtree.

def deleteNode(root, key):
    # Write code here
    …

def preOrder(root):
    # Write code here
    …

def CountGreater(root, x):
    # Write code here
    …
# Driver program to test above function
root = None
root = insert(root, 9)
root = insert(root, 5)
root = insert(root, 10)
root = insert(root, 0)
root = insert(root, 6)
root = insert(root, 11)
root = insert(root, -1)
root = insert(root, 1)
root = insert(root, 2)

print("Preorder traversal of the constructed AVL tree is")
preOrder(root)

print("Number of elements greater than 9 are")
print(CountGreater(root, 9))

root = deleteNode(root, 10)

print("Preorder traversal after deletion of 10")
preOrder(root)
print('Number of elements greater than 9 are')
print(CountGreater(root, 9))
```

11.4 Minimum Number of Nodes in an AVL Tree with given Height

Given the height of an AVL tree ‘h’, the task is to find the minimum number of nodes the tree can have.

Input: H = 0

Output: N = 1

Only '1' node is possible if the height
of the tree is '0' which is the root node.

Input: H = 3

Output: N = 7

Recursive approach:
In an AVL tree, we have to maintain the height balance property, i.e. difference in the height of the left and the right subtrees cannot be other than -1, 0 or 1 for each node.

We will try to create a recurrence relation to find minimum number of nodes for a given height, n(h).

- For height = 0, we can only have a single node in an AVL tree, i.e. n(0) = 1
- For height = 1, we can have a minimum of two nodes in an AVL tree, i.e. n(1) = 2
- Now for any height ‘h’, root will have two subtrees (left and right). Out of which one has to be of height h-1 and other of h-2. [root node excluded]
- So, n(h) = 1 + n(h-1) + n(h-2) is the required recurrence relation for h>=2 [1 is added for the root node]

```
# Function to find minimum number of nodes

def AVLnodes(height):
    # Write code here
    …
# Driver Code
H = 3
print(AVLnodes(H))
```

12. Graph Traversal

12.1 Breadth First Search

The **Breadth First Search (BFS)** algorithm is used to search a graph data structure for a node that meets a set of criteria. It starts at the root of the graph and visits all nodes at the current depth level before moving on to the nodes at the next depth level.
For a given graph G, print BFS traversal from a given source vertex.

```
# BFS traversal from a given source vertex.

from collections import defaultdict
 # This class represents a directed graph using adjacency list representation
class Graph:
     # Constructor
    def __init__(self):
        # Default dictionary to store graph
        self.graph = defaultdict(list)
    # Function to add an edge to graph
    def addEdge(self, u, v):
        self.graph[u].append(v)

    # Function to print a BFS of graph
    def BFS(self, s):
      # Write code here
      …

# Create a graph given in the above diagram
g = Graph()
g.addEdge(0, 1)
g.addEdge(0, 2)
g.addEdge(1, 2)
g.addEdge(2, 0)
g.addEdge(2, 3)
g.addEdge(3, 3)

print("Following is Breadth First Traversal" " (starting from vertex 2)")
g.BFS(2)
```

Output: Following is Breadth First Traversal (starting from vertex 2)
2 0 3 1

12.2 Depth First Search

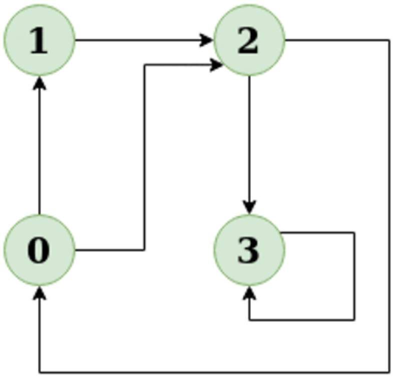
Depth First Traversal (or DFS) for a graph is similar to Depth First Traversal of a tree. The only catch here is, that, unlike trees, graphs may contain cycles (a node may be visited twice). To avoid processing a node more than once, use a boolean visited array. A graph can have more than one DFS traversal.

For a given graph G, print DFS traversal from a given source vertex.

Input: n = 4, e = 6
0 -> 1, 0 -> 2, 1 -> 2, 2 -> 0, 2 -> 3, 3 -> 3

Output: DFS from vertex 1: 1 2 0 3

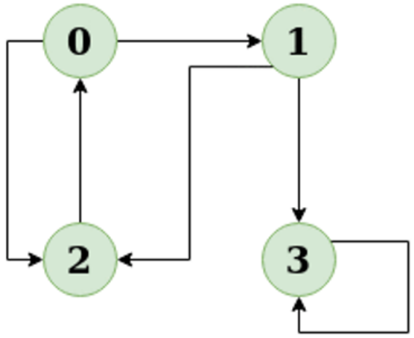
Explanation:
DFS Diagram:



Input: n = 4, e = 6
2 -> 0, 0 -> 2, 1 -> 2, 0 -> 1, 3 -> 3, 1 -> 3

Output: DFS from vertex 2: 2 0 1 3

Explanation:
DFS Diagram:



```
# DFS traversal from a given graph
from collections import defaultdict

# This class represents a directed graph using adjacency list representation
class Graph:
    # Constructor
    def __init__(self):
        # Default dictionary to store graph
        self.graph = defaultdict(list)

    # Function to add an edge to graph
    def addEdge(self, u, v):
        self.graph[u].append(v)
    # A function used by DFS
    def DFSUtil(self, v, visited):
        # Write code here
        …

    # The function to do DFS traversal. It uses recursive DFSUtil()

    def DFS(self, v):
      # Write code here
       …
# Driver's code
g = Graph()
g.addEdge(0, 1)
g.addEdge(0, 2)
g.addEdge(1, 2)
g.addEdge(2, 0)
g.addEdge(2, 3)
g.addEdge(3, 3)
print("Following is Depth First Traversal (starting from vertex 2)")
# Function call
g.DFS(2)
```

12.3 Best First Search (Informed Search)

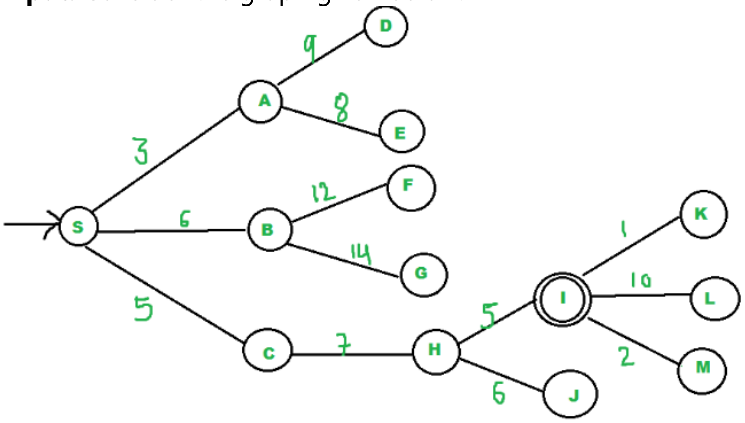
The idea of Best First Search is to use an evaluation function to decide which adjacent is most promising and then explore. Best First Search falls under the category of Heuristic Search or Informed Search.

Implementation of Best First Search:
We use a priority queue or heap to store the costs of nodes that have the lowest evaluation function value. So the implementation is a variation of BFS, we just need to change Queue to PriorityQueue.

Algorithm:
Best-First-Search(Graph g, Node start)

```
  1) Create an empty PriorityQueue
     PriorityQueue pq;
  2) Insert "start" in pq.
     pq.insert(start)
  3) Until PriorityQueue is empty
       u = PriorityQueue.DeleteMin
       If u is the goal
          Exit
       Else
          Foreach neighbor v of u
             If v "Unvisited"
                 Mark v "Visited"
                 pq.insert(v)
          Mark u "Examined"
End procedure
```

Input: Consider the graph given below.



- We start from source "S" and search for goal "I" using given costs and Best First search.
- pq initially contains S
  - We remove S from pq and process unvisited neighbors of S to pq.
  - pq now contains {A, C, B} (C is put before B because C has lesser cost)
- We remove A from pq and process unvisited neighbors of A to pq.
  - pq now contains {C, B, E, D}
- We remove C from pq and process unvisited neighbors of C to pq.
  - pq now contains {B, H, E, D}
- We remove B from pq and process unvisited neighbors of B to pq.
  - pq now contains {H, E, D, F, G}
- We remove H from pq.

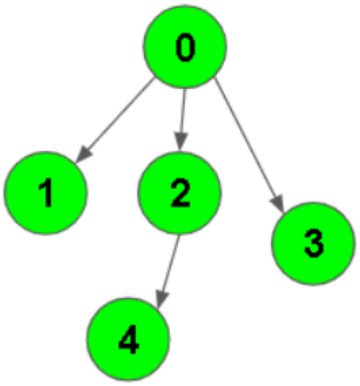
- Since our goal “I” is a neighbor of H, we return.

```
from queue import PriorityQueue
v = 14
graph = [[] for i in range(v)]

# Function For Implementing Best First Search
# Gives output path having lowest cost

def best_first_search(actual_Src, target, n):
    # Write code here
    …
# Function for adding edges to graph
def addedge(x, y, cost):
    # Write code here
    …
# The nodes shown in above example(by alphabets) are
# implemented using integers addedge(x,y,cost);
addedge(0, 1, 3)
addedge(0, 2, 6)
addedge(0, 3, 5)
addedge(1, 4, 9)
addedge(1, 5, 8)
addedge(2, 6, 12)
addedge(2, 7, 14)
addedge(3, 8, 7)
addedge(8, 9, 5)
addedge(8, 10, 6)
addedge(9, 11, 1)
addedge(9, 12, 10)
addedge(9, 13, 2)

source = 0
target = 9
best_first_search(source, target, v)
```

12.4 Breadth First Traversal of a Graph

Given a directed graph. The task is to do Breadth First Traversal of this graph starting from 0.
One can move from node u to node v only if there's an edge from u to v. Find the BFS traversal of the graph starting from the 0th vertex, from left to right according to the input graph. Also, you should only take nodes directly or indirectly connected from Node 0 in consideration.

Input: Consider the graph given below where V = 5, E = 4, edges = {(0,1), (0,2), (0,3), (2,4)}



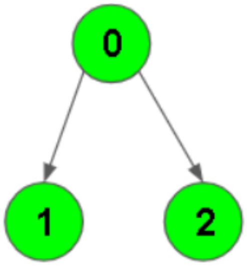
Output: 0 1 2 3 4

Explanation:

0 is connected to 1, 2, and 3.

2 is connected to 4.
So starting from 0, it will go to 1 then 2 then 3. After this 2 to 4, thus BFS will be 0 1 2 3 4.

Input: Consider the graph given below where V = 3, E = 2, edges = {(0, 1), (0, 2)}



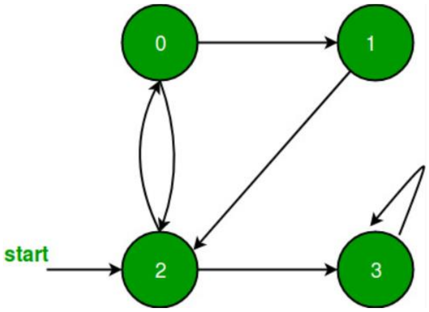
Output: 0 1 2
Explanation:
0 is connected to 1, 2. So starting from 0, it will go to 1 then 2, thus BFS will be 0 1 2.
Your task is to complete the function **bfsOfGraph()** which takes the integer V denoting the number of vertices and adjacency list as input parameters and returns a list containing the BFS traversal of the graph starting from the 0th vertex from left to right.

```
from typing import List
from queue import Queue
class Solution:

    # Function to return Breadth First Traversal of given graph.
    def bfsOfGraph(self, V: int, adj: List[List[int]]) -> List[int]:
        # Write code here
        …
# Driver Code
T=int(input())
for i in range(T):
       V, E = map(int, input().split())
       adj = [[] for i in range(V)]
       for _ in range(E):
             u, v = map(int, input().split())
             adj[u].append(v)
       ob = Solution()
       ans = ob.bfsOfGraph(V, adj)
       for i in range(len(ans)):
                print(ans[i], end = " ")
       print()
```

12.5 Depth First Search (DFS) for Disconnected Graph

Given a Disconnected Graph, the task is to implement DFS or Depth First Search Algorithm for this Disconnected Graph.

Input: Consider the graph given below.



Output: 0 1 2 3

Procedure for DFS on Disconnected Graph:
Iterate over all the vertices of the graph and for any unvisited vertex, run a DFS from that vertex.

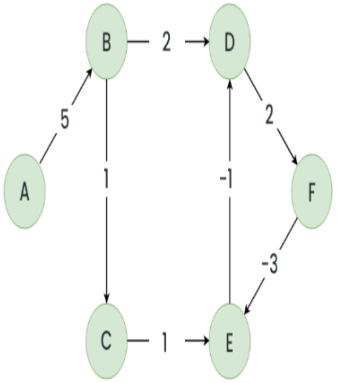
```
# DFS traversal for complete graph
from collections import defaultdict

# This class represents a directed graph using adjacency list representation
class Graph:
    # Constructor
    def __init__(self):
        # Default dictionary to store graph
        self.graph = defaultdict(list)

    # Function to add an edge to graph
    def addEdge(self, u, v):
        # Write code here
        …

    # A function used by DFS
    def DFSUtil(self, v, visited):
        # Write code here
        …

    # The function to do DFS traversal.
    # It uses recursive DFSUtil
    def DFS(self):
        # Write code here
        …
# Driver's code
print("Following is Depth First Traversal")
g = Graph()
g.addEdge(0, 1)
g.addEdge(0, 2)
g.addEdge(1, 2)
g.addEdge(2, 0)
g.addEdge(2, 3)
g.addEdge(3, 3)

# Function call
g.DFS()
```

Try:

1. **Detect a negative cycle in a Graph (Bellman Ford):** A Bellman-Ford algorithm is also guaranteed to find the shortest path in a graph, similar to Dijkstra's algorithm. Although Bellman-Ford is slower than Dijkstra's algorithm, it is capable of handling graphs with negative edge weights, which makes it more versatile. The

shortest path cannot be found if there exists a negative cycle in the graph. If we continue to go around the negative cycle an infinite number of times, then the cost of the path will continue to decrease (even though the length of the path is increasing).

Consider a graph G and detect a negative cycle in the graph using Bellman Ford algorithm.



13. Minimum Spanning Tree (MST)

13.1 Kruskal's Algorithm

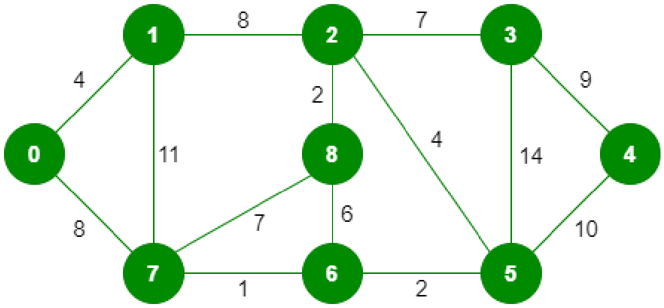
In Kruskal's algorithm, sort all edges of the given graph in increasing order. Then it keeps on adding new edges and nodes in the MST if the newly added edge does not form a cycle. It picks the minimum weighted edge at first and the maximum weighted edge at last.

MST using Kruskal's algorithm:

1. Sort all the edges in non-decreasing order of their weight.
2. Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If the cycle is not formed, include this edge. Else, discard it.
3. Repeat step#2 until there are (V-1) edges in the spanning tree.

Kruskal's algorithm to find the minimum cost spanning tree uses the greedy approach. The Greedy Choice is to pick the smallest weight edge that does not cause a cycle in the MST constructed so far.

Input: For the given graph G find the minimum cost spanning tree.



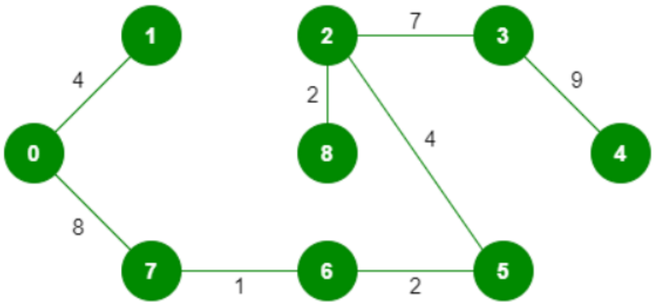
The graph contains 9 vertices and 14 edges. So, the minimum spanning tree formed will be having (9 – 1) = 8 edges.

After sorting:

Weight	Source	Destination
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

Now pick all edges one by one from the sorted list of edges.

Output:



```
# Kruskal's algorithm to find minimum Spanning Tree of a given connected,
# undirected and weighted graph

# Class to represent a graph
class Graph:
    def __init__(self, vertices):
        self.V = vertices
        self.graph = []

    # Function to add an edge to graph
    def addEdge(self, u, v, w):
        self.graph.append([u, v, w])

    def find(self, parent, i):
        …

    def union(self, parent, rank, x, y):
        …

    def KruskalMST(self):
      # write your code here
      …
```

```
# Driver code
g = Graph(4)
g.addEdge(0, 1, 10)
g.addEdge(0, 2, 6)
g.addEdge(0, 3, 5)
g.addEdge(1, 3, 15)
g.addEdge(2, 3, 4)

# Function call
g.KruskalMST()
```

Output: Following are the edges in the constructed MST

```
2 -- 3 == 4

0 -- 3 == 5

0 -- 1 == 10
```

Minimum Cost Spanning Tree: 19

13.2 Prim's Algorithm

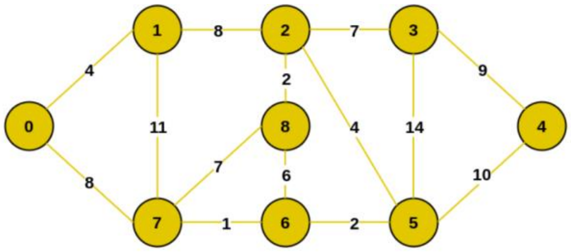
The Prim's algorithm starts with an empty spanning tree. The idea is to maintain two sets of vertices. The first set contains the vertices already included in the MST, and the other set contains the vertices not yet included. At every step, it considers all the edges that connect the two sets and picks the minimum weight edge from these edges. After picking the edge, it moves the other endpoint of the edge to the set containing MST.

Prim's Algorithm:

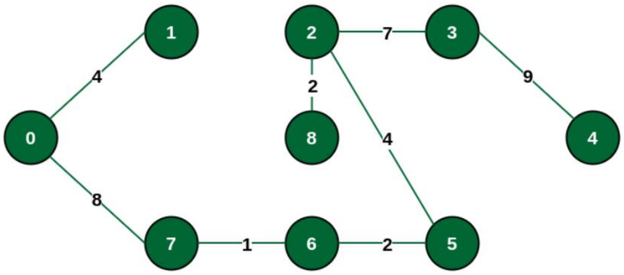
The working of Prim's algorithm can be described by using the following steps:

1. Determine an arbitrary vertex as the starting vertex of the MST.
2. Follow steps 3 to 5 till there are vertices that are not included in the MST (known as fringe vertex).
3. Find edges connecting any tree vertex with the fringe vertices.
4. Find the minimum among these edges.
5. Add the chosen edge to the MST if it does not form any cycle.
6. Return the MST and exit

Input: For the given graph G find the minimum cost spanning tree.



Output: The final structure of the MST is as follows and the weight of the edges of the MST is (4 + 8 + 1 + 2 + 4 + 2 + 7 + 9) = 37.



```
# Prim's Minimum Spanning Tree (MST) algorithm.
# The program is for adjacency matrix representation of the graph

# Library for INT_MAX
import sys

class Graph():
    def __init__(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)]
                      for row in range(vertices)]

    # A utility function to print
    # the constructed MST stored in parent[]
    def printMST(self, parent):
        print("Edge \tWeight")
        for i in range(1, self.V):
            print(parent[i], "-", i, "\t", self.graph[i][parent[i]])

    # A utility function to find the vertex with
    # minimum distance value, from the set of vertices
    # not yet included in shortest path tree
    def minKey(self, key, mstSet):
       # write your code here
       …

    def primMST(self):
       # write your code here
       …
# Driver's code
g = Graph(5)
g.graph = [[0, 2, 0, 6, 0],
           [2, 0, 3, 8, 5],
           [0, 3, 0, 0, 7],
           [6, 8, 0, 0, 9],
           [0, 5, 7, 9, 0]]

g.primMST()
```

Output:

```
Edge    Weight

0 - 1     2

1 - 2     3

0 - 3     6

1 - 4     5
```

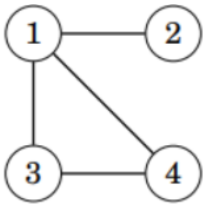
13.3 Total Number of Spanning Trees in a Graph

If a graph is a complete graph with n vertices, then total number of spanning trees is $n^{(n-2)}$ where n is the number of nodes in the graph. In complete graph, the task is equal to counting different labeled trees with n nodes for which have Cayley's formula.

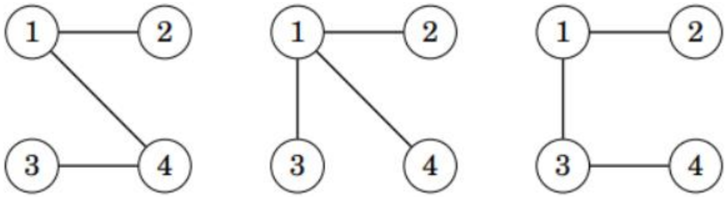
Laplacian matrix:

A Laplacian matrix L, where L[i, i] is the degree of node i and L[i, j] = −1 if there is an edge between nodes i and j, and otherwise L[i, j] = 0.

Kirchhoff's theorem provides a way to calculate the number of spanning trees for a given graph as a determinant of a special matrix. Consider the following graph,



All possible spanning trees are as follows:



In order to calculate the number of spanning trees, construct a Laplacian matrix L, where L[i, i] is the degree of node i and L[i, j] = −1 if there is an edge between nodes i and j, and otherwise L[i, j] = 0.
for the above graph, The Laplacian matrix will look like this

$$L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

The number of spanning trees equals the determinant of a matrix.

The Determinant of a matrix that can be obtained when we remove any row and any column from L.
For example, if we remove the first row and column, the result will be,

$$\det(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}) = 3.$$

The determinant is always the same, regardless of which row and column we remove from L.

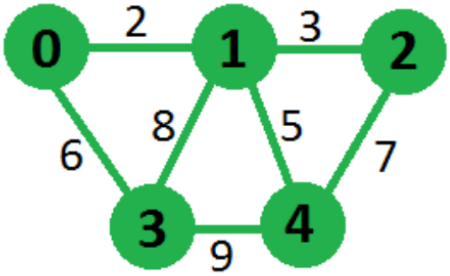
```
# Finds the number of spanning trees in a graph using Matrix Chain Multiplication.
MAX = 100
MOD = 1000000007
```

```
# Matrix Multiplication
def multiply(A, B, C):
    # write your code here
    …

# Function to find Nth power of A
def power(A, N, result):
    # write your code here
    …
# Function to find number of Spanning Trees in a Graph
# using Matrix Chain Multiplication.
def numOfSpanningTree(graph, V):
    # write your code here
    …

# Driver program
V = 4 # Number of vertices in graph
E = 5 # Number of edges in graph
graph = [[0, 1, 1, 1],
         [1, 0, 1, 1],
         [1, 1, 0, 1],
         [1, 1, 1, 0]]
print(numOfSpanningTree(graph, V))
```

13.4 Minimum Product Spanning Tree

A minimum product spanning tree for a weighted, connected, and undirected graph is a spanning tree with a weight product less than or equal to the weight product of every other spanning tree. The weight product of a spanning tree is the product of weights corresponding to each edge of the spanning tree. All weights of the given graph will be positive for simplicity.

Input:



Output: Minimum Product that we can obtain is 180 for above graph by choosing edges 0-1, 1-2, 0-3 and 1-4

This problem can be solved using standard minimum spanning tree algorithms like Kruskal and prim's algorithm, but we need to modify our graph to use these algorithms. Minimum spanning tree algorithms tries to minimize the total sum of weights, here we need to minimize the total product of weights. We can use the property of logarithms to overcome this problem.

log(w1* w2 * w3 * …. * wN) = log(w1) + log(w2) + log(w3) ….. + log(wN)

We can replace each weight of the graph by its log value, then we apply any minimum spanning tree algorithm which will try to minimize the sum of log(wi) which in turn minimizes the weight product.

```
# Minimum product spanning tree
import math
```

```
# Number of vertices in the graph
V = 5

# A utility function to find the vertex with minimum key value, from the set
# of vertices not yet included in MST
def minKey(key, mstSet):
    # write your code here
    …

# A utility function to print the constructed MST stored in parent[] and
# print Minimum Obtainable product
def printMST(parent, n, graph):
    # write your code here
    …

# Function to construct and print MST for a graph represented using adjacency
# matrix representation inputGraph is sent for printing actual edges and
# logGraph is sent for actual MST operations
def primMST(inputGraph, logGraph):
    # write your code here
    …

# Method to get minimum product spanning tree
def minimumProductMST(graph):
    # write your code here
    …

# Driver code
graph = [ [ 0, 2, 0, 6, 0 ],
          [ 2, 0, 3, 8, 5 ],
          [ 0, 3, 0, 0, 7 ],
          [ 6, 8, 0, 0, 9 ],
          [ 0, 5, 7, 9, 0 ], ]

# Print the solution
minimumProductMST(graph)
```

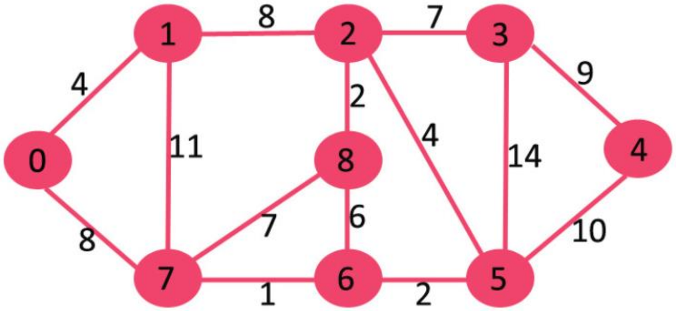
13.5 Reverse Delete Algorithm for Minimum Spanning Tree

In Reverse Delete algorithm, we sort all edges in decreasing order of their weights. After sorting, we one by one pick edges in decreasing order. We include current picked edge if excluding current edge causes disconnection in current graph. The main idea is delete edge if its deletion does not lead to disconnection of graph.

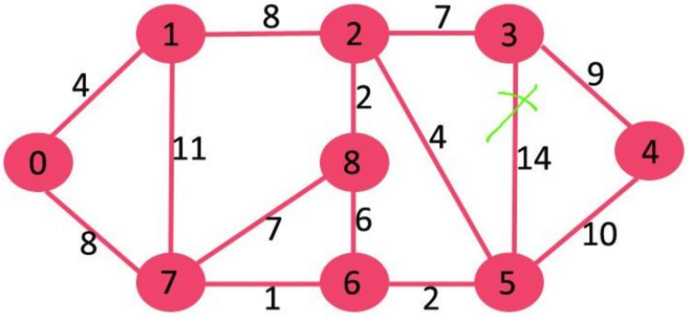
Algorithm:

1. Sort all edges of graph in non-increasing order of edge weights.
2. Initialize MST as original graph and remove extra edges using step 3.
3. Pick highest weight edge from remaining edges and check if deleting the edge disconnects the graph or not.
   If disconnects, then we don’t delete the edge.
   Else we delete the edge and continue.

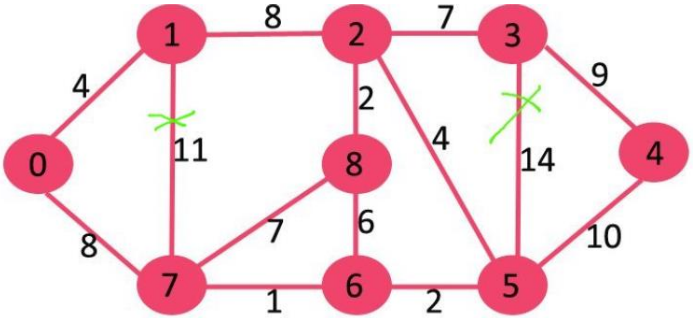
Input: Consider the graph below



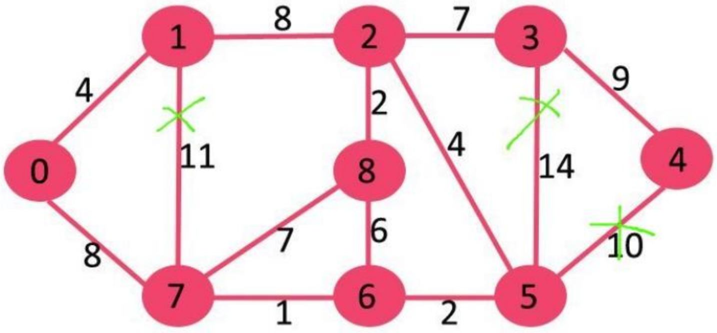
If we delete highest weight edge of weight 14, graph doesn't become disconnected, so we remove it.



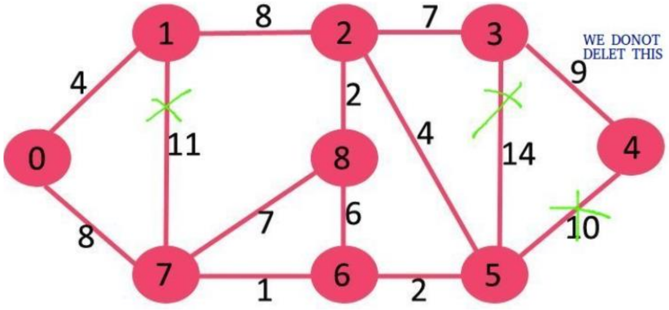
Next we delete 11 as deleting it doesn't disconnect the graph.



Next we delete 10 as deleting it doesn't disconnect the graph.



Next is 9. We cannot delete 9 as deleting it causes disconnection.



We continue this way and following edges remain in final MST.

Edges in MST

(3, 4)
(0, 7)
(2, 3)
(2, 5)
(0, 1)
(5, 6)
(2, 8)
(6, 7)

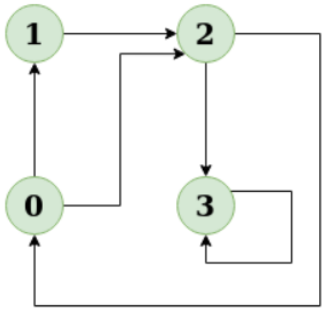
```
# Find Minimum Spanning Tree of a graph using Reverse Delete Algorithm

# Graph class represents a directed graph using adjacency list representation
class Graph:
    def __init__(self, v):

        # No. of vertices
        self.v = v
        self.adj = [0] * v
        self.edges = []
        for i in range(v):
            self.adj[i] = []

    # function to add an edge to graph
    def addEdge(self, u: int, v: int, w: int):
        # write code here
        …

    def dfs(self, v: int, visited: list):
       # write code here
        …

    # Returns true if graph is connected
    # Returns true if given graph is connected, else false
    def connected(self):
       # write code here
       …

    # This function assumes that edge (u, v) exists in graph or not
    def reverseDeleteMST(self):
        # write code here
        …

# Driver Code
# create the graph given in above figure
```

```
V = 9
g = Graph(V)

# making above shown graph
g.addEdge(0, 1, 4)
g.addEdge(0, 7, 8)
g.addEdge(1, 2, 8)
g.addEdge(1, 7, 11)
g.addEdge(2, 3, 7)
g.addEdge(2, 8, 2)
g.addEdge(2, 5, 4)
g.addEdge(3, 4, 9)
g.addEdge(3, 5, 14)
g.addEdge(4, 5, 10)
g.addEdge(5, 6, 2)
g.addEdge(6, 7, 1)
g.addEdge(6, 8, 6)
g.addEdge(7, 8, 7)

g.reverseDeleteMST()
```

Try:

1. **Detect Cycle in a Directed Graph:** Given the root of a Directed graph, The task is to check whether the graph contains a cycle or not.

Input: N = 4, E = 6



Output: Yes
Explanation: The diagram clearly shows a cycle 0 -> 2 -> 0

14. Final Notes

The only way to learn programming is program, program and program on challenging problems. The problems in this tutorial are certainly NOT challenging. There are tens of thousands of challenging problems available – used in training for various programming contests (such as International Collegiate Programming Contest (ICPC), International Olympiad in Informatics (IOI)). Check out these sites:

- The ACM - ICPC International collegiate programming contest (https://icpc.global/)
- The Topcoder Open (TCO) annual programming and design contest (https://www.topcoder.com/)
- Universidad de Valladolid's online judge (https://uva.onlinejudge.org/).
- Peking University's online judge (http://poj.org/).
- USA Computing Olympiad (USACO) Training Program @ http://train.usaco.org/usacogate.
- Google's coding competitions (https://codingcompetitions.withgoogle.com/codejam, https://codingcompetitions.withgoogle.com/hashcode)
- The ICFP programming contest (https://www.icfpconference.org/)

- BME International 24-hours programming contest (https://www.challenge24.org/)
- The International Obfuscated C Code Contest (https://www0.us.ioccc.org/main.html)
- Internet Problem Solving Contest (https://ipsc.ksp.sk/)
- Microsoft Imagine Cup (https://imaginecup.microsoft.com/en-us)
- Hewlett Packard Enterprise (HPE) Codewars (https://hpecodewars.org/)
- OpenChallenge (https://www.openchallenge.org/)

Coding Contests Scores

Students must solve problems and attain scores in the following coding contests:

Name of the contest	Minimum number of problems to solve	Required score
• CodeChef	20	200
• Leetcode	20	200
• GeeksforGeeks	20	200
• SPOJ	5	50
• InterviewBit	10	1000
• Hackerrank	25	250
• Codeforces	10	100
• BuildIT	50	500
	Total score need to obtain	2500

Student must have any one of the following certifications:

1. HackerRank - Problem Solving Skills Certification (Basic and Intermediate)
2. GeeksforGeeks – Data Structures and Algorithms Certification
3. CodeChef - Learn Data Structures and Algorithms Certification
4. Interviewbit – DSA pro / Python pro
5. Edx – Data Structures and Algorithms
5. NPTEL – Programming, Data Structures and Algorithms
6. NPTEL – Introduction to Data Structures and Algorithms
7. NPTEL – Data Structures and Algorithms
8. NPTEL – Programming and Data Structure

V. TEXT BOOKS:

1. Rance D. Necaise, “Data Structures and Algorithms using Python”, Wiley Student Edition.
2. Benjamin Baka, David Julian, “Python Data Structures and Algorithms”, Packt Publishers, 2017.

VI. REFERENCE BOOKS:

1. S. Lipschutz, “Data Structures”, Tata McGraw Hill Education, 1st edition, 2008.
2. D. Samanta, “Classic Data Structures”, PHI Learning, 2nd edition, 2004.

VII. ELECTRONICS RESOURCES:

1. https://www.tutorialspoint.com/data_structures_algorithms/algorithms_basics.htm
2. https://www.codechef.com/certification/data-structures-and-algorithms/prepare
3. https://www.cs.auckland.ac.nz/software/AlgAnim/dsToC.html
4. https://online-learning.harvard.edu/course/data-structures-and-algorithms

VIII. MATERIALS ONLINE

1. Course Content
2. Lab manual

ESSENCE OF INDIAN TRADITIONAL KNOWLEDGE

III Semester: Common for all branches

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC10	MC	-	-	-	-	-	-	-
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: Nil		

Prerequisite: No Prerequisites

COURSE OBJECTIVES:

The course should enable the students to:

I. Understand the concept of Traditional knowledge and its importance

II. Know the need and importance of protecting traditional knowledge.

III. Know the various enactments related to the protection of traditional knowledge.

IV. Understand the concepts of Intellectual property to protect the traditional knowledge

MODULE-I INTRODUCTION TO TRADITIONAL KNOWLEDGE

Define traditional knowledge, nature and characteristics, scope and importance, kinds of traditional knowledge, the physical and social contexts in which traditional knowledge develop, the historical impact of social change on traditional knowledge systems. Indigenous Knowledge (IK), characteristics, traditional knowledge vis-à-vis indigenous knowledge, traditional knowledge Vs western knowledge traditional knowledge vis-à-vis formal knowledge

MODULE-II PROTECTION OF TRADITIONAL KNOWLEDGE

Protection of traditional knowledge: The need for protecting traditional knowledge Significance of TK Protection, value of TK in global economy, Role of Government to harness TK.

MODULE-III LEGAL FRAMEWORK AND TK

A: The Scheduled Tribes and Other Traditional Forest Dwellers (Recognition of Forest Rights) Act, 2006, Plant Varieties Protection and Farmer's Rights Act, 2001 (PPVFR Act);

B: The Biological Diversity Act 2002 and Rules 2004, the protection of traditional knowledge bill, 2016. Geographical indicators act 2003.

MODULE-IV TRADITIONAL KNOWLEDGE AND INTELLECTUAL PROPERTY

Systems of traditional knowledge protection, Legal concepts for the protection of traditional knowledge, Certain non IPR mechanisms of traditional knowledge protection, Patents and traditional knowledge, Strategies to increase protection of traditional knowledge, global legal FORA for increasing protection of Indian Traditional Knowledge.

MODULE-V TRADITIONAL KNOWLEDGE IN DIFFERENT SECTORS:

Traditional knowledge and engineering, Traditional medicine system, TK and biotechnology, TK in agriculture, Traditional societies depend on it for their food and healthcare needs, Importance of conservation and sustainable development of environment, Management of biodiversity, Food security of the country and protection of TK. 139.

Text Books:

1. Traditional Knowledge System in India, by Amit Jha, 2009.
2. Traditional Knowledge System and Technology in India by Basanta Kumar Mohanta and Vipin Kumar Singh, Pratibha Prakashan 2012.

Reference Books:

1. Traditional Knowledge System in India by Amit Jha Atlantic publishers, 2002.
2. "Knowledge Traditions and Practices of India" Kapil Kapoor1, Michel Danino2

COMPLEX ANALYSIS AND SPECIAL FUNCTIONS

IV Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC12	Foundation	3	1	0	4	40	60	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisite: Basic Principles of complex functions

I. COURSEOVERVIEW:

This course Complex Analysis and Special Functions provides an introduction to complex analysis which is theory of complex functions with complex variable. The course includes complex functions and differentiation, complex integration, power series expansion of complex function and special functions. The mathematical skills derived from this course form a necessary base to analytical and design concepts encountered in the program.

II. COURSE OBJECTIVES:

The course will enable the students to learn:

I. The applications of complex variable and conformal mapping in two dimensional complex potential theories.
II. The fundamental calculus theorems and criteria for the independent path on contour integral used in problems of engineering.
III. The concepts of special functions and its application for solving the partial differential equations in physics and engineering.
IV. The mathematics of combinatorial enumeration by using generating functions and complex analysis for understanding the numerical growth rates.

III. COURSE SYLLABUS:

MODULE-I COMPLEX FUNCTIONS AND DIFFERENTIATION (09)

Complex functions differentiation and integration: Complex functions and its representation on argand plane, concepts of limit, continuity, differentiability, analyticity, Cauchy-Riemann conditions and harmonic functions; Milne-Thomson method. Bilinear Transformation.

MODULE -II COMPLEX INTEGRATION (09)

Line integral: Evaluation along a path and by indefinite integration; Cauchy's integral theorem; Cauchy's integral formula; Generalized integral formula; Power series expansions of complex functions and contour Integration: Radius of convergence.

MODULE -III POWER SERIES EXPANSION OF COMPLEX FUNCTION (09)

Expansion in Taylor's series, Maclaurin's series and Laurent series. Singular point; Isolated singular point; Pole of order m; Essential singularity; Residue: Cauchy Residue Theorem.

Evaluation of Residue by Laurent Series and Residue Theorem.
Evaluation of integrals of the type$\int_0^{2\pi} f(cos\theta . sin\theta)\, d\theta$,$\int_{-\infty}^{\infty} f(x)\, dx$

MODULE -IV SPECIAL FUNCTIONS-I (09)

Improper integrals; Beta and Gamma functions: Definitions; Properties of Beta and Gamma function; Standard forms of Beta functions; Relationship between Beta and Gamma functions.

MODULE -V SPECIAL FUNCTIONS-II (09)

Bessel's Differential equation: Bessel function, properties of Bessel function, Recurrence relations of Bessel function, Generating function and Orthogonality of Bessel function, Trigonometric expansions involving Besse function.

IV. TEXT BOOKS

1. Kreyszig, "Advanced Engineering Mathematics", John Wiley & Sons Publishers, 10th Edition, 2010.
2. B. S. Grewal, "Higher Engineering Mathematics", Khanna Publishers, 43rd Edition, 2015.

V. REFERENCE BOOKS:

1. T.K.V Iyengar, B.Krishna Gandhi, “Engineering Mathematics - III”, S.Chand& Co., 12th Edition, 2015.
2. RK Jain & SRK Iyengar, “Advanced Engineering Mathematics”, Narosa Publishers, 5th Edition, 2016.

VI. WEB REFERENCES:

1. http://www.efunda.com/math/math_home/math.cfm
2. http://www.ocw.mit.edu/resourcs/#Mathematics
3. http://www.sosmath.com
4. http://www.mathworld.wolfram.com

VII. E-TEXT BOOKS:

1. http://www.keralatechnologicaluniversity.blogspot.in/2015/06/erwin-kreyszig-advanced-engineering-mathematics-ktu-ebook-download.html
2. http://www.faadooengineers.com/threads/13449-Engineering-Maths-II-eBooks

ANALOG AND PULSE CIRCUITS

IV Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC09	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisites: **There are no prerequisites to take this course.**								

I. COURSE OVERVIEW:

This course provides circuit analysis to design high frequency amplifiers and wave shaping circuits using discrete components. It covers multistage amplifiers, power amplifiers, feedback concepts, sampling gates and multivibrators. Analog electronics are widely used in radio and audio equipment and in many applications where signals are derived from analog sensors and transducers.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The analysis of transistor amplifiers using low frequency and high frequency signals.
II. The response of a linear wave shaping circuits of low pass and high pass filters.
III. The generation of nonlinear oscillations by using regenerative feedback circuit for multivibrators.

III. COURSE SYLLABUS:

MODULE – I: MULTISTAGE AMPLIFIERS PROBABILITY (08)

Classification of Amplifiers, Distortion in amplifiers, Different coupling schemes used in amplifiers, Millers theorem and its dual for single stage amplifier, Frequency response and Analysis of multistage amplifiers, Cascode amplifier, Darlington pair. Transistor at High Frequency: Hybrid - model of Common Emitter transistor model, fα, β and unity gain bandwidth, Gain band width product.

MODULE – II: FEEDBACK AMPLIFIERS (08)

Concepts of feedback, classification of feedback amplifiers, general characteristics of negative feedback amplifiers, effect of feedback on amplifier characteristics, voltage series, voltage shunt, current series and current shunt feedback configurations.

MODULE – III: OSCILLATORS AND LARGE SIGNAL AMPLIFIERS (12)

Condition for Oscillations, RC type Oscillators-RC phase shift and Wien-bridge Oscillators, LC type Oscillators, generalized analysis of LC oscillators, hartley and colpitts Oscillators, frequency and amplitude stability of Oscillators, crystal Oscillator.

Class A Power Amplifier- Series fed and Transformer coupled, Conversion Efficiency, Class B Power Amplifier- Push Pull and Complimentary Symmetry configurations, Conversion Efficiency, Principle of operation of Class AB and Class C Amplifiers. Tuned Amplifiers: Single Tuned Amplifiers – Q-factor, frequency response of tuned amplifiers, Concept of stagger tuning and synchronous tuning.

MODULE – IV: LINEAR WAVE SHAPING AND SAMPLING GATES (08)

Linear wave shaping circuits: High pass RC and low pass RC circuits, response to step and square inputs with different time constants, high pass RC circuit as a differentiator, low pass RC circuit as an integrator.
Sampling gates: basic operating principle of sampling gate, uni and bi directional sampling gates.

MODULE – V: MULTIVIBRATORS (09)

Multivibrators: Bistable multivibrator, unsymmetrical triggering, symmetrical triggering; Schmitt trigger; Monostable multivibrator, Astable multivibrator.

IV. TEXT BOOKS:

1. Jacob Millman, Christos C Halkias, “Integrated Electronics” McGraw Hill Education, 2nd Edition, 2010.

2. Thomas L. Floyd, “Electronic Devices Conventional and Current Version”, Pearson Education,2015.
3 A. Anand Kumar, “Pulse and Digital Circuits”, PHI learning, 2nd Edition, 2005.

V. REFERENCE BOOKS:

1. David A. Bell, “Electronic Devices and Circuits”, Oxford, 5th Edition, 1986.
2. Robert L. Boylestead, Louis Nashelsky, “Electronic Devices and Circuits Theory”, Pearson Education, 11th Edition, 2009.
3. Millman J., Taub, “Pulse, Digital and Switching Waveforms”, Tata McGraw-Hill, 2nd Edition, 2007.

VI. WEB REFERENCES:

1. www.nptel.ac.in
2. notes.specworld.in/pdc-pulse-and-digital-circuits
3. http:// www.introni.it/pdf/Millman-Taub- Pulse and Digital Switching Waveforms1965.pdf
4. https://www.jntubook.com/pulse-digital-circuits-textbook-free-download/

ANALOG AND DIGITAL COMMUNICATIONS

IV Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC10	Core	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

Communications emphasize on generation, transmission and reception of audio, video and telephony signals. The course is intended to understand various analog and pulse modulation schemes. Further, it emphasis the knowledge on various digital modulation techniques and linear block codes. Communication system principles are used for real world applications of radio and TV broadcasting systems.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The need of modulation, generation and detection techniques of analog and pulse modulation systems.

II. Familiarize with digital systems like Pulse code modulation (PCM), Differential pulse code modulation (DPCM), Delta modulation (DM) and Adaptive DM.

III. The applications of spread spectrum techniques in secured digital communication systems.

III. COURSE SYLLABUS:

MODULE – I: AMPLITUDE MODULATION (09)

Introduction to communication system, need for modulation, Frequency division multiplexing, Amplitude modulation - time and frequency domain description, single tone modulation, power relations in AM waves, generation of AM waves - switching modulator, detection of AM Waves - envelope detector, DSBSC modulation - time and frequency domain description, generation of DSBSC Waves - Balanced modulators, coherent detection of DSB-SC modulated waves, SSB modulation - time and frequency domain description, frequency discrimination and phase discrimination methods for generating SSB, demodulation of SSB Waves, principle of vestigial side band modulation.

MODULE – II: ANALOG MODULATION (09)

Basic concepts of phase modulation, Frequency modulation: Single tone frequency modulation, Narrow band FM, Wide band FM, constant average power, transmission bandwidth of FM wave -generation of FM signal direct method and Armstrong method, detection of FM Signal: balanced slope detector, phase locked loop, comparison of FM and AM., concept of pre-emphasis and de-emphasis.

MODULE – III: ANGLE AND DIGITAL PULSE MODULATIONS (09)

Pulse modulation: types of Pulse modulation- Pulse amplitude modulation (PAM), Pulse width modulation (PWM), Pulse position modulation (PPM), comparison of FDM and TDM.

Elements of digital communications: Pulse code modulation, pulse code modulation (PCM) generation and reconstruction, quantization noise, uniform and non-uniform quantization and companding, Differential pulse code modulation (DPCM), Delta modulation (DM) and Adaptive DM, Noise in DM.

MODULE – IV: DIGITAL MODULATION TECHNIQUES (09)

Amplitude shift keying (ASK)- modulator, coherent ASK detector, Frequency shift keying(FSK)- modulator, Non-coherent FSK detector, Binary phase shift keying(BPSK)- modulator, detector, principles of QPSK, Differential PSK, Probability of error for ASK,FSK, PSK.

MODULE – V: SPREAD SPECTRUM MODULATION AND ERROR CONTROL CODES (09)

Spread spectrum modulation: Use of spread spectrum; Direct sequence spread spectrum (DSSS); Code division multiple access using DSSS, frequency hopping spread spectrum; PN-Sequences: Generation and characteristics.

.

Linear Block Codes: Introduction to error control coding; Matrix description of linear block codes, error detection and error correction capabilities of linear block codes; Hamming code; Binary cyclic codes algebraic structure, encoding and decoding.

IV. TEXT BOOKS:

1. Simon Haykin, “Analog and Digital Communications”, John Wiley, 2005.
2. Wayne Tomasi, “Electronics Communication Systems-Fundamentals through Advanced”, 5th Edition, 2009.
3. K. Sam Shanmugam, “Digital and Analog Communication Systems”, John Wiley & Sons, 2nd Edition, 2005.

V. REFERENCE BOOKS:

1. B.P.Lathi, “Modern Analog and Digital Communication”, Oxford reprint, 3rd Edition, 2004.
2. Singh, Sapre, “Communication Systems Analog and Digital”, TMH, 2nd Edition, 2004.
3. Herbert Taub, Donald L Schilling, Goutam Saha, “Principles of Communication Systems”, 3rd Edition, McGraw-Hill, 2008.

VI. WEB REFERENCES:

1. http://www.uotechnology.edu.iq
2. http://nptel.ac.in/
3. http://www.iare.ac.in

VII. E-TEXT BOOKS:

1. http://www.bookboon.com/
2. http://www.jntubook.com
3. http://www.smartzworld.com
4. http://www.archive.org

ELECTROMAGNETIC WAVES AND TRANSMISSION LINES

IV Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC11	PCC	L	T	P	C	CIA	SEE	Total
		3	1	0	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		
Prerequisites: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This course gives the necessary information about the formation of magnetic fields when electric current flows and structures to conduct electromagnetic waves. It covers the fundamental concepts of electro-magnetic wave theory and introduces the basic laws of electromagnetic fields, time varying Maxwell's equations, wave propagation and transmission lines. It provides a platform for advanced courses such as antennas and wave propagation, microwave engineering, transmission via wired links, radio channels and optical fiber networks.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The knowledge required to understand various engineering applications involving electromagnetic fields.
II. The wave propagation characteristics of electromagnetic wave in bounded and unbounded media.
III. The basic theory of transmission lines, appropriate tools (smith chart) to analyze transmission lines.

III. COURSE SYLLABUS:

MODULE – I: ELECTROSTATICS (12)

Electrostatics: Coulomb's law, electric field intensity, fields due to different charge distributions; Electric flux density, Gauss law and its applications; Scalar electric potential; Energy density, illustrative problems; Conductors and dielectrics-characterization; Convection and conduction currents; Dielectric constant, isotropic and homogeneous dielectrics; Continuity equation and relaxation time, conductivity, power absorbed in conductor, Poisson's and Laplace's equations; Capacitance: Parallel plate, co axial, spherical capacitors; Method of images; Illustrative problems.

MODULE – II: MAGNETOSTATICS (12)

Magnetostatics: Biot-savart law; Ampere's circuital law and applications; Magnetic flux density; Magnetic scalar and vector potentials; Forces due to magnetic fields; Ampere's force law; Boundary conditions: Dielectric- dielectric, dielectric conductor interfaces; Inductances and magnetic energy; Illustrative problems; Maxwell's equations (Time varying fields): Faraday's law; Inconsistency of ampere's law for time varying fields and definition for displacement current density; Maxwell's equations in differential form, integral form and word Statements.

MODULE – III: UNIFORM PLANE WAVES (12)

Uniform plane waves: Wave equations for conducting and perfect dielectric media; Relation between E and H; Wave propagation in lossless and conducting media, Loss tangent, Intrinsic impedance; Skin depth; Polarization, Illustrative problems.

Reflection/refraction of plane waves: Reflection and refraction at normal incidence, reflection and refraction at oblique incidence; Standing waves; Brewster angle, critical angle, total internal reflection, surface impedance; Poynting vector and poynting theorem-applications; Power loss in plane conductor; Illustrative problems.

MODULE – IV: TRANSMISSION LINE CHARACTERISTICS (12)

Transmission line characteristics: Types, transmission line parameters, transmission line equations, characteristic impedance, propagation constant; Phase and group velocities; Infinite line concepts, Loss less /low loss transmission line characterization; condition for distortion less and minimum attenuation in transmission lines; Loading: Types of loading; Illustrative problems.

MODULE – V: UHF TRANSMISSION LINES AND APPLICATIONS (12)

UHF transmission lines and applications: Input impedance relations; SC and OC lines; Reflection coefficient, VSWR; UHF lines as circuit elements, $\lambda/4$, $\lambda/2$ and $\lambda/8$ lines, impedance transformations, significance of Z_{min} and Z_{max}; Smith chart: Configuration and applications; Single and double stub matching; Illustrative problems.

IV. TEXT BOOKS:

1. Matthew N.O. Sadiku, “Elements of Electromagnetic”, Oxford University Press, 4th Edition, 2009.
2. E.C. Jordan, K.G. Balmain, “Electromagnetic waves and Radiating Systems”, PHI learning, 2nd Edition, 2000.
3. Umesh Sinha, Satya Prakashan, “Transmission lines and Networks”, Tech India Publications, 1st Edition, 2010.

V. REFERENCE BOOKS:

1. Nathan Ida, “Engineering Electromagnetic”, Springer (India) Pvt. Ltd, 2nd Edition, 2005
2. William H. Hayt Jr., John A. Buck, “Engineering electromagnetic”, Tata McGraw Hill, 7th Edition, 2006.
3. G. Sashibushana Rao, “Electromagnetic Field theory and Transmission Lines, Wiley India, 2013.
4. John D. Ryder, “Networks, Lines and Fields”, PHI learning, 2nd Edition, 1999.

VI. WEB REFERENCES:

1. http:// web.stanford.edu/class
2. http://www.electronicagroup.com
3. http://www.cpri.in/about-us/departmentsunits/library-and-information-centre/digital-library-links.html
4. http://nptel.ac.in/courses/antennas
5. http://www.tutorialspoint.com/discrete_mathematics

IC APPLICATIONS

IV Semester: ECE									
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC12	Core	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This course deals with the fundamental concepts of operational amplifier, linear & nonlinear application of op-amp and digital Integrated circuits. It covers design and analysis of frequency selective and tuning circuits like oscillators, active filters, Phase locked loops and its use for communication applications. Along with switching applications like that of comparators, learn IC based design of voltage regulators, digital IC's for combination and sequential circuit designs. This course forms the basis for the next level of course VLSI Design.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic building blocks, characteristics and applications of operational amplifier.
II. The functional details of logic families, combinatorial and sequential digital circuits (ICs) used in digital design.
III. Different IC models which are basic for Mixed signal integrated circuits in future

III. COURSE SYLLABUS:

MODULE – I: OPERATIONAL AMPLIFIER (08)

Operational Amplifier: Differential Amplifier, DC and AC analysis of dual input balanced output configuration, dual input unbalanced output. Characteristics of Op-amps, Op-amp block diagram, ideal and practical Op-amp specifications. DC characteristics: Input & output offset voltages & currents, drift. AC characteristics: Frequency response, slew rate, CMRR and PSRR.

MODULE – II: APPLICATIONS OF OPERATIONAL (09)

Linear applications of Op-amps: Inverting and non-inverting amplifier, integrator, differentiator, instrumentation amplifier, AC amplifier. Non-linear applications of Op-Amps: Comparators, multi vibrators, triangular, saw tooth, square wave generators, log and anti-log amplifiers. Introduction to voltage regulators, features of 723 Regulator, three terminal voltage regulators.

MODULE – III: ACTIVE FILTERS AND TIMERS (09)

Active Filters: Classification of filters, 1st order low pass and high pass filters, 2nd order low pass, high pass, band pass, band reject and all pass filters.
Timers: Introduction to 555 timer, functional diagram, mono-stable, astable operations and applications, schmitt trigger.
PLL: Introduction, block schematic, principles and description of individual blocks,565 PLL.

MODULE – IV: DATA CONVERTERS (10)

Data converters: Introduction, classification, need of data converters. DAC techniques: weighted resistor DAC, R-2R ladder DAC, inverted R-2R DAC, and IC 1408 DAC. ADC techniques: Flash converters, successive approximation, integrating ADC. DAC/ADC characteristics.

MODULE – V: DIGITAL IC APPLICATIONS (09)

Study of digital logic families such as Resistor Transistor Logic(RTL), Diode Transistor Logic(DTL), Transistor Logic(TTL), Emitter Coupled Logic and CMOS. Characteristics of digital logic families containing fan-in, fan-out, power dissipation, propagation delay and noise margin, Familiarity with commonly available 74XX & CMOS 40XX series ICs-Flip Flops (IC 7474, IC 7473), Shift Registers, Universal Shift Register (IC 74194), Synchronous counters (74LS93,74HC163), Decade Counters, (74HC190).

IV. TEXT BOOKS:

1. D.Roy Chowdhury, “Linear Integrated Circuits”, Newage international (p) Ltd, 2nd Edition, 2003.
2. Ramakanth A. Gayakwad, “Op-Amps &linear ICs”, PHI, 3rd Edition, 2003.
3. JohnF.Wakerly, “Digital Design Principles and Practices”, Prentice Hall, 3rd Edition, 2005.
4. M. MorrisMano,Michael D. Ciletti, “Digital Design”, Pearson Education/PHI, 3rd Edition, 2008.
5. Matthew N.O. Sadiku, “Elements of Electromagnetic”, Oxford University Press, 4th Edition, 2009.

V. REFERENCE BOOKS:

1. Salivahanan, “Linear Integrated Circuits and Applications”, TMH, 1st Edition, 2008.
2. Nathan Ida, “Engineering Electromagnetic”, Springer (India) Pvt. Ltd, 2nd Edition, 2005.

VI. WEB REFERENCES:

1. https://www.nptel.ac.in
2. https://www.svecw.edu.in
3. https://www.smartzworld.com
4. https://www.crectirupati.com http:// web.stanford.edu/class

EXPERIENTIAL ENGINEERING EDUCATION (EXEED) – FABRICATION / MODEL DEVELOPMENT

IV Semester: Common for all branches								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC14	Foundation	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		2	0	0	1	30	70	100
Contact Classes: 28	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 28		
Prerequisite: There are no prerequisites to take this course								

COURSE OVERVIEW:

This course provide the environment to develop high-tech, ecological and socially responsible products from concept and design to production. The course covers hands-on learning in product and industrial design and product development with product lifecycle management.

OBJECTIVES:

The students will try to learn:

I. The design thinking process and Identify opportunities through customer needs analysis.
II. Product specifications based on customer needs that are desirable, feasible, and viable through applied creativity.
III. The Implementation techniques for planning and executing a prototype design services.

WEEK NO	TOPIC
WEEK – I	Introduction To Product Design
WEEK – II	Design Thinking Skills
WEEK – III	Identifying Customer Needs
WEEK – IV	Product Specifications
WEEK – V	Applied Creativity
WEEK – VI	Prototyping
WEEK – VII	Design Of Services
WEEK –VIII	Product Architecture
WEEK - IX	Financial Analysis
WEEK - X	Design For Environment
WEEK - XI	Product Development Process
WEEK - XII	Reverse Engineering
WEEK - XIII	Value Engineering
WEEK - XIV	Assessment

ANALOG AND PULSE CIRCUITS LABORATORY

IV Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC13	Core	L	T	P	C	CIA	SEE	Total
		0	0	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes:36		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This laboratory course builds on the lecture course "Electronic circuit analysis" and “pulse and digital circuits” which is mandatory for all students of electronics and communication engineering. The course aims at practical experience with the characteristics and theoretical principles of linear and non linear devices and pulse circuits.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Single stage and multi stage amplifiers and oscillators.
II. The principles of feedback amplifiers and oscillators through simulation.
III. The operations of circuits for linear and nonlinear wave shaping.
IV. The characteristics of different multivibrators.

III. COURSE SYLLABUS:

Week – 1: BASIC AMPLIFIERS/ LINEAR WAVESHAPING

a. Simulate frequency response of common emitter amplifier and common base amplifier.
b. Design RC low pass and high pass circuit for different time constants.

Week – 2: BASIC AMPLIFIERS/ LINEAR WAVESHAPING

a. Design RC low pass and high pass circuit for different time constants
b. Simulate frequency response of common emitter amplifier and common base amplifier.

Week – 3: TWO STAGE RC COUPLED AMPLIFIER/ NON-LINEAR WAVESHAPING

a. Simulate frequency response of two stage RC coupled amplifier.
b. Design transfer characteristics of clippers and clampers

Week – 4: TWO STAGE RC COUPLED AMPLIFIER/ NON-LINEAR WAVESHAPING

a. Design transfer characteristics of clippers and clampers.
b. Simulate frequency response of two stage RC coupled amplifier.

Week – 5: SINGLE TUNED AMPLIFIERS/ TRANSISTOR AS A SWITCH

a. Simulate a single tuned amplifier.
b. Design of transistor as a switch. styles

Week – 6: SINGLE TUNED AMPLIFIERS/ TRANSISTOR AS A SWITCH

a. Design of transistor as a switch.
b. Simulate a single tuned amplifier.

Week – 7: FEEDBACK AMPLIFIERS/ COMPARATOR

a. Simulate voltage series feedback amplifier and current shunt feedback amplifier.
b. Design of comparator circuit.

Week – 8: FEEDBACK AMPLIFIERS/ COMPARATOR

a. Design of comparator circuit.
b. Simulate voltage series feedback amplifier and current shunt feedback amplifier

Week – 9: RC PHASE SHIFT OSCILLATOR USING TRANSISTOR/ MULTIVIBRATORS

a. Simulate sine wave generated for a particular frequency by an RC phase shift oscillator.
b. Design different types of multivibrators and plot its waveforms.

Week – 10: RC PHASE SHIFT OSCILLATOR USING TRANSISTOR/ MULTIVIBRATORS

a. Design different types of multivibrators and plot its waveforms.
b. Simulate sine wave generated for a particular frequency by an RC phase shift oscillator.

Week – 11: OSCILLATORS/ SCHMIT TRIGGER

a. Simulate sine wave generated for a particular frequency by Colpitts and Hartley oscillator.
b. Design a Schmitt trigger circuit.

Week – 12: OSCILLATORS/ SCHMIT TRIGGER

a. Design a Schmitt trigger circuit.
b. Simulate sine wave generated for a particular frequency by Colpitts and Hartley oscillator.

Week – 13: POWER AMPLIFIERS/ UJT AS A RELAXATION OSCILLATOR

a. Simulate class A power amplifier (transformer less) and class B power amplifier.
b. Design of UJT as a relaxation oscillator.

Week – 14: POWER AMPLIFIERS/ UJT AS A RELAXATION OSCILLATOR

a. Design of UJT as a relaxation oscillator.
b. Simulate class A power amplifier (transformer less) and class B power amplifier.

IV. REFERENCE BOOKS:

1. Jacob Millman, Herbert Taub , Mothiki S. PrakashRao, “Pulse Digital and Switching Waveforms”, Tata McGraw-Hill, 3rd Edition, 2008.
2. David A. Bell, “Solid State Pulse Circuits”, PHI, 4th Edition, 2002.
3. J. Millman, C. C. Halkias, “Integrated Electronics”, Tata McGraw-Hill. 1st Edition, 2008.
4. B. P. Singh, Rekha Singh, “Electronic Devices and Circuits”, Pearson, 1st Edition, 2006.
5. Behzad Razavi, “Design of Analog CMOS Integrated Circuits”, Tata McGraw-Hill, 1st Edition, 2002.

V. WEB REFERENCES:

1. http://www.tedpavlic.com/teaching/osu/ece327/
2. http://www.ee.iitkgp.ac.in

ANALOG AND DIGITAL COMMUNICATIONS LABORATORY

IV Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC14	Core	L	T	P	C	CIA	SEE	Total
		0	0	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes:36**		

Prerequisite: There are no prerequisites to take this course.

I. COURSE OVERVIEW:

Communication engineering is the field of study concerned with the transmission of information either in analog or digital form. The objective of this course provides a platform to the students to understand the basics of analog and digital communication systems, modulation techniques, data transmission, multiplexing, etc

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic theory of communication system in practice.
II. The concept of analog to digital conversion for pulse modulation techniques.
III. The analog and digital modulation techniques using MATLAB tool.

III. COURSE SYLLABUS:

Week – 1: AMPLITUDE MODULATION AND DEMODULATION

Generation of amplitude modulation and demodulation using hardware and MATLAB

Week – 2: DSB-SC MODULATOR & DETECTOR

Generation of AM-Double Side Band Suppressed Carrier (DSB-SC) signal using Balanced Modulator

Week – 3: FREQUENCY MODULATION AND DEMODULATION

Generation of frequency modulation and demodulation using hardware and MATLAB

Week – 4: SAMPLING THEOREM VERIFICATION

Verification of sampling theorem for under, perfect, over sampling cases using hardware and MATLAB.

Week – 5: PULSE WIDTH MODULATION AND DEMODULATION

Generation of Pulse width modulation and demodulation using hardware and MATLAB.

Week – 6: PULSE POSITION MODULATION AND DEMODULATION

Generation of pulse position modulation and demodulation using hardware and MATLAB.

Week – 7: PULSE CODE MODULATION GENERATION AND DTECTION

Generation of pulse code modulation and demodulation using hardware and understanding the concept analog to digital conversion

Week – 8: DIFFERENTIAL PULSE CODE MODULATION

Generation of differential pulse code modulation and demodulation using hardware

Week – 9: DELTA MODULATION

Generation of delta modulation and demodulation using hardware.

Week – 10: TIME DIVISION MULTIPLEXING & DE MULTIPLEXING

To study the operation of Time-Division multiplexing and demultiplexing.

Week – 11: FREQUENCY SHIFT KEYING GENERATION AND DETECTION

Generation of Frequency shift keying modulation and demodulation using hardware

Week – 12: BINARY PHASE SHIFT KEYING GENERATION AND DETECTION

Generation of Phase shift keying modulation and demodulation using hardware

Week – 13: DIFFERENTIAL PHASE SHIFT KEYING GENERATION AND DETECTION

Generation of Differential Phase shift keying modulation and demodulation using hardware

Week – 13: AMPLITUDE SHIFT KEYING GENERATION AND DETECTION

Generation of Amplitude Shift Key modulation and demodulation using hardware

IV. REFERENCE BOOKS:

1. K. Sam Shanmugam, "Digital and Analog Communication Systems", John Wiley & Sons, 2nd Edition, 2005.
2. B.P.Lathi, "Modern Analog and Digital Communication", Oxford reprint, 3rd Edition, 2004.
3. Singh, Sapre, "Communication Systems Analog and Digital", TMH, 2nd Edition, 2004

V. WEB REFERENCES:

1. https://ocw.mit.edu/courses/electrical.../6...digital-communications.../lecture-notes
2. https://everythingvtu.wordpress.com
3. http://www.iare.ac.in

IC APPLICATIONS LABORATORY

IV Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC15	Core	L	T	P	C	CIA	SEE	Total
		0	0	2	1	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 26				Total Classes:26		
Prerequisite: There are no prerequisites to take this course.								

I. COURSE OVERVIEW:

This course imparts hands-on knowledge for integrated circuit applications. It enables the students to design linear and non-linear applications using op-amp and pulse generation circuits using timer IC. Provide the capability to use vivado tool for performing the combinational and sequential circuits.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Implement different circuits and verify circuit concepts.
II. Study the concepts of multi vibrators and filters.
III. Verify the operations of the 555 timers and PLLs and their applications.
IV. Design and verify combinational and sequential circuits.

III. COURSE SYLLABUS:

Week – 1: INVERTING, NON-INVERTING AND DIFFERENTIAL AMPLIFIERS

To construct and test the performance of an Inverting, Non-inverting amplifier and Differential amplifier using IC741.

Week – 2: INTEGRATOR AND DIFFERENTIATOR

To construct and test the performance of an Integrator and Differentiator using IC 741.

Week – 3: SECOND ORDER ACTIVE LOWPASS, HIGHPASS AND BANDPASS FILTERS

To design and verify the operation of the Active low pass and High pass using IC 741.

Week – 4: SECOND ORDER ACTIVE BAND PASS AND BANDREJECT FILTERS

To design and verify the operation of the Band pass and Band reject filters using IC 741.

Week – 5: ASTABLE MULTIVIBRATORS USING 555

To design and construct an astable multivibrators using IC 555.

Week – 6: MONOSTABLE MULTIVIBRATORS 555

To design and construct Monostable multivibrators using IC 555.

Week – 7: SCHMITT TRIGGER USING 555

To design and construct schmitt trigger using NE555 Timer.

Week – 8: PLL USING IC 565

Verifying characteristics of PLL.

Week – 9: INSTRUMENTATION AMPLIFIER

To design and verify the operation of instrumentation amplifier using IC 741.

Week – 10: DIGITAL TO ANALOG CONVERTER

To design and verify the operation of R-2R and Inverted R-2R DAC Converter using IC 741.

Week – 11: IC 723

To design and implement voltage regulator using IC 723.

Week – 12: RTL LOGIC

Verify Functionality of NOR and NAND gate using RTL Logic.

Week – 13: DTL LOGIC

Verify Functionality of NOR and NAND gate using DTL Logic..

IV. REFERENCE BOOKS:

1. D. Roy Chowdhury, “Linear Integrated Circuits”, New age international (p) Ltd, 2nd Edition,2003
2. Ramakanth A. Gayakwad, “Op-Amps & linear ICs”, PHI, 3rdEdition,2003.
3. John F. Wakerly, “Digital Design Principles and Practices”, Prentice Hall, 3rdEdition,2005.

V. WEB REFERENCES:

1. Salivahanan, “Linear Integrated Circuits and Applications”, TMH, 1st Edition, 2008.

FUNDAMENTALS OF DATABASE SYSTEMS

IV Semester: CE / EEE / ME / ECE / AE									
Course Code	Category	Hours / Week			Credits	Maximum Marks			
ACSC18	SKILL	L	T	P	C	CIA	SEE	Total	
		-	-	-	-	-	-	-	
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: Nil			

I. COURSE OVERVIEW

The fundamentals of Database systems are vital components of modern information systems. Database applications all pervasive and range in size from small in-memory databases to terabytes or even larger in various applications domains. The course focuses and the fundamentals of knowledgebase and relational database management systems, and the current developments in database theory and their practices.

II. COURSE OBJECTIVES:

The course should enable the students to:

I. Understand the role of database management system in an organization and learn the database concepts.
II. Design databases using data modeling and data normalization techniques.
III. Construct database queries using relational algebra and calculus.
IV. Understand the concept of a database transaction and related database facilities.
V. Learn how to evaluate set of queries in query processing.

III. COURSE SYLLABUS:

MODULE: I CONCEPTUAL MODELING (10)

Introduction to file and database systems: Database system structure, data models: entity relationship model, relational model.

MODULE: II RELATIONAL APPROACH (08)

Relational algebra and calculus: Relational algebra, selection and projection, set operations, renaming, joins, division, examples of algebra queries, relational calculus, tuple relational calculus.

MODULE : III BASIC SQL QUERY AND NORMALIZATION (10)

SQL data definition; Queries in SQL: updates, views, integrity and security, relational database design.

Normal Forms: 1NF, 2NF, 3NF and BCNF.

MODULE : IV TRANSACTION MANAGEMENT (09)

Transaction processing: Introduction, need for concurrency control, desirable properties of transaction, schedule and recoverability, Serializability and schedules

MODULE : V CONCURRENCY CONTROL (08)

Concurrency control; Types of locks: Two phases locking, deadlock, timestamp based concurrency control, recovery techniques, concepts, immediate update, deferred update, shadow paging.

IV. TEXT BOOKS:

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, "Database System Concepts", McGraw-Hill, 4thEdition, 2002.

V. REFERENCE BOOKS:

1. Ramez Elmasri, Shamkant B. Navathe, "Fundamental Database Systems", Pearson Education, 3rdEdition, 2003.

2. Raghu Ramakrishnan, "Database Management System", Tata McGraw-Hill Publishing Company, 3rd Edition, 2003.
3. Hector Garcia Molina, Jeffrey D. Ullman, Jennifer Widom, "Database System Implementation", Pearson Education, United States, 1st Edition, 2000.
4. Peter Rob, Corlos Coronel, "Database System, Design, Implementation and Management", Thompson Learning Course Technology, 5th Edition, 2003.

VI. WEB REFERENCES:

1. https://www.youtube.com/results?search_query=DBMS+onluine+classes
2. http://www.w3schools.in/dbms/
3. http://beginnersbook.com/2015/04/dbms-tutorial/

VII. E-TEXT BOOKS

1. http://www.e-booksdirectory.com/details.php?ebook=10166
2. http://www.e-booksdirectory.com/details.php?ebook=7400re

ANTENNAS AND WAVE PROPAGATION

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC18	Core	3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisites: Electromagnetic Waves and Transmission Lines

I. COURSE OVERVIEW:

This course will cover the fundamentals of antenna, radiation phenomenon, antenna theory, different types of antennas, antenna arrays, design and measurements, concepts of antenna wave propagation (influence of earth’s atmosphere on radio waves). Antennas had wide range of application in government and commercial fields and able to design the antennas like Yagi-Uda and Microstrip. The course presents fundamental theory together with techniques for the practical design, measurement and application of antennas over the RF (radio-frequency) to millimeter wave frequency range.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. Principles of radiation, antenna parameters and working principle of VHF, UHF and Microwave antennas used in communications, broad casting, radar, navigation and similar systems.

II. Familiarize with basic antenna types and common structures, measurement of antenna characteristics and application of antennas over the radio frequency (RF) to micro wave (MW) frequency range.

III. The applications of smart, wideband and ultra wideband (UWB) antennas for wireless communications, satellite communication, and radar systems.

III. COURSE SYLLABUS:

MODULE –I: ANTENNA BASICS (09)

Antenna fundamentals: Introduction, Basic Antenna Parameters-Patterns, Beam Area, Radiation Intensity, Beam Efficiency, Directivity-Gain-Resolution, Antenna Apertures, Effective Height, illustrative Problems, Fields from Oscillating Dipole, Field Zones, Front-to-Back Ratio, Antenna Theorems, Radiation, Retarded Potentials, Radiation from Small Electric Dipole, Quarter Wave Monopole and Half Wave Dipole, Current Distributions, Field Components, Radiated Power, Radiation Resistance, Loop Antennas- Introduction, Small circular Loop, Comparison of Far Fields of Small Loop and Short Dipole.

MODULE –II: VHF,UHF AND MICROWAVE ANTENNAS-I (10)

Arrays with Parasitic Elements, Yagi-Uda Array, Folded Dipoles and their Characteristics, Helical Antennas-Helical Geometry, Helix modes, Practical Design Considerations for Monofilar Helical Antenna in Axial and Normal Modes, Horn Antennas- Types, Fermat’s Principle, Optimum Horns, Design Considerations of Pyramidal Horns, Illustrative Problems.

MODULE –III: VHF,UHF AND MICROWAVE ANTENNAS-II (10)

Microstrip Antennas-Introduction, Basic characteristics of micro strip antennas, Feeding Methods, Methods of Analysis, Rectangular and Circular micro strip antennas, Basic concepts of Smart antennas, concepts and benefits of smart antennas, fixed weight beam forming, adaptive beam forming.

Reflector Antennas- Introduction, Paraboloidal Reflectors- Geometry, Pattern Characteristics, Feed Methods Lens Antennas: Introduction, Geometry of Non-metallic Dielectric Lenses ,Zoning, Tolerances, Applications, Slot Antenna, Babinet‟s Principle, Applications.

MODULE –IV: ANTENNA ARRAYS AND MEASUREMENTS (08)

Antenna Arrays: Point Sources- Definition, Patterns, Arrays of 2 Isotropic Sources – Different Cases, Principle of Pattern Multiplication, Uniform Linear Arrays- Broadside Arrays, End-fire Arrays, EFA with Increased Directivity, Derivation of their Characteristics and Comparison, BSAs with Non-Uniform Amplitude Distributions, General considerations and Binomial Arrays, Illustrative Problems. Antenna Measurements: Introduction, Concepts – Reciprocity, Near and Far Fields, Coordinate System, Sources of Errors Patterns to be Measured, Pattern Measurement Arrangement Directivity

Measurement, Gain Measurements (by Comparison, Absolute and 3-Antenna Methods).

MODULE –V: RADIO WAVE PROPAGATION (08)

Wave Propagation - I: Introduction, definitions, categorizations ,different Modes of Wave Propagation; Ground wave propagation: Introduction, plane earth reflections, , wave tilt, curved earth reflections; Space wave propagation: Introduction, field strength variation with distance and height, effect of earth's curvature, absorption, super refraction, M-Curves, duct propagation, scattering phenomena, tropospheric propagation, fading and path loss calculations;

Wave propagation – II: Sky wave propagation: Introduction, structure of ionosphere, refraction and reflection of sky waves by ionosphere; Ray path, critical frequency, MUF, LUF, OF, virtual height and skip distance; Relation between MUF and skip distance; Multi-hop propagation.

IV. TEXT BOOKS:

1. John D. Kraus, Ronald J. Marhefka, Ahmad S. Khan, "Antennas and Wave Propagation", TMH, 4th Edition, 2010.
2. C.A. Balanis, "Antenna Theory", John Wiley and Sons, 2nd Edition, 2001.

V. REFERENCE BOOKS:

1. E.C. Jordan, K.G. Balmain, "Electromagnetic Waves and Radiating Systems", PHI, 2nd Edition, 2000.
2. E.V.D. Glazier, H.R.L. Lamont, "Transmission and Propagation", Her Majesty's Stationery Office,1958.
3. F.E. Terman, "Electronic and Radio Engineering", McGraw-Hill, 4th Edition, 1955.
4. K.D. Prasad, Satya Prakashan, "Antennas and Wave Propagation", Tech India Publications, 1st Edition,2001.

VI. WEB REFERENCES:

1. http:// web.stanford.edu/class
2. http://www.electronicagroup.com
3. http://www.cpri.in/about-us/departmentsunits/library-and-information-centre/digital-library-links.html
4. http://nptel.ac.in/courses/antennas

VII. E-TEXT BOOKS:

1. http://www.ebookgalaxy.in/2016/01/antennas-and-wave-propagation-by-g-s-n.html#.WBGI7NJ97IU
2. https://www.jntubook.com/antennas-wave-propagation-textbook
3. http://117.55.241.6/library/E-Books/Antennas_mcgraw-hill_2nd_ed_1988-john_d_kraus.pdf
4. http://www.archive.org

MICROPROCESSORS AND MICROCONTROLLERS

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC19	Core	3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		
Prerequisites: Digital System Design								

I. COURSE OVERVIEW:

Processor and controller cores are the key components in most of the modern embedded and system on-chip designs. This course outlines the architecture and signal description of Intel microprocessor and microcontrollers. The instruction set and assembly language programming along with I/O and memory interfacing techniques are covered. The knowledge acquired from this course will enable the students in development of embedded hardware projects and models for engineering and scientific applications.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The signal descriptions along with functional architecture and hardware interfacing skills using microprocessors and microcontrollers.
II. The instruction set and logic to build assembly language programs for arithmetic, logic and automated electronic systems.
III. The essential concepts of development through a practical hands-on approach on advanced ARM processors and Internet of Things based systems.

III. COURSE SYLLABUS:

MODULE –I: 8086 MICROPROCESSOR (10)

8086 Architecture, Register Organization, Memory Segmentation, Signal descriptions of 8086, modes of operation with timing diagrams, interrupts Addressing modes and Instruction Set of 8086. Simple Programs involving arithmetical, Logical, Branch and Call Instructions, Sorting, String Manipulations.

MODULE –II: INTERFACING DEVICES (09)

PIO 8255 modes of operation of 8255, stepper motor interfacing, interfacing to D/A and A/D converters, Semiconductor memory interfacing, dynamic RAM interfacing, USART.

MODULE –III: 8051 MICROCONTROLLER (09)

8051 Microcontroller – Internal architecture and pin configuration, 8051 addressing modes, instruction set, Bit addressable features.

I/O Port structures, assembly language programming using data transfer, arithmetic, logical and branch instructions.

MODULE –IV: SYSTEM DESIGNUSING MICROCONTROLLER (08)

8051 Timers/Counters, Serial data communication and its programming, 8051 interrupts, Interrupt programming. Real world interfacing of 8051 with external memory, expansion of I/O ports, stepper motor, ADC, DAC, LCD.

MODULE –V: ARM ARCHITECTURE (09)

ARM Processor fundamentals, ARM Architecture – Register, CPSR, Pipeline, exceptions and interrupts interrupt vector table, ARM instruction set – Data processing, Branch instructions, load store instructions, Software interrupt instructions, Program status register instructions, loading constants, Conditional execution, Introduction to Thumb instructions.

IV. TEXT BOOKS:

1. A.K Ray, K. M. Bhurchandani, “Advanced Microprocessors and Peripherals” Tata McGraw-Hill Education, 2nd Edition, 2006.
2. Kenneth. J. Ayala, “The 8051 Microcontroller”, Cengage Learning, 3rd Edition, 2004.
3. Andrew N SLOSS, Dominic SYMES, Chris WRIGHT, “ARM System Developers guide”, Elsevier, 1st Edition, 2012.

V. REFERENCE BOOKS:

1. N. Senthil Kumar, M. Saravanan, S. Jeevanathan, S. K. Shah, “Microprocessors and Interfacing”, Oxford University, 1st Edition, 2012.
2. Lyla B. Das, “The x86 Microprocessors”, Pearson India, 2nd Edition, 2014.
3. D. V. Hall, “Microprocessors and Interfacing”, Tata McGraw-Hill Education, 3rd Edition 2013.

VI. WEB REFERENCES:

1. http://www.daenotes.com/electronics/digital-electronics/Intel-80858bitmicroprocessor#axzz2I9yUSe7I
2. https://www.smartzworld.com/notes/microprocessors-and-microcontrollers-mpmc/
3. http://www.iare.ac.in

VII. E-TEXT BOOKS:

1. http://engineersevanigam.blogspot.in/2013/07/microprocessors-and-interfacing-by.html
2. https://www.scribd.com/doc/153593067/Microprocessor-by-A-P-Godse-D-A-Godse

ELECTRONIC MEASUREMENTS AND INSTRUMENTATION

V Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC20	Core	L	T	P	C	CIA	SEE	Total
		3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes:15	Practical Classes: Nil				Total Classes: 60		
Prerequisites: Electronic Devices and Circuits								

I. COURSE OVERVIEW:

The purpose of this course is to design, realization and use of electronic systems for the measurement of electrical and non-electrical quantities. It gives an emphasis on analog and digital instruments, oscilloscopes, signal generators, signal analyzers, AC/DC bridges and transducers. The knowledge of measurements and instrumentation is used to test and analyze the performance of measuring instruments in the field of science, engineering and technology.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The performance characteristics and working principle of analog and digital instruments for measuring electrical quantities.
II. The analysis of various signals by using oscilloscopes and signal analyzers which have built in signal generators.
III. The measurement of unknown resistive and reactive components by using various AC and DC bridge circuits.
IV. The construction and working of transducers for the conversion of physical quantities into electrical quantities.

III. COURSE SYLLABUS:

MODULE –I: INTRODUCTION TO MEASURING INSTRUMENTS (08)

Block schematics of measuring systems, performance characteristics, Static characteristics: Accuracy, resolution, precision, gauss error, types of errors, Dynamic characteristics : Repeatability, reproducibility, fidelity, lag; Analog measuring instruments: D' Arsonval movement, DC voltmeters and ammeter, AC voltmeters and current meters, ohmmeters, multimeters, meter protection, extension of range, digital voltmeters: Ramp type, staircase, dual slope integrating type, successive approximation type, Specifications of instruments.

MODULE –II: OSCILLOSCOPE (09)

Oscilloscopes: CRT, block schematic of CRO, time base circuits, delay lines, high frequency CRO considerations, Applications. Special purpose oscilloscopes: Dual trace, dual beam CROs, Sampling oscilloscopes, Storage oscilloscopes, Digital storage CROs.

MODULE –III: SIGNAL GENERATOR AND SIGNAL ANALYZERS (09)

Signal Generators: AF and RF signal generators, sine and square wave generators, function generators: arbitrary waveform generator, sweep frequency generators, video signal generators, and specifications.

Signal Analyzers: AF, HF wave analyzers, heterodyne wave analyzers, harmonic distortion, spectrum analyzers, power analyzers.

MODULE –IV: AC AND DC BRIDGES (10)

Measurements using DC and AC bridges: Wheat stone bridge, Kelvin bridge, AC bridges, Maxwell, Hay,
Schering, Wien, Anderson bridges, wagner & ground connection.

MODULE –V: TRANSDUCERS (09)

Transducers: Classification, strain gauges, bounded, unbounded; Force and displacement transducers, resistance thermometers, hotwire anemometers, LVDT, thermocouples, synchros, special resistance thermometers, piezoelectric transducers, variable capacitance transducers, magneto strictive transducers.
Measurement of Physical Parameters: Flow measurement, displacement meters, liquid level measurement, temperature - measurements, data acquisition systems.

IV. TEXT BOOKS:

1. K.LalKishore, “Electronic Measurements and Instrumentation”, Pearson Education, 2nd Edition, 2010.
2. H.S.Kalsi, “Electronic Instrumentation”, TMH, 2nd Edition, 2004.
3. A.K.Sawhney, “Electrical and Electronics Measurements and Instrumentation”, 19th Edition, 2011.

V. REFERENCE BOOKS:

1. DavidA.Bell, “Electronic Instrumentation and Measurements”, Oxford University Press, 1st Edition, 2007.
2. A.D.Helbincs, W.D.Cooper, “Modern Electronic Instrumentation and Measurement Techniques”, PHI, 56th Edition, 2003.
3. B.M.Oliver, J.M.Cage, “Electronic Measurements and Instrumentation”, TMH, Reprint, 2009.
4. T.R.Padmanabham, “Industrial Instrumentation”, Springer, 1st Edition,2009.

VI. WEB REFERENCES:

1. https://www.scribd.com/
2. https://www.worldcat.org/
3. https://www.infibeam.com/
4. https://www.abebooks.co.uk

VII. E-TEXT BOOKS:

1. https://www.vssut.ac.in/lecture_notes/lecture1423813026.pdf
2. fmcet.in/ECE/EC2351_uw.pdf
3. https://books.askvenkat.com/tag/measurement-and-instrumentation-lecture-notes-pdf
4. https://www.jntubook.com/electronics-measurements-instrumentation-textbook-free-d

CONTROL SYSTEMS

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC21	Core	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Mathematical Transform Techniques

I. COURSE OVERVIEW:

This course deals with the basic concepts of block diagram reduction technique, time response analysisof first order and second order systems. It deals with various time and frequency domain analysis. It elaborates the concept of stability and its assessment for linear time invariant systems. This course addresses the various real time issues and how the control strategies are used in automation areas associates with variety of engineering streams.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The mathematical models of dynamic systems using the concepts of basic sciences.
II. The system performance using time domain and frequency domain analysis for standard inputs.
III. The classification of controllers and compensators as per the desired dynamic response of the system.
IV. The different ways of system representation such as transfer function and state space.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION TO CONTROL SYSTEMS (08)

Control systems: Introduction, open loop and closed loop systems, examples, comparison, mathematical modeling for Physical Systems: Electrical circuits, dc generator and motors, Mechanical systems, computational systems. Linearization of nonlinear systems. Transfer function representation.

MODULE-II: BLOCK DIAGRAM REDUCTION AND TRANSIENT RESPONSE ANALYSIS (10)

Block Diagrams: Block diagram representation of various systems, block diagram algebra, characteristics of feedback systems, AC servomotor, signal flow graph, Mason’s gain formula; Time-domain specifications,–shifted unit step, shifting theorem, convolution integral, impulse response, unit step response of first and second order systems, steady state errors, system error and Non-unity feedback systems, derivative and proportional derivative, integral and PID controllers.

MODULE-III: CONCEPT OF STABILITY AND ROOT LOCUS TECHNIQUE (09)

Concept of stability: Necessary and sufficient conditions for stability, BIBO stability, Routh’s and Routh Hurwitz stabilitycriterions and limitations.

Root locus technique: Introduction, Root locus plots, construction of root locus, graphical determination of „k‟ for specified damping ratio, relative stability, effect of adding zeros and poles on stability.

MODULE-IV: FREQUENCY DOMAIN ANALYSIS (10)

Frequency domain analysis: Introduction, frequency domain specifications, Bodeplot, Nyquist plot, Nyquist stability criterion, calculation of gain margin and phase margin, determination of transfer function, correlation between time and frequency responses.

MODULE-V: STATE SPACE ANALYSIS AND COMPENSATORS (08)

State Space Analysis: Concept of state, state variables and state model, Solution of state equations ,derivation of state models fromblock diagrams, diagonalization, solving the time invariant state equations, state transition matrix and properties, concept of controllability, observability and pole placement, lead lag compensators: via root locus and frequency domain methods.

IV. TEXT BOOKS:

1. I J Nagrath, M Gopal, “Control Systems Engineering”, New Age International Publications, 3rd Edition,2007.
2. K Ogata, “Modern Control Engineering”, Prentice Hall, 4th Edition, 2003.

3. N C Jagan, “Control Systems”, BS Publications, 1st Edition, 2007.

V. REFERENCE BOOKS:

1. Anand Kumar, “Control Systems”, PHI Learning, 1st Edition, 2007.
2. S Palani, “Control Systems Engineering”, Tata McGraw-Hill Publications, 1st Edition, 2001.
3. N K Sinha, “Control Systems”, New Age International Publishers, 1st Edition, 2002**.**

VI. WEB REFERENCES:

1. https://www.researchgate.net
2. https://www.aar.faculty.asu.edu/classes
3. https://www.facstaff.bucknell.edu/
4. https://www.electrical4u.com
5. https://www.iare.ac.in

VII. E-TEXT BOOKS:

1. https://www.jntubook.com/
2. https://www.freeengineeringbooks.com

CELLULAR AND MOBILE COMMUNICATIONS

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC22	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Analog and Digital Communications

I. COURSE OVERVIEW:

The cellular mobile communication allows the users to communicate with others in different locations without the use of any physical connection. It covers the operation, performance criteria, handoff mechanism and channel assignments of the cellular system. The applications include Wi-Fi, Bluetooth, cell phones and wireless power transfer.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The cellular mobile system, cell coverage, cell site and mobile antennas system for interference reduction.

II. The wireless system standard applications for the global system for mobile communications, code division multiple access and time division multiple access technologies.

III. The advanced intelligent network for wireless communications and future public land mobile telecommunications.

III. COURSE SYLLABUS:

MODULE - I: CELLULARMOBILERADIOSYSTEMS (10)

Introduction to cellular mobile System, performance criteria, uniqueness of mobile radio environment, operation of cellular systems, hexagonal shaped cells, analog and digital Cellular systems, General description of the problem, concept of frequency channels, Frequency reuse, Co-channel Interference Reduction Factor, desired C/I from a normal case in a omni directional Antenna system, Cell splitting, consideration of the components of Cellular system.

MODULE - II: INTERFERENCEANDCELLCOVERAGEFORSIGNALANDTRAFFIC (09)

Introduction to Co-Channel Interference, real time Co-Channel interference, Co-Channel measurement, design of Antenna system, Antenna parameters and their effects, diversity receiver, non-co channel interference-different types,Signal reflections in flat and hilly terrain, effect of human made structures, phase difference between direct and reflected paths, constant standard deviation, straight line path loss slope, general formula for mobile propagation over water and flat open area, near and long distance propagation antenna height gain, form of point to point model. Small-scale fading and multipath: Small scale multipath propagation, types of small - Scale fading; Fading effects due to multipath time delay spread, flat fading, frequency selective fading.

MODULE - III: CELLSITEANDMOBILEANTENNAS (10)

Sum and difference patterns and their synthesis, omni directional antennas, directional antennas for interference reduction, space diversity antennas, umbrella pattern antennas, minimum separation of cell site antennas, high gain antennas, Numbering and grouping, setup access and paging channels channel assignments to cell sites and mobile units, channel sharing and borrowing, sectorization, overlaid cells, non-fixed channel assignment.

Handoff, dropped calls and cell splitting, types of handoff, handoff invitation,delaying handoff, forced handoff, mobile assigned handoff. Intersystem handoff, cell splitting, micro cells, vehicle locating methods, dropped call rates and their evaluation, channel planning for wireless systems, Indoor propagation models-partition losses (Same Floor), partition losses between floors, log- distance path loss model.

MODULE - IV: WIRELESSSYSTEMSANDSTANDARDS (08)

Second generation and Third generation Wireless Networks and Standards, WLL, Bluetooth, GSM, IS95, DECT, GSM architecture, GSM channels, multiplex access scheme, TDMA, CDM.

MODULE - V: INTELLIGENT NETWORK FOR WIRELESSCOMMUNICATIONS (08)

Intelligent cell concept, advanced intelligent network, SS7 network and ISDN for AIN, AIN for mobile communication, Common channel signaling, asynchronous transfer mode technology, future public land mobile telecommunication system, wireless information superhighway, Gateway, TCP/IP Model and the OSI Network Model.

IV. TEXTBOOKS:

1. Theodore.S.Rapport, “Wireless Communications”, Pearson Education, 2nd Edition, 2010.
2. Upen Dalal,“Wireless Communication”, Oxford University Press, 2010.
3. Kaveh Pahlvan, Prashant Krishnamurthy,“Principle of wireless networks”, A United Approach”, Pearson Education, 2004.
4. Andrea Goldsmith,“Wireless Communications”, Cambridge University Press, 2005.

V. REFERENCEBOOKS:

1. Theodore.S.Rapport, “Wireless Communications", Pearson Education, 3rd Edition 2003.
2. Lee, “Wireless and Mobile Communications”, McGraw Hill, 3rd Edition, 2006.
3. Jon W. Mark and Weihua Zhqung, “Wireless Communication and Networking”, PHI, 1st Edition, 2005.
4. R.Blake, “Wireless Communication Technology", Thompson Asia Pvt.Ltd., 1st Edition, 2004.

VI. WEBREFERENCES:

1. https://accessengineeringlibrary.com
2. http://www.radio-electronics.com
3. https://www.jntubook.com
4. http://www.iare.ac.in

VII. E-TEXTBOOKS:

1. http://www.iitg.ernet.in/scifac/qip/public_html/cd_cell/EC632.pdf
2. https://books.google.co.in/books/about/Cellular_and_Mobile_Communications
3. https://technicalpublications.org/.../books/CellularandMobileCommunications

OPTICAL COMMUNICATIONS

V Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC23	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisites: Analog and Digital Communications								

I. COURSE OVERVIEW:

Optical Communication permits the communication of data in the form of audio and video signal over transparent medium with high data rate to long distances. It covers types of optical fibers, signal degradation, optical detectors, optical amplifiers and optical networks. The applications are used in medical, defense, internet communication and cable television signals.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The types, principles and components used in optical communication.

II. Noise and signal degradation sources in optical communication system.

III. The function of optical network system, synchronous optical networking / Synchronous digital hierarchy, wavelength-division multiplexing, Solitons.

III. COURSE SYLLABUS:

MODULE-I :INTRODUCTION TO OPTICAL FIBERS (10)

Introduction-general optical fiber communication system- basic optical laws and definitions, propagation of light, propagation of light in a cylindrical dielectricrod; rays and modes; different types of optical fibers, mode analysis for optical propagation through fibers, modal analysis of a step index fiber, linearly polarized modes, single mode fibers and graded - index fiber.

MODULE -II: SIGNAL DEGRADATION AND OPTICAL SOURCES (09)

Attenuation- Absorption, scattering losses, bending losses, core and cladding losses; signal distortion in optical waveguides; Material Dispersion, Waveguide Dispersion; Optical sources; Semiconductor device fabrication, LED and LASER diode; Principles of operation, concepts of line width, phase noise, switching and modulation characteristics.

MODULE -III: OPTICAL SOURCES AND DETECTORS (08)

Sources: Intrinsic and extrinsic material-direct and indirect band gaps-LED-LED structures surface emitting LED-Edge emitting LED-quantum efficiency and LED power-light source materials-modulation of LED-LASER diodes.

Optical detectors: pin detector, avalanche photodiode - Principles of operation, concepts of responsively,sensitivity and quantum efficiency, noise in detection.

Multichannel Transmission Technique-Multichannel Frequency Modulation, Subcarrier multiplexing.WDM Concepts and Components

MODULE -IV: OPTICAL AMPLIFIERS (08)

Basic concepts, semiconductor amplifier, erbium-doped fiber amplifier, Raman amplifier, Brillouin amplifier - principles of operation, amplifier noise, signal to noise ratio, gain, gain bandwidth, gain and noise dependencies, inter modulation effects, saturation induced crosstalk, wavelength range of operation.

MODULE -V: OPTICAL NETWORKS AND DISPERSION COMPENSATION (10)

System design consideration Point – to –Point link design –Link power budget –rise time budget, SONET/SDH, ATM, IP, wavelength routed networks, soliton communication system, fiber soliton, soliton based communication system design, high capacity and WDM soliton.

IV. TEXT BOOKS:

1. Keiser. G, “Optical Fiber Communications”, Tata McGraw Hill, 4th Edition, New Delhi, 2008.
2. Agrawal. G.P, “Fiber-Optic Communication Systems”, John Wiley & Sons, 3rd Edition, 2002.

V. REFERENCE BOOKS:

1. John Gowar, “Optical Communication Systems”, Prentice Hall, 2nd Edition, 1993.
2. Franz, Jain, “Optical communication, Systems and Components”, Narosa Publications, 1st Edition New Delhi, 2000.
3. Karminvov, T. Li, “Optical Fiber Telecommunications”, Vol A & B, Academic Press, 2002.

VI. WEB REFERENCES:

1. http://nptel.ac.in
2. http://nptel.ac.in/courses
3. https://onlinecourses.nptel.ac.in

VII. E-TEXT BOOKS:

1. https://eceagmr.files.wordpress.com
2. http://www.slac.stanford.edu
3. https://www.utdallas.edu

SATELLITE COMMUNICATIONS

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC24	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Analog and Digital Communications

I. COURSE OVERVIEW:

Satellites form an essential part of telecommunications worldwide. This course will cover the fundamentals of satellite communications, LEO satellites, MEO satellites, GEO satellites, and orbits. Transponders on communication satellites, link budget calculations, multiple access techniques and the propagation of radio waves through the earth's atmosphere. This course also covered the growth of VSAT systems and internet packet communications in satellite.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basics of orbital mechanics, the types of satellite orbits, the location of groundstations, and the look angles from ground stations to the satellite.

II. Organization of communications between a satellite and a ground station (TTC and M, payload, data and channels), basic analysis of link budget and examples of calculations.

III. The application of the satellite communication in the fields of multi spectralremote sensing, resource management and mineral exploration.

III. COURSE SYLLABUS:

MODULE-I: COMMUNICATIONS SPACECRAFT AND ORBITS (10)

Definition, Basic Principles, Orbital parameters, overview of present and future trends of satellite communications introduction to satellite systems: Low earth orbit (LEO); Medium earth orbit (MEO); Geo synchronous earth orbit (GEO); Geostationary earth orbit (GEO); Orbital mechanics: Orbital elements; Orbital elements; Locating the satellite with respect to the earth; Coverage angle; Slant range; Inclined orbits; Orbital perturbations due to earth‘s oblateness and moon and sun; Eclipse of GEO satellite; Sun transit outage.

MODULE -II: SPACE SEGMENT (09)

Placement of a communication satellite in GEO satellite sub systems: Telemetry, tracking and command system, power system, satellite antenna equipment, communications subsystem and transponders, TWT amplifier operation, satellite frequency bands and allocations; Satellite link: Basic transmission theory, system noise temperature and G/T ratio, basic link analysis, design of satellite links for a specified C/N with and without frequency Re-use , link budget; Propagation effects: Introduction, atmospheric absorption, cloud attenuation, troposphere and ionospeheric scintillation and low angle fading; Effects of rain: Rain induced attenuation, rain induced cross polarization interference.

MODULE -III: COMMUNICATION SATTELLITE ACCESS SYSYTEMS (08)

Multiple Access: Frequency division multiple access (FDMA), Time division multiple access (TDMA), frame structure, burst structure, satellite switched TDMA, on-board processing, demand assignment multiple access (DAMA), types of demand assignment, characteristics.

Code Division Multiple Access (CDMA) / Spread Sprectrum Multiple Access (SSMA); Direct sequence CDMA (DS-CDMA) or DS spread spectrum transmission and reception, adjacent channel interference, inter modulation, handover, satellite diversity.

MODULE -IV: EARTH STATION AND VSAT SYSTEMS TECHNOLOGY (08)

Earth Station: Transmitters, receivers, antennas, tracking systems, terrestrial interface, power test methods, lower orbit considerations; VSAT (Very Small Aperture Terminal) Systems: Overview of VSAT systems, VSAT network architecture, access control, multiple access selection. NGSO constellation design: Orbits, coverage, frequency bands, delay and throughput, non geostationary orbit
(NGSO) constellation design and problems.

MODULE -V: SATELLITE PACKET COMMUNICATION (10)

Message transmission by FDMA: M/G/1 queue, message transmission by TDMA, pure aloha, satellite packet switching, slotted aloha, packet reservation, tree algorithm; Over view of future satellite communication systems, introduction to satellite laser communication, data relay communication satellites, satellite mobile services, applications. Classification of remote sensing systems, orbits, Payloads, Types of images: Image Classification, Interpretation, Applications. Development of Satellite Navigation Systems, GPS system, Applications.

IV. TEXT BOOKS:

1. Dennis roddy, “Satellite Communications”, 4th Edition, 2004.
2. Pratt. Bostian, Allnutt, “Satellite Communications”, Wiley India, 2nd Edition, 2006.
3. Gérard Maral, “Satellite Communication Systems”, 1993.

V. REFERENCE BOOKS:

1. Rappaport T.S., “Wireless Communications”, Pearson Education, 2nd Edition, 2010.
2. Bruce Elbert, “Introduction to Satellite Communication”, 1987.

VI. WEB REFERENCES:

1. http://nptel.ac.in/courses/106105082/33
2. https://onlinecourses.nptel.ac.in/noc16_ec10/preview

VII. E-TEXT BOOKS:

1. http://www4.zippyshare.com/v/72052755/file.html
2. http://www.jntumaterials.co.in/2015/07/satellite-communications-by-dennis-roddy.html
3. Dennis roddy, “Satellite Communications”, 4th Edition, 2004.
4. Pratt. Bostian, Allnutt, “Satellite Communications”, Wiley India, 2nd Edition, 2006.
5. Gérard Maral, “Satellite Communication Systems”, 1993.

WIRELESS COMMUNICATIONS AND NETWORKS

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC25	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Analog and Digital Communications

I. COURSE OVERVIEW

This course is intended to provide an overview of transmitting information from one point to another without using any connection like wires, cables or any physical medium. It covers the fundamentals of cellular communications, radio propagation, equalization, diversity and wireless networks. It focuses on performance analysis and design of a wireless communication system such as mobile telephone, satellite communication, TV and radio transmissions.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The basic concepts of frequency reuse, handoff, multipath channels and multiple access techniques used in wireless communication systems.

II. The process of fading mechanism, types of equalizers and diversity techniques.

III. The wireless network standards together with network protocols.

III. COURSE SYLLABUS:

MODULE-I: THE CELLULAR CONCEPT SYSTEM DESIGN FUNDAMENTALS (10)

Introduction, frequency reuse, channel assignment strategies, handoff strategies; Prioritizing handoffs, practical handoff considerations, interference and system capacity; Co-channel interference and system capacity, channel planning for wireless systems, adjacent channel interference, power control for reducing interference, trunking and grade of service, improving coverage & capacity in cellular systems; Cell splitting, sectoring.

MODULE-II: MOBILE RADIO PROPAGATION-LARGE-SCALE PATH LOSS (09)

Large-Scale Path Loss: Introduction to radio wave propagation, free space propagation model, relating power to electric field, the three basic propagation mechanisms; Reflection: Reflection from dielectrics, Brewster angle, reflection from prefect conductors, ground reflection (Two-Ray) mode; Diffraction Fresnel zone geometry, knife- edge diffraction model, multiple knife-edge diffraction, scattering, outdoor propagation models; Longley-Ryce model, Okumura Model, Hata Model, PCS extension to hata Model, Walfisch and Bertoni model, wideband PCS microcell model, indoor propagation models-partition losses (Same Floor), partition losses between floors, log- distance path loss model, ericsson multiple breakpoint model, attenuation factor model, signal penetration into buildings, ray tracing and site specific modeling.

MODULE-III: MOBILE RADIO PROPAGATION- SMALL -SCALE PATH LOSS (08)

Small-scale fading and multipath: Small scale multipath propagation; Factors influencing small scale fading, Doppler shift, impulse response model of a multipath channel; Relationship between bandwidth andreceived power, small; Scale multipath measurements; Direct RF pulse system, spread spectrum sliding correlator channel sounding, frequency domain channels sounding, parameters of mobile multipath channels; Time dispersion parameters.

Coherence Bandwidth, Doppler spread and coherence time, types of small - Scale fading; Fading effects due to multipath time delay spread, flat fading, frequency selective fading, fading effects due to Doppler Spread -Fast fading, slow fading, statistical models for multipath fading channels; Clarke model for flat fading, spectral shape due to Doppler spread in Clarke model, simulation of Clarke and Gans Fading model, level crossing and fading statistics, two-ray Rayleigh fading model.

MODULE-IV: EQUALIZATION AND DIVERSITY (08)

Introduction, fundamentals of equalization, training a generic adaptive equalizer, equalizers in a communication receiver, linear equalizers, non-linear equalization; Decision feedback equalization (DFE), maximum likelihood sequence estimation (MLSE) equalizer, algorithms for adaptive equalization; Zero forcing algorithm, least mean square algorithm, recursive least squares algorithm; Diversity techniques; Derivation of selection diversity improvement, derivation of maximal ratio combining improvement, practical space diversity consideration; Selection diversity, feedback or scanning diversity, maximal ratio combining, equal gain combining, polarization diversity, frequency diversity, time diversity, RAKE receiver.

MODULE-V: WIRELESS NETWORKS (10)

Introduction to wireless networks, advantages and disadvantages of wireless local area networks, WLAN topologies, WLAN standard IEEE 802.11, IEEE 802.11 medium access control, comparison of IEEE 802.11 a,b,g and n standards, IEEE 802.16 and its enhancements, wireless PANs, Hipper LAN, WLL.

IV. TEXT BOOKS:

1. Theodore .S. Rapport, “Wireless Communications”, Pearson Education, 2nd Edition, 2010.
2. Upen Dalal, “Wireless communication”, Oxford University press, 2010.
3. Kaveh Pahlvan, Prashant Krishnamurthy, “Principle of Wireless Networks”, A United Approach, Pearson Education, 2004.
4. Andrea Goldsmith, “Wireless Communications”, Cambridge University Press, 2005.

V. REFERENCE BOOKS:

1. P.Nicopolitidis, M.S. Obaidat, G.I.Papadimitria, A.S. Pomportsis,”Wireless Networks” John Wiley &sons, 1st Edition, 2003.
2. Vijay K Garg,”Wireless Communications and Networks”, Morgan Kaufmann Publishers an Imprint of Elsevier, USA 2009 (Indian Reprint).
3. Mark Ciampa Jorge Olenewa, “Wireless Communication and Networking”, IE, 2009.
4. X.Wang, H.V.Poor , “Wireless Communication System”, Pearson 2nd Education, 2004.
5. Jochen Schiller, “Mobile Communication”, Pearson Education, 2nd Edition, 2003.

VI. WEB REFERENCES:

1. http://keshi.ubiwna.org/2017IoTCOMM/Wireless_Communications_&_Networking_Stallings_2nd.
2. https://www.google.com/wirelesscommunicationnetwork.
3. https://www3.nd.edu/~mhaenggi/ee598q/books/stallings_jagadish.pdf

VII. E-TEXT BOOKS:

1. https://www.oreilly.com/library/view/wireless-communications-principles/0130422320/
2. https://groups.google.com/forum/#!topic/kluecm2010-2014/7Q5gRhqh51g

EXPERIENTIAL ENGINEERING EDUCATION (ExEEd) – PROJECT BASED LEARNING

V Semester: Common for all branches								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC20	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		2	-	-	1	30	70	100
Contact Classes: 36	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 36**		
Prerequisite: There are no prerequisites to take this course								

I. COURSE OVERVIEW:

Project-based learning (PBL) is collaborative, learner-centered instructional approach where students work in groups to construct their knowledge using modern tools. It often requires students to collaborate, design, revise, and share their ideas and experiences with authentic audiences and supportive peer groups rather than collect resources, organize work, and manage long-term activities. Project-Based Learning begins with the assignment of tasks that will lead to the problem identification, modeling, simulation and analyzing the results.

II. COURSE OBJECTIVES:

The students will try to learn:

I. To emphasize learning activities that is long-term, interdisciplinary and student-centric.
II. To inculcate independent learning by problem solving with social context.
III. To engages students in rich and authentic learning experiences.
IV. To provide every student the opportunity to get involved either individually or as a group so as to develop team skills and learn professionalism.

III. COURSE SYLLABUS

I. Defining the Problem
II. Gathering requirements
III. Design / *Mode*ling
IV. Implementation
V. Testing
VI. Report

VIRTUAL INSTRUMENTATION LABORATORY

V Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC30	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	-	3	1.5	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes: 36**		
Prerequisites: Electronic Measurements and Instrumentation								

I. COURSE OVERVIEW:

The Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a development environment designed by National Instruments that creates graphic-based programs called virtual instruments (VIs) that simulate actual laboratory instruments. The experimental objective of this lab is to use LabVIEW to design basic operations and data acquisition using myDAQ and myRIO cards. Building these systems will demonstrate the potential for using simulated instruments in a laboratory. These programs will also obtain data from outside the computer and incorporate it into a program design.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The basic applications and theory of the LabVIEW graphical programming environment.
II. The basic programming concepts in LabVIEW.
III. The different data acquisition system concepts.
IV. The real time applications using LabVIEW.
V. How to design, implement, and distribute stand-alone applications using LabVIEW.
VI. The single and multiple-loop design patterns for application functionality.

III. COURSE SYLLABUS:

Week – 1: OPEN AND RUN A VIRTUAL INSTRUMENT
Open the front panel and block diagram in Lab VIEW software

Week-2: SUM OF „n‟ NUMBERS USING „FOR‟ LOOP AND WHILE LOOP
FACTORIAL OF A GIVE NUMBER USING FOR LOOP AND WHILE LOOP
Design a program to find the sum of _n‘ numbers using FOR loop and WHILE loop
Design a program to perform the factorial of a given number using FOR loop and WHILE loop.

Week -3: BUNDLE AND UNBUNDLE CLUSTER
Design a program to bundle and unbundle a cluster.

Week-4: APPLICATION USING FORMULA NODE & DISCRETE COSINE TRANSFORM
Design a program to create a sine wave using formula node and to perform discrete cosine transformon the given signal.

Week-5: FLAT AND STACKED SEQUENCE
Design a program to perform functions using flat and stacked sequence.

Week-6: AMPLITUDE MODULATION
Design a program to perform Amplitude Modulation.

Week-7: REAL TIME TEMPERATURE MONITORING USING VIRTUAL INSTRUMENTATION.
Design a program for real time temperature monitoring by using virtual instrumentation

Week-8: MEASURE DISTANCE USING IR RANGER AND MYDAQ
Design a program for measure distance using ir ranger and myDAQ

Week-9: MEASUREMENT OF VIBRATIONS USING PIZEO ELECTRIC TRANSDUCER AND MYDAQ
Design a program for measurement of vibrations using pizeo electric transducer and myDAQ
Week-10: MEASUREMENT OF VIBRATIONS USING PIZEO ELECTRIC TRANSDUCER AND MYRIO

Design a program for measurement of vibrations using pizeo electric transducer and myRIO

Week-11: INTERFACE SERVO MOTOR AND DC MOTORS USING MYDAQ
Acquire the data from the sensors by using myDAQ and myRIO

Week-12: INTERFACE SERVO MOTOR AND DC MOTORS USING MYRIO
Design a program to interface servo motor and dc motors using myRIO

Week-13: MEASURE DISTANCE USING IR RANGER AND MYRIO
Design a program to develop signal generator by using myRIO cards

Week-14: DEVELOPING SIGNAL GENERATOR USING DAQ CARDS
Design a program to develop signal generator by using myDAQ cards

IV. REFERENCE BOOKS:

1. Jim Kring, Jeffrey Travis , "LabVIEW for Everyone: Graphical Programming Made Easy and Fun", Prentice Hall, 3rd Edition, 2006.
2. Richard Jennings Gary W.Johnson, "Labview Graphical Programming", McGraw-Hill Education, 4th Edition, 2011.
3. Rick Bitter, Taqi Mohiuddin,, Matt Nawrocki, "LabView: Advanced Programming Techniques", CRC Press, 2nd Edition, 2006.
4. Sanjay Gupta, "Virtual Instrumentation using LABVIEW", McGraw-Hill Education, 2nd Edition, 2010.

V. WEB REFERENCE BOOKS:

1. http://www.ni.com/pdf/manuals/373427j.pdf
2. http://home.hit.no/~hansha/documents/labview/Introduction%20to%20LabVIEW.htm
3. https://www.pearsonhighered.com/samplechapter/0130153621.pdf
4. http://k12lab-support-pages.s3.amazonaws.com/lvbasichome1.html

MICROPROCESSORS AND MICROCONTROLLERS LABORATORY

V Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC31	Core	L	T	P	C	CIA	SEE	Total
		-	-	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes: 36		

Prerequisites: Digital System Design

I. COURSE OVERVIEW:

This laboratory course will facilitates the students to program 8086 microprocessor and 8051 microcontroller. Win862 software will be used for writing and debugging assembly language programs. The course includes performing arithmetic and logical operations, string manipulations, code conversions and interfacing of I/O devices to processor/controller. The hands-on experience acquired by the student's during the course makes them to carry out processor/controller based projects and extend their knowledge on the latest trends and technologies in the field of embedded system.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. Assembly language programming skills ranging from simple arithmetic operations to interfacing real time systems.

II. The usage of software tools to design, debug and test microprocessor/microcontroller based projects using assembly language programming.

III. The design of microcomputer and microcontroller based real-time applications in the fields of communication systems, home based automation systems, automobiles and unmanned applications.

III. COURSE SYLLABUS:

Week-1 : DESIGN A PROGRAM USING WIN862

Design and develop an Assembly language program using 8086 microprocessor and to show the following aspects.

(a) Programming
(b) Execution
(c) Debugging
(d) To Demonstrate the win 862 software and Trainer kit for 8086 Microprocessor

Week -2 : 16 BIT ARITHMETIC AND LOGICAL OPERATIONS

Write an ALP program to perform 16 Bit arithmetic and logical operations using WIN862 software.

Week -3: MULTIBYTE ADDITION AND SUBTRACTION

(a) Write an ALP program to perform multi byte addition and subtraction
(b) Write an ALP program to perform 3*3 matrix multiplication and addition

Week -4 : PROGRAMS TO SORT NUMBERS

(a) Write an ALP program to perform ascending order using 8086
(b) Write an ALP program to perform descending order using 8086

Week -5: PROGRAMS FOR STRING MANIPULATIONS OPERATIONS

(a) Write an ALP program to insert or delete a byte in the given string
(b) Write an ALP program to search a number/character in a given string
(c) Write an ALP program to move a block of data from one memory location to the other
(d) Write an ALP program for reverse of a given string.

Week -6: CODE CONVERSIONS

(a) Write an ALP program to convert packed BCD to Unpacked BCD
(b) Write an ALP program to convert packed BCD to ASCII
(c) Write an ALP program to convert hexadecimal to ASCII

Week -7 : INTERFACING STEPPER MOTOR

(a) Write an ALP program to rotate stepper motor in clockwise direction
(b) Write an ALP program to rotate stepper motor in anti clockwise direction

Week -8 : INTERFACING ADC and DAC DEVICES
(a) Write an ALP program to convert analog to digital using 8086
(b) Write an ALP program to convert digital to analog using 8086

Week -9 : INTERFACING KEYBOARD TO 8086
Write an ALP program to interface keyboard to 8086

Week -l0: SERIAL AND PARALLEL COMMUNICATION
(a) Parallel communication between two microprocessors using 8255
(b) Serial communication between two microprocessor kits using 8251

Week -11: INTERFACING TRAFFIC LIGHT CONTROLLER AND TONE GENERATOR
(a) Write a program to interface traffic light controller
(b) Write an ALP program to interface tone generator

Week -12: ARITHMETIC AND LOGICAL OPERATIONS USING 8051
Write an ALP program to perform 16 Bit arithmetic and logical operations using 8051 microcontroller.

Week -13: TIMER/COUNTER
Write an ALP Program and verify Timer/Counter using 8051

Week -14 : INTERFACING KEYBOARD TO 8051
Write an ALP program to interface keyboard to 8051

IV. TEXT BOOKS:

1. Ray A.K, Bhurchandi K.M, “Advanced Microprocessor and Peripherals”, TMH, 2nd Edition, 2012.
2. Muhammad Ali Mazidi, J.G. Mazidi, R.D McKinlay, “The 8051 Microcontroller and Embedded Systems using Assembly and C”, Pearson education, 2nd Edition, 2009.
3. Douglas V. Hall, “Microprocessors and Interfacing Programming and Hardware”, TMGH, 2nd Edition, 1994.

V. REFERENCE BOOKS:

1. Kenneth J. Ayala, “The 8051 Microcontroller”, Thomson Learning, 3rd Edition, 2005.
2. Manish K. Patel, “The 8051 Microcontroller Based Embedded Systems”, McGraw Hill, 1st Edition, 2014.
3. Ajay V Deshmukh, “Microcontrollers”, TATA McGraw Hill Publications, 2nd Edition, 2012.

OBJECT ORIENTED PROGRAMMING DEVELOPMENT AND LANGUAGES

V Semester: **AE / ECE / EEE / ME / CE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC23	Skill	L	T	P	C	CIA	SEE	Total
		-	-	-	-	-	-	-
Contact Classes: Nil	Total Tutorials: Nil	Total Practical Classes: Nil				Total Classes: Nil		

I.COURSE OVERVIEW:

Java's unique architecture enables programmers to develop a single application that can run across multiple platforms seamlessly and reliably. This course, enable the students to gain extensive experience with Java and its object oriented features to create robust console and GUI applications and store and retrieve data from relational databases.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic object-oriented programming concepts and apply them in problem solving.
II. The inheritance concepts for reusing the program.
III. The programs to implement event handling, user interfaces and graphical interfaces with the help of Java.

III. COURSE SYLLABUS

MODULE-I: FUNDAMENTALS OF OBJECT-ORIENTED PROGRAMMING

Object oriented paradigm: Basic concepts of Object-Oriented Programming, Benefits of OOP, Applications of OOP; Java Evolution: Java Features, How Java differs from C and C++, Java and Internet,Java and World Wide Web, Web Browsers, Hardware and Software Requirements, Java Environment. Overview of Java Language: Simple Java Program, Java Program Structure, Java Tokens, Java Statements, Implementing a Java Program, Java Virtual Machine, Constants, Variables, Data types, Scope of Variables, Symbolic Constants, Type Casting and type promotions, Operators, Operator Precedence and Associativity, Control Statements, break, continue, Arrays-Multi dimensional arrays, Wrapper Classes, Simple examples.

MODULE-II: CLASSES AND OBJECTS

Classes and Objects, constructors, methods, this keyword, garbage collection, finalize, overloading methods and constructors, access control, static members, nested and inner classes, command line arguments, variable length arguments. Inheritance: Forms of inheritance, specialization, specification, construction, extension, limitation, combination, benefits and costs of inheritance. Super uses- final - polymorphism, method overriding - dynamic method dispatch, abstract classes, exploring String class.

MODULE-III: PACKAGES AND INTERFACES

Defining and accessing a package, understanding CLASSPATH, access protection importing packages, Interfaces: Defining and implementing an interface, Applying interfaces, Variables in interfaces and extended interfaces. Exploring java.lang and java.util packages.

Exception Handling: Fundamentals, usage of try, catch, multiple catch clauses, throw, throws and finally. Java Built in Exceptions and creating own exception subclasses.

MODULE- IV: MULTITHREADED PROGRAMMING

Java Thread life cycle model: Thread creation, Thread Exceptions, Thread Priority, Synchronization ,Messaging, Runnable Interface - Interthread Communication - Deadlock - Suspending, Resuming and stopping threads.

I/O Streams: File, Streams, Advantages, The stream classes, Byte streams, Character streams.

MODULE- V APPLET PROGRAMMING

Event handling: basics of event handling, Event classes, Event Listeners, delegation event model, handling mouse and keyboard events, adapter classes, AWT Class hierarchy, AWT Controls, Layout Managers and Menus, limitations of AWT. How Applets differ from Applications: Applet Life Cycle, Creating an Applet, Running the AppletDesigning a Webpage, Applet Tag, Adding Applet to HTML file, More about Applet Tag, Passing parameters to Applets, Aligning the display.

IV. TEXTBOOKS:

1. Herbert Schildt, "The Complete Reference Java J2SE", TMH Publishing Company Ltd, New Delhi, 5th Edition, 2008.
2. Cay Horstmann, "Big Java", John Wiley and Sons, 2nd Edition, 2006.

V. REFERENCES BOOKS:

1. H.M.Dietel and P.J.Dietel, "Java How to Program", Pearson Education/PHI, 6th Edition 2008.
2. Cay.S.Horstmann and Gary Cornell, "Core Java 2" Vol 1, Fundamentals", Pearson Education, 7th Edition, 2007.
3. Cay.S.Horstmann and Gary Cornell, "Core Java 2, Vol 2, Advanced Features", Pearson Education. 7th Edition, 2008.

VI. WEB REFERENCES:

1. http://www.javatpoint.com/java-tutorial
2. http://www.javatutorialpoint.com/introduction-to-java/

VII. E-Text Books:

1. http://bookboon.com/en/java-programming-language-ebooks
2. https://en.wikibooks.org/wiki/Java_Programming

MICROWAVE AND RADAR ENGINEERING

VI Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC32	Core	L	T	P	C	CIA	SEE	Total
		3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisites: Electromagnetic Waves and Transmission Lines

I. COURSE OVERVIEW:

This course allows students to study and analyze microwave and radar systems at high frequencies, typically in the MHz and GHz range where lumped elements (e.g., resistors, capacitors, inductors) are no longer appropriate. It introduces the concepts waveguides, components, microwave tubes and radar transmitters & receivers. The applications include cellular communications, high-speed digital and analog circuits, wireless networks and radar.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of wave guide components and electromagnetic wave propagation for microwave communication using Maxwell's equations.

II. The generation of microwave signals to measure different parameters using microwave test bench.

III. The principle and operation of radar systems and radar range equation for communication.

III. COURSE SYLLABUS:

MODULE – I: WAVEGUIDES AND COMPONENTS (08)

Introduction, microwave spectrum and bands, applications of microwaves, types of waveguides, rectangular waveguides, field equations in rectangular waveguide, field components of TM and TE waves for rectangular waveguide, modes of TM and TE waves in rectangular waveguide, impossibility of TEM waves, cut off frequency of rectangular waveguide; Wave impedance in rectangular waveguide: Wave impedance for a TM and TE wave in rectangular waveguide, Dominant mode and degenerate modes, mode characteristics of phase velocity, group velocity, wavelength and impedance relations; waveguide multiport junctions: E plane Tee, H plane Tee, Magic Tee, applications of Magic Tee, hybrid ring; Ferrites: Faraday rotation principle, gyrator, isolator, circulator illustrative problems.

MODULE – II :MICROWAVE LINEAR BEAM AND CROSS FIELD TUBES (OTYPE AND MTYPE (10)

Microwave linear beam tubes (O type): Limitations of conventional tubes at microwave frequencies; Klystron: Velocity modulation process, bunching process, output power and beam loading; Multicavity Klystron amplifiers: Beam current density, output current and output power of two cavity Klystron; Reflex Klystron: Velocity modulation, power output and efficiency. Helix Traveling Wave tube: Slow wave structures, amplification process, conventional current; Microwave cross field tubes (M type): Introduction, cross-field effects; Magnetrons: Different types, 8-cavity cylindrical travelling wave Magnetron, Hull cut-off and Hartree conditions, modes of resonance and PI-mode operation.

MODULE – III: MICROWAVE MEASUREMENTS AND CW AND PULSE RADAR (10)

Description of microwave bench: Different blocks and their features, precautions; Microwave power measurement: Bolometer; Measurement of attenuation; Frequency standing wave measurements: measurement of low and high VSWR; Cavity Q; Impedance measurements.

Radar Range equation; Pulse Radar: Block diagram and Operation; Maximum unambiguous range; Radar wave forms; Prediction of Target range; Integration of echo pulses, PRF and Range ambiguities; system losses. CW Radar: Introduction, Block Diagram, Isolation between transmitter and receiver, Non-zero IF receiver, Receiver bandwidth requirements, Applications; Frequency Modulated CW radar: Range and Doppler measurement, Mathematical Analysis, Block Diagram and characteristics, FM-CW altimeter, multiple frequency CW radar, Ambiguity Diagram & its application.

MODULE – IV: RADAR DETECTION IN NOISE (08)

Moving target indication (MTI) on A scope, butterfly effect, MTI using delay line canceller (DLC), Doppler measurement using Pulse radar, MTI radar (with power amplifier transmitter), MTI radar (with power oscillator transmitter), filter characteristics of DLC, blind speeds, double DLCs, Blind speeds, Staggered PRFs. Matched Filter (MF) receiver, MF response characteristics; Correlation Receiver, Efficiency of non-matched filters, Matched filter with non-white noise, Automatic Detection of radar signals: Tapped Delay Line (TDL) detection, CFAR receiver, Radar Clutter: Land and Sea clutter (without mathematical treatment)

MODULE – V: RADAR TRANSMITTERS & RECEIVERS (09)

Hybrid Linear-Beam Amplifier and Crossed-Field Amplifiers, Solid State Sources & Amplifiers, Methods for employing solid-state transmitters. Receiver Noise Figure (NF) - Noise Temperature; Measurement of NF, NF of Mixers, Basics of Radar Displays and Duplexers.

IV. TEXT BOOKS:

1. M. Kulkarni, “Microwave and radar engineering “, Umesh Publications,5th Edition,2016.
2. Samuel Y. Liao, “Microwave Devices and Circuits”, Pearson, 3rd Edition, 2003.
3. Merrill I Skolnik, “Introduction to Radar Systems”, TMH Special Indian Edition, 2nd Edition, 2007.

V. REFERENCE BOOKS:

1. Herbert J. Reich, J.G. Skolnik, P.F. Ordung and H.L. Krauss, “Microwave Principles, CBS Publishers and Distributors, New Delhi, 1st Edition, 2004.
2. F.E. Terman, “Electronic and Radio Engineering”, Tata McGraw-Hill Publications, 4th Edition, 1955.
3. Warren L. Stutzman, Gary A. Thiele, “Antenna Theory and Design”, 3rd Edition, 2012.

VI. WEB REFERENCES:

1. https://www. montana.edu/aolson/ee433/EE43308_L1-3.pdf
2. https://www.microwaves101.com/uploads/MESA-front.pdf
3. https:// www.onlinecourses.nptel.ac.in/noc20_ee63/preview
4. https://www.iare.ac.in

VII. E-TEXT BOOKS:

1. https://www.technicalsymposium.com/allenggebooks.html
2. https:// www.gradeup.co/best-books-for-microwave-engineering
3. https://www.aliexpress.com/item/EBOOK..Microwave Engineering

DIGITAL SIGNAL PROCESSING

VI Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC33	Core	L	T	P	C	CIA	SEE	Total
		3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisites: Signals and Systems

I. COURSE OVERVIEW:

This course provides the design of discrete-time systems and analytical tools to analyze the discrete signals and systems. It focuses on the classification of discrete-time signals and systems, linear time invariant systems, discrete fourier transform, fast fourier transform algorithms, digital filter design and multi rate signal processing. Digital signal processing applications are used in speech processing, image processing, audio and video data compression, communication systems.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The representation, classification and analysis of discrete time signals and systems in time and frequency domain.

II. The design and realization structures of finite and infinite impulse response filters and multi rate filters.

III. The implementation of digital filter algorithms using MATLAB tool.

III. COURSE SYLLABUS:

MODULE – I: REVIEW OF DISCRETE TIME SIGNALS AND SYSTEMS: (09)

Discrete time signal definition; Signal classification; Elementary signals; Transformation of elementary signals; Concept of digital frequency; Discrete time system definition; System classification; Linear time invariant (LTI) system; Properties of the LTI system; Time domain analysis of discrete time systems; Impulse response; The convolution sum; Methods of evaluating the convolution sum; Filtering using overlap-save and overlap-add method; Realization of digital filters: Concept of IIR and FIR filters; Realization structures for IIR and FIR filters using direct form-I and direct form-II, cascade, lattice and parallel.

MODULE – II: DISCRETE FOURIER TRANSFORM AND EFFICIENT COMPUTATION (09)

Introduction to discrete time Fourier transform (DTFT); Discrete Fourier transform (DFT) definition; Properties of DFT; Linear and circular convolution using DFT; Fast-Fourier-transform (FFT): Direct computation of DFT; Need for efficient computation of the DFT (FFT algorithms); Radix-2 FFT algorithm for the computation of DFT and IDFT using decimation-in-time and decimation-in-frequency algorithms; General Radix-N FFT.

MODULE – III: STRUCUTRE OF IIR FILTERS: (09)

Analog filters: Butterworth filters; Chebyshev type-1 and type-2 filters; Analog transformation of prototype LPF to HPF/BPF/BSF.

Transformation of analog filters into equivalent digital filters using impulse invariant method and bilinear transform method; Matlab programs of IIR filters.

MODULE – IV: SYMMETRIC AND ANTISYMMETRIC FIR FILTERS (09)

Design of linear phase FIR filters windowing and frequency sampling methods; Equiripple linear phase FIR filters; Parks-McClellan algorithm and remez algorithm; Least-mean-square error filter design; Design of FIR differentiators; Matlab programs of FIR filters; Comparison of FIR and IIR.

MODULE – V: APPLICATIONS OF DSP: (09)

Multirate signal processing; Decimation; Interpolation; Polyphase structures for decimation and interpolation filters; Structures for rational sampling rate conversion; Applications of multirate signal processing for design of phase shifters, interfacing of digital systems with different sampling rates, sub band coding of speech signals. Analysis of finite word length effects: Representation of numbers; ADC quantization noise, coefficient quantization error, product quantization error, truncation and rounding errors; Limit cycle due to product round-off error; Round-off noise power; Limit cycle oscillations due to overflow in digital filters; Principle of scaling; Dead band effects.

IV. TEXT BOOKS:

1. John G. Proakis, Dimitris G. Manolakis, “Digital Signal Processing, Principles, Algorithms and Applications”, Prentice Hall, 4th Edition, 2007
2. Sanjit K Mitra, “Digital Signal Processing, A Computer Base Approach”, McGraw-Hill Higher Education, 4th Edition, 2011.
3. Emmanuel C, Ifeacher, Barrie. W. Jervis, “DSP-A Practical Approach”, Pearson Education, 2nd Edition, 2002.
4. A.V. Oppenheim, R.W. Schaffer, “Discrete Time Signal Processing”, PHI, 2nd Edition, 2006.

V. REFERENCE BOOKS:

1. Li tan, “Digital Signal Processing: Fundamentals and Applications”, Elsevier Science and Technology Books, 2nd Edition, 2008.
2. Robert J.schilling, Sandra. L.harris, “Fundamentals of Digital Signal Processing using Matlab”, Thomson Engineering, 2nd Edition, 2005.
3. Salivahanan, Vallavaraj, Gnanapriya, “Digital Signal Processing”, McGraw-Hill Higher Education, 2nd Edition, 2009.

VI. WEB REFERENCES:

1. www.edufind.com
2. https://www.tutorialspoint.com/digital_signal_processing/index.htm

VII. E-TEXT BOOKS:

1. https://users.dimi.uniud.it/~antonio.dangelo/MMS/materials/Guide_to_Digital_Signal_Process.pdf

BUSINESS ECONOMICS AND FINANCIAL ANALYSIS

IV Semester: CSE / CSIT / CSE(DS), CSE(CS)

V Semester: AE / CE / EEE | VI Semester: ECE / ME / IT / CSE(AI&ML)

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AHSC13	Foundation	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisite: There is no prerequisite is required to this course								

I. COURSE OVERVIEW:

The course is designed in such a way that it gives an overview of concepts of Economics. Managerial Economics enables students to understand micro environment in which markets operate how price determination is done under different kinds of competitions. Financial Analysis gives clear idea about concepts, conventions and accounting procedures along with introducing students to fundamentals of ratio analysis and interpretation of financial statements. Break Even Analysis is very helpful to the Business Concern for Decision Making, controlling and forward Strategic Planning. Ratio analysis gives an idea about financial forecasting, financial planning, controlling the business and decision making.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The concepts of business economics and demand analysist o helps in optimal decision making in business environment**.**

II. The functional relationship between Production and factors of production and able to compute breakeven point to illustrate the various uses of breakeven analysis.

III. The features, merits and demerits of different forms of business organizations existing in the modern business environment and market structures.

IV. The concept of capital budgeting and allocations of the resources through capital budgeting methods and compute simple problems for project management.

V. Various accounting concepts and different types of financial ratios for knowing financial positions of business concern.

III. COURSE OBJECTIVES:

MODULE – I: INTRODUCTION AND DEMAND ANALYSIS (07)

Definition, nature and scope of business economics; Demand analysis; Demand determinants, law of demand and its exceptions; Elasticity of demand: Definition, types, measurement and significance of elasticity of demand, demand forecasting, factors governing demand forecasting.

MODULE – II: PRODUCTION AND COST ANALYSIS (10)

Production function; Isoquants and isocosts, MRTS, least cost combination of inputs, Cobb-Dougles production function, internal and external economies of scale, cost analysis; Cost concepts: Break even analysis (BEA), determination of break-even point (simple problems), managerial significance.

MODULE – III: MARKETS AND NEW ECONOMIC ENVIRONMENT (08)

Types of competition and markets, features of perfect competition, monopoly and monopolistic competition, price-output determination in case of perfect competition and monopoly business.

Features and evaluation of different forms of business organizations: Sole proprietorship, partnership, joint stock company, public enterprises and their types.

MODULE – IV: CAPITAL BUDGETING (10)

Capital and its significance, types of capital, estimation of fixed and working capital requirements, methods and sources of raising capital, capital budgeting: features of capital budgeting proposals; Methods of capital budgeting: Payback period, accounting rate of return(ARR), net present value method and internal rate of return method (simple problems).

MODULE – V: INTRODUCTION TO FINANCIAL ACCOUNTING AND FINANCIAL ANALYSIS (10)
Financial accounting objectives, functions, importance; Accounting concepts and accounting conventions -double-entry book keeping, journal, ledger, trial balance; Final accounts: Trading account, profit and loss account and balance sheet with simple adjustments; Financial analysis: Analysis and interpretation of liquidity ratios, activity ratios, capital structure ratios and profitability ratios (simple problems), Du Pont chart.

IV. TEXT BOOKS:

1. Aryasri, "Managerial Economics and Financial Analysis", TMH publications, 4th Edition, 2012.
2. M. Kasi Reddy, Saraswathi, "Managerial Economics and Financial Analysis", PHI Publications, New Delhi, 2nd Edition, 2012.
3. Varshney, Maheswari, "Managerial Economics", Sultan Chand Publications, 11th Edition, 2009.

V. REFERENCE BOOKS:

1. S. A. Siddiqual, A. S. Siddiqual, "Managerial Economics and Financial Analysis", New Age International Publishers, Hyderabad, Revised 1st Edition, 2013.
2. S. N. Maheswari, S. K. Maheswari, "Financial Accounting", Vikas publications, 3rd Edition, 2012.
3. J. V. Prabhakar Rao, P. V. Rao, "Managerial Economics and Financial Analysis", Maruthi Publishers, Reprinted Edition, 2011.
4. Vijay Kumar, Appa Rao, "Managerial Economics and Financial Analysis", Cengage Publications, 1st Edition, Paperback, 2011.

VI. WEB REFERENCES:

1. https:// www.slideshare.net/glory1988/managerial-economics-and- financial analysis
2. https:// thenthata.web4kurd.net/mypdf/managerial-economics-and- financial analysis
3. https:// bookshallcold.link/pdfread/managerial-economics-and-financial analysis
4. https:// www.gvpce.ac.in/syllabi/Managerial Economics and financial analysis

INFORMATION THEORY AND CODING TECHNIQUES

VI Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC34	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Probability Theory and Stochastic Processes								

I. COURSE OVERVIEW:

Information theory and coding is the study of the quantification, storage and Communication of digital information with the properties of codes for specific applications. These courses covers classifications of error control coding and source coding techniques, coding algorithms for audio, speech, image and video compression techniques. The applications include cryptography, error detection and correction in digital communication systems.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The concepts, principles and applications of information theory on communication systems.
II. The data compression techniques with text, audio, speech, image and video for real world applications
III. The block codes and convolutional codes for coding and decoding of digital data.

III. COURSE SYLLABUS:

MODULE-I: INFORMATION THEORY (09)

Information – Entropy, information rate, classification of codes, Kraft McMillan inequality, source coding theorem, mathematical model of information, a logarithmic measure of information, Shannon-Fano coding, Huffman coding, extended Huffman coding–Joint and conditional entropies, mutual information, discrete memory less channels–BSC, BEC–Channel capacity, Shannon limit.

MODULE-II: ERROR CONTROL CODING: BLOCK CODES (09)

Definitions and Principles: Hamming weight, Hamming distance, minimum distance decoding, single parity codes, Hamming codes, repetition codes, linear block codes, cyclic codes, syndrome calculation, shortened cyclic codes, majority logic decoding for cyclic codes, encoder and decoder, CRC.

MODULE-III: ERROR CONTROL CODING: CONVOLUTIONAL CODES (09)

Convolutional codes – code tree, trellis, state diagram - Encoding Decoding: Sequential search and Viterbi algorithm.

Principle of turbo coding, types of errors, error control strategies.

MODULE-IV: SOURCE CODING: TEXT, AUDIO AND SPEECH (09)

Source code: Definition, techniques, Text: Adaptive Huffman coding, arithmetic coding, variable-length codes, LZW algorithm – Audio: Linear predictive **coding (LPC),** Perceptual coding, masking techniques, psychoacoustic model, MEG audio layers I,II,III, Dolby AC3 - Speech: Channel vocoder, linear predictive coding.

MODULE-V: SOURCE CODING: IMAGE AND VIDEO (09)

Image and Video Formats – GIF, TIFF, SIF, CIF, QCIF – Image compression: READ, JPEG – Video Compression: Principles-I,B,P frames, motion estimation, motion compensation, H.261, MPEG standard, standards-based and nonstandard approaches to coding.

IV. TEXT BOOKS:

1. R Bose, “Information Theory, Coding and Cryptography”, TMH 2007.
2. Fred Halsall, “Multidedia Communications: Applications, Networks, Protocols and Standards”, Perason Education Asia, 2002.

V. REFERENCE BOOKS:

1. K Sayood, “Introduction to Data Compression” Elsevier, 3 Edition, 2006

2. S Gravano, “Introduction to Error Control Codes”, Oxford University Press, 2007.
3. Amitabha Bhattacharya, “Digital Communication”, TMH 2006.

VI. WEB REFERENCES:

1. https://www.youtube.com/watch?v=Uk9zFrEGguM
2. https://lecturenotes.in/subject/540/information-theory-coding-itc

VII. E-TEXT BOOKS:

1. http://web.stanford.edu/class/ee376a/files/scribes/lecture_notes.pdf
2. http://www.everythingvtu.wordpress.com

SOFTWARE RADIO AND COGNITIVE RADIO

VI Semester: **ECE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AECC35	**Elective**	3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		

Prerequisites: Analog and Digital Communications

I. COURSE OVERVIEW:

Software-based approaches enable engineers to build wireless system radios that are easier to manufacture, more flexible, and more cost effective. Software defined radio (SDR) and cognitive radio (CR) review the techniques, challenges, and tradeoffs of DSP software design. Coverage includes constructing RF front-ends; using digital processing to overcome RF design problems; direct digital synthesis of modulated waveforms; A/D and D/A conversions; smart antennas; object-oriented software design ; and choosing among DSP microprocessors, FPGAs, and ASICs.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The radio frequency translation for software defined radio.
II. The spectrum scarcity problem and how cognitive radio deals with this problem.
III. The technologies to allow an efficient use of TVWS for radio communications.
IV. The various research challenges for deployment of cognitive radio.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION TO SOFTWARE DEFINED RADIO (08)

Brief History, What is a Software-Defined Radio?, Networking and SDR, RF architectures for SDR, Processing architectures for SDR, Software Environments for SDR.

MODULE –II: RADIO FREQUENCY IMPLEMENTATION ISSUES (09)

The purpose of the RF Front-End, Dynamic range: The principal challenge of receiver design. RF receiver front-end topologies, Enhanced flexibility of the RF Chain with Software Radios, Importance of the components to overall performance, Transmitter architectures and their Issues, noise and distortion in the RF Chain, ADC and DAC distortion.

MODULE –III: ANALOG TO DIGITAL AND DIGITAL TO ANALOG (10)

Parameters of ideal data converters, Parameters of practical data converters, Techniques to improve data converter performance, Sigma-Delta Structures: ADC and DAC.

Applications for Software-Defined Radio: Cognitive Radio, Bumblebee Behavioral Model, Reinforcement Learning, Vehicular Networking.

MODULE –IV: COGNITIVE RADIO: TECHNIQUES AND SIGNAL PROCESSING (10)

History and background, Communication policy and Spectrum Management, Cognitive radio cycle, Cognitive radio architecture, SDR architecture for cognitive radio, Spectrum sensing Single node sensing: energy detection, cyclostationary and wavelet based sensing- problem formulation and performance analysis based on probability of detection versus SNR. Cooperative sensing: different fusion rules, wideband spectrum sensing- problem formulation and performance analysis based on probability of detection vs SNR.

MODULE –V: COGNITIVE RADIO: HARDWARE AND APPLICATIONS (08)

Hardware platforms for Cognitive radio (USRP and WARP), Details of USRP board, Cognitive wireless communication applications.

IV. TEXT BOOKS:

1. Travis F. Collins, Robin Getz, Di Pu, Alexander M. Wyglinski, “Software-Defined Radio for Engineer”, Artech House, 2018.
2. J.H. Reed, “Software-Defined Radio - A Modern Approach to Radio Engineering” Prentice-Hall, 2002.
3. Hüseyin Arslan, “Cognitive Radio, Software Defined Radio and Adaptive Wireless Systems”, Springer, ISBN 978-1-4020-5541-6 (HB), 2007.

V. REFERENCE BOOKS:

1. RF and Baseband Techniques for Software Defined Radio, Peter B. Kenington, Artech House, 2005.
2. Implementing Software Defined Radio- Eugene Grayver Springer, 2013.
3. Cognitive Radio Technology - Bruce A. Fette,, Elsevier, 2006.

MOBILE COMMUNICATION SYSTEM

VI Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC36	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Wireless Communications and Networks

I. COURSE OVERVIEW:

Mobile communication system allows people to communicate without utilizing any physical connection. It covers the basic cellular mobile system, frequency reuse, coverage, handoff mechanism, channel assignments of the cellular system and Ad hoc wireless networks. The applications include Wi-Fi, Bluetooth, cell phones, GPS and satellite television.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The cellular mobile system, cell coverage and frequency reuse concept for improving system coverage.
II. The various Handoff strategies in mobile communication system for preventing loss of interruption of service.
III. The development of Ad hoc wireless network between two or more wireless devices without establishing substantial network infrastructure.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION TO CELLULAR MOBILE RADIO SYSTEMS (09)

Limitations of Conventional Mobile Telephone Systems. Basic Cellular Mobile System, First, Second, Third and Fourth Generation Cellular Wireless Systems. Uniqueness of Mobile Radio Environment-Fading-Tie Dispersion Parameters, Coherence Bandwidth, Doppler Spread and Coherence Time.

Fundamentals of Cellular Radio System Design: Concept of Frequency Reuse, Co-Channel Interference, Co-Channel Interference Reduction Factor, Desired C/I from a Normal Case in a Omni Directional Antenna System, System Capacity Improving Coverage and Capacity in Cellular Systems Cell Splitting, Sectoring, Microcell Zone Concept.

MODULE –II: CO-CHANNEL INTERFERENCE AND NON CO-CHANNEL INTERFERENCE (09)

Measurement of Real Time Co-Channel Interference, Design of Antenna System, Antenna Parameters and their effects, diversity techniques-space diversity, polarization diversity, frequency diversity, time diversity.

Non Co-Channel Interference: Adjacent Channel Interference, Near end far end interference, cross talk, effects on coverage and interference by power decrease, antenna height decrease, effects of cell site components.

MODULE –III: CELL COVERAGE FOR SIGNAL AND TRAFFIC (09)

Signal Reflections in flat and Hilly Terrain, effects of Human Made Structures, phase difference between direct and reflected paths, constant standard deviation, straight line path loss slope, general formula for mobile propagation over water and flat open area, near and long-distance propagation, path loss from a point to point prediction model in different conditions, merits of lee model.

Frequency Management and Channel Assignment: Numbering and Grouping, Setup Access and Paging Channels, Channel Assignments to Cell Sites and Mobile Units.

MODULE –IV: HANDOFFS AND DROPPED CALLS (09)

Handoff Initiation, types of Handoff, Delaying Handoff, advantages of Handoff, Power Difference Handoff, Forced Handoff, Mobile Assisted and Soft Handoff, Intersystem handoff, Introduction to Dropped Call Rates and their Evaluation.

MODULE –V: AD HOC WIRELESS NETWORKS (09)

Introduction, Cellular and Ad Hoc wireless Networks, Applications and Ad Hoc Wireless Networks, Issues in Ad Hoc Wireless Networks, Ad Hoc Wireless Internet, MAC Protocols for Ad Hoc Wireless, Introduction, issues in designing AMAC Protocol for Ad Hoc wireless Networks, Design Goals of AMAC protocol for Ad Hoc Wireless Networks, Classification of MAC Protocols.

IV. TEXT BOOKS:

1. W.C.Y. Lee, “Mobile Cellular Telecommunications”, McGraw Hill, 2nd Edition, 1989.
2. Theodore. S. Rapport, “Wireless Communications”, Pearson Education, 2nd Edition, 2002.

V. REFERENCE BOOKS:

1. C. Siva ram Murthy and B.S. Manoj, “Ad Hoc Wireless Networks: Architectures and Protocols”, PHI, 2004.
2. Simon Haykin, Michael Moher, “Modern Wireless Communications”, Pearson Education, 2005.
3. Vijay Garg, “Wireless Communications and Networking”, Elsevier Publications, 2007.
4. Andrea Goldsmith, “Wireless Communications”, Cambridge University Press, 2005.

VI. WEB REFERENCES:

1. http:// www.radio-electronics.com
2. https://accessengineeringlibrary.com
3. https://www.jntubook.com
4. http://www.iare.ac.in

COMPUTER ORGANIZATION

VI Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC37	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Digital System Design								

I. COURSE OVERVIEW:

This course intended to provide the structure, internal working and implementation of a computer system. The fundamentals of various functional units of computer, computer instructions, addressing modes, computer arithmetic and logic unit, registers, data transfer, memory and input output system. It focuses on analysis of computer performance and functioning in modern computers.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic concepts of the various functional units and characteristics of computer systems.
II. The concepts of central processing unit design and perform basic operations with signed and unsigned integers in decimal and binary number systems.
III. The function of each element of a memory hierarchy and compare the different methods for computer input and output.

III. COURSE SYLLABUS:

MODULE - I: INTRODUCTION TO COMPUTER ORGANIZATION (09)

Basic computer organization, CPU organization, memory subsystem organization and interfacing, input or output subsystem organization and interfacing, simple computer levels of programming languages, assembly language instructions, and a simple instruction set architecture.

MODULE - II: ORGANIZATION OF A COMPUTER (09)

Register transfer: Register transfer language, register transfer, bus and memory transfers, arithmetic micro operations, logic micro operations, and shift micro operations; Control memory.

MODULE - III: CPU AND COMPUTER ARITHMETIC (10)

CPU design: Instruction cycle, data representation, memory reference instructions, input- output, and interrupt, addressing modes, data transfer and manipulation, program control.

Computer arithmetic: Addition and subtraction, floating point arithmetic operations, decimal arithmetic unit.

MODULE - IV: INPUT-OUTPUT ORGANIZATION AND MEMORY ORGANIZATION (09)

Memory organization: Memory hierarchy, main memory, auxiliary memory, associative memory, cache memory, virtual memory; Input or output organization: Input or output Interface, asynchronous data transfer, modes of transfer, priority interrupt, direct memory access.

MODULE - V: MULTIPROCESSORS (08)

Pipeline: Parallel processing, pipelining-arithmetic pipeline, instruction pipeline; Multiprocessors: Characteristics of multiprocessors, inter connection structures, inter processor arbitration, inter processor communication and synchronization.

IV. TEXTBOOKS:

1. M. Morris Mano, “Computer Systems Architecture”, Pearson, 3rd Edition, 2015.
2. Patterson, Hennessy, “Computer Organization and Design: The Hardware/Software Interface”, Morgan Kaufmann, 5th Edition, 2013.

V. REFERENCE BOOKS:

1. John. P. Hayes, “Computer System Architecture”, McGraw-Hill, 3rd Edition, 1998.
2. Carl Hamacher, Zvonko G Vranesic, Safwat G Zaky, “Computer Organization”, McGraw- Hill, 5th Edition, 2002.

3. William Stallings, “Computer Organization and Architecture”, Pearson Edition, 8th Edition, 2010.

VI. WEB REFERENCES:

1. http:// www.radio-electronics.com
2. https://accessengineeringlibrary.com
3. https://www.jntubook.com
4. http://www.iare.ac.in

COMPUTER ARCHITECTURE

OE – I: VI Semester: ECE / EEE OE –II: VII Semester: AERO / MECH / CIVIL								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC24	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		

I. COURSE OVERVIEW:

This course introduces the principles of computer organization and the basic architecture concepts. The main objective of this course is to give students to a clear understanding of the modern computer architecture. It also helps the students to know about hardware and software implementation of (ALU) arithmetic and logic unit to solve addition, subtraction, multiplication and division. It also defines the constituent parts of the system, how they are interconnected, and how they interoperate in order to implement the architectural specification. In this course, students will learn the basics of hardware components from basic gates to memory and I/O devices, instruction set architectures and designs to improve the performance.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The organization and architecture of computer systems and electronic computers.
II. The assembly language program execution, instruction format and instruction cycle.
III. How to design a simple computer using hardwired and micro programmed control methods.
IV. The basic components of computer systems besides the computer arithmetic.
V. The input-output organization, memory organization and management, and pipelining.

III. SYLLABUS

MODULE – I: INTRODUCTION TO COMPUTER ORGANIZATION (09)

Basic computer organization, CPU organization, memory subsystem organization and interfacing, input or output subsystem organization and interfacing, simple computer levels of programming languages, assembly language instructions, a simple instruction set architecture.

MODULE –II: ORGANIZATION OF A COMPUTER (09)

Register transfer: Register transfer language, register transfer, bus and memory transfers, arithmetic micro operations, logic micro operations, shift micro operations; Control memory.

MODULE –III: CPU AND COMPUTER ARITHMETIC (09)

CPU design: Instruction cycle, data representation, memory reference instructions, input-output, and interrupt, addressing modes, data transfer and manipulation, program control.

Computer arithmetic: Addition and subtraction, floating point arithmetic operations, decimal arithmetic unit.

MODULE –IV: INPUT-OUTPUT ORGANIZATION (09)

Input or output organization: Input or output Interface, asynchronous data transfer, modes of transfer, priority interrupt, direct memory access.

MODULE –V: MEMORY ORGANIZATION (09)

Memory organization: Memory hierarchy, main memory, auxiliary memory, associative memory, cache memory, virtual memory; Pipeline: Parallel processing, Instruction pipeline.

IV. TEXT BOOKS:

1. M. Morris Mano, “Computer Systems Architecture”, Pearson, 3rd Edition, 2015.
2. Patterson, Hennessy, “Computer Organization and Design: The Hardware/Software Interface”, Morgan Kaufmann, 5th Edition, 2013.

V. REFERENCE BOOKS:

1. John. P. Hayes, “Computer System Architecture”, McGraw-Hill, 3rd Edition, 1998.
2. Carl Hamacher, Zvonko G Vranesic, Safwat G Zaky, “Computer Organization”, McGraw-Hill, 5th Edition, 2002.
3. William Stallings, “Computer Organization and Architecture”, Pearson Edition, 8th Edition, 2010.

VI. WEB REFERENCES:

1. https://www.tutorialspoint.com/computer_logical_organization/
2. https://www.courseera.org/learn/comparch
3. https://www.cssimplified.com/.../computer-organization-and-assembly-language-programming

VI. E-TEXT BOOKS:

1. https://www.groupes.polymtl.ca/inf2610/.../ComputerSystemBook.pdf
2. https://www.cse.hcmut.edu.vn/~vtphuong/KTMT/Slides/TextBookFull.pdf

ADVANCED DATA STRUCTURES

OE – I: VI Semester: ECE / EEE
OE –II: VII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
ACSC25	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

The course is intended to provide the foundations of the practical implementation and usage of Advanced Data Structures. It also covers some classical results and recent advancements on data structures, and the algorithms acting upon them. Typical topics include in sorting and searching, reorganizing lists and search trees based on the online sequence of queries to speed up searches, improving efficiency based on the distribution of queries, performing fast text retrieval by constructing indexes, and improving space efficiency of data structures for large data sets. The main objective of this course is to ensure that the student evolves into a competent programmer capable of designing and analyzing the implementations of different data structures for different kinds of problems.

II.COURSE OBJECTIVES:

The students will try to learn:

I. The basic data structures and techniques of algorithm analysis.
II. The dictionaries, hashing mechanisms and skip lists for faster data retrieval.
III. The comprehension of heaps, priority queues and its operations.
IV. Briefly about balanced trees and their operations.
V. The tries and pattern matching algorithms.

III. SYLLABUS:

MODULE – I: OVERVIEW OF DATA STRUCTURES (09)

Algorithms; Performance analysis: Time complexity and Space complexity, Asymptotic notation. Review of basic data structures - The list ADT, Stack ADT, Queue ADT, Linked list – Single linked list, Double linked list, Circular linked list.

MODULE – II: DICTIONARIES, HASH TABLES (09)

Dictionaries: Linear list representation, Skip list representation, operations - insertion, deletion and searching, Hash table representation, hash functions, collision resolution - separate chaining, open addressing - linear probing, quadratic probing, double hashing, rehashing, extendible hashing, comparison of hashing and skip lists.

MODULE – III: PRIORITY QUEUES (09)

Priority Queues – Definition, ADT, Realizing a Priority Queue using Heaps, Insertion, Deletion,.

Application-Heap Sort, External Sorting- Model for external sorting, Multiway merge, Polyphase merge.

MODULE – IV: SEARCH TREES (09)

Binary Search Trees - Definition, ADT, Operations - Searching, Insertion, Deletion, AVL Trees - Definition, ADT, Balance factor, Operations – Insertion, Deletion, Searching, Introduction to Red – Black and Splay Trees, B-Trees, B-Tree operations - insertion, deletion, searching, Comparison of Search Trees.

MODULE – V: PATTERN MATCHING AND TRIES (09)

Pattern matching algorithms - the Boyer - Moore algorithm, the Knuth – Morris - Pratt algorithm. Tries – Definition, concepts of digital search tree, Binary trie, Patricia, Multi-way trie.

IV. TEXT BOOKS:

1. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, “Fundamentals of Computer Algorithms”, Universities Press Private Limited, India, 2nd Edition, 2008.
2. G.A. V.Pai, “Data Structures and Algorithms”, Tata McGraw Hill, New Delhi, 1st Edition, 2008.
3. Richard F Gilberg, Behrouz A Forouzan, “Data Structures - A Pseudocode Approach with C”, Cengage Learning, Thomson Press (India) Ltd, 2nd Edition, 2006.

V. REFERENCE BOOKS:

1. D. Samanta, “Classic Data Structures”, Prentice Hall of India Private Limited, 2nd Edition, 2003.
2. Aho, Hop craft, Ullman, “Design and Analysis of Computer Algorithms”, Pearson Education India, 1st Edition, 1998.
3. Goodman, Hedetniemi, “Introduction to Design and Analysis of Algorithms”, Tata McGraw Hill, New Delhi, India, 1st Edition, 2002.
4. Adam Drozdek, “Data Structures and Algorithms in C++”, Thomson Course Technology, 3rd Edition, 2005.
5. M. T. Goodrich, R. Tomassia, “Data structures and Algorithms in Java”, Wiley India, 3rd Edition, 2011.

VI. WEB REFERENCE:

1. https://www.tutorialspoint.com/data_structures_algorithms/data_structures_basics.htm
2. https://www.geeksforgeeks.org/data-structures/
3. http://www.nptelvideos.in/2012/11/data-structures-and-algorithms.html

VII. E-TEXT BOOKS:

1. https://pdfs.semanticscholar.org/19ec/55ed703eb24e1d98a4abd1a15387281cc0f8.pdf
2. https://www.academia.edu/35961658/Data.Structures.A.Pseudocode.Approach.with.C.2nd.edition_1_.pdf
3. https://sonucgn.files.wordpress.com/2018/01/data-structures-by-d-samantha.pdf

ARTIFICIAL INTELLIGENCE

OE – I: VI Semester: ECE / EEE **OE –II:** VII Semester: AERO / MECH / CIVIL								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC26	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**			**Total Classes: 45**			

I. COURSE OVERVIEW:

Artificial Intelligence has emerged as an increasingly impactful discipline in science and technology. Al applications are embedded in the infrastructure of many products and industries search engines, medical diagnoses, speech recognition, robot control, web search advertising and even toys. This course provides a broad overview of modern artificial Intelligence, learn how machines can engage in problem solving, reasoning, learning, and interaction design, test and implement algorithms.

II.COURSE OBJECTIVES:

The students will try to learn:

I. Gain a historical perspective of AI and its foundations.

II. Become familiar with basic principles of AI toward problem solving, inference, knowledge representation, and learning.

III. Explore the current scope, potential, limitations, and implications of intelligent systems.

III. SYLLABUS:

MODULE – I: INTRODUCTION (09)

Introduction: AI problems, Intelligent agents: Agents and Environments, the concept of rationality, the nature of environments, Structure of agents, Problem solving agents, Problem formulation.

MODULE – II: KNOWLEDGE REPRESENTATION & REASONS (09)

Knowledge – Based Agents, the Wumpus world.
Propositional Logic: Reasoning patterns in propositional logic - Resolution, Forward & Backward Chaining. Inference in First order logic: Propositional vs. first order inference.

MODULE – III: SEARCHING: (09)

Searching for solutions, uniformed search strategies – Depth limited search, bi-direction search, Comparing uninformed search strategies.

Search with partial information (Heuristic search), TSP problem, best first search, A* search, Hill climbing, Simulated annealing search.

MODULE – IV: CONSTRAIN SATISFACTION PROBLEMS (09)

Backtracking search for CSPs local search for constraint satisfaction problems. Game Playing: Games, Min - Max algorithm, Optimal decisions in multiplayer games, Alpha-Beta pruning.

MODULE – V: PLANNING: (09)

Classical planning problem, Language of planning problem, planning with state – space search, forward state spare search, backward state space search, Heuristics for state space search, Partial order planning Graphs, Planning graphs.

IV. TEXT BOOKS:

1. Stuart Russel, Peter Norvig, “Artificial Intelligence – A Modern Approach”, Pearson Education.

3rd Edition, 2009.

V. REFERENCE BOOKS:

1. E.Rich and K.Knight, “Artificial Intelligence”, Tata McGraw Hill, 3rd Edition, 2008.
2. Patterson, “Artificial Intelligence and Expert Systems”, PHI, 2nd Edition, 2009.
3. Giarrantana/ Riley, “Expert Systems: Principles and Programming”, Thomson, 4th Edition, 2004.
4. Ivan Bratka, “PROLOG Programming for Artificial Intelligence, Pearson Education, 3rd Edition, 2000.

CYBER CRIME AND COMPUTER FORENSICS

OE – I: VI Semester: ECE / EEE
OE –II: VII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AITC19	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

This course is designed to introduce the participant to the cybercrime prevention, detection and incident management processes, policies, procedures and cybercrime governance activities. The course is focus on cybercrime management standards, guidelines and procedures as well as the implementation and governance of these activities. In addition, it also provides the students an understanding of the new and advanced digital investigation techniques for machines, systems and networks since new technologies are opening today the door to new criminal approaches.

II.COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of computer forensics and different types of forensics systems.
II. The methodologies to analyze and validate the forensics data.
III. The different tools and tactics that is associated with cyber forensics.

III. SYLLABUS:

MODULE – I: INTRODUCTION (09)

Introduction: Computer forensics fundamentals, types of computer forensics technology, types of computer forensics systems, vendor and computer forensics services.

MODULE – II: COMPUTER FORENSICS EVIDENCE AND CAPTURE (09)

Data recovery, evidence collection and data seizure, duplication and preservation of digital evidence, computer image verification and authentication.

MODULE – III: COMPUTER FORENSIC ANALYSIS (09)

Discover of electronic evidence, identification of data, reconstructing past events, fighting against macro threats.

Information warfare arsenal, tactics of the military, tactics of terrorist and rogues, tactics of private companies.

MODULE – IV: INFORMATION WARFARE (09)

Arsenal, surveillance tools, hackers and theft of components, contemporary computer crime, identity theft and identity fraud, organized crime &terrorism, avenues prosecution and government efforts, applying the first amendment to computer related crime, the fourth amendment and other legal issues.

MODULE – V: COMPUTER FORENSIC CASES (09)

Developing forensic capabilities, searching and seizing computer related evidence, processing evidence and report preparation, future issues.

IV. TEXT BOOKS:

1. John R. Vacca, “Computer Forensics: Computer Crime Scene Investigation”, Cengage Learning, 2nd Edition, 2005. (UNIT I – IV)
2. Marjie T Britz, “Computer Forensics and Cyber Crime: An Introduction”, Pearson Education, 2nd Edition, 2008. (UNIT IV – V)

V. REFERENCE BOOKS:

1. MariE-Helen Maras, “Computer Forensics: Cybercriminals, Laws, and Evidence”, Jones & Bartlett Learning; 2nd Edition, 2014.
2. Chad Steel, “Windows Forensics”, Wiley, 1st Edition, 2006.
3. Majid Yar, “Cybercrime and Society”, SAGE Publications Ltd, Hardcover, 2nd Edition, 2013.
4. Robert M Slade, “Software Forensics: Collecting Evidence from the Scene of a Digital Crime”, Tata McGraw Hill, Paperback, 1st Edition, 2004.

ETHICAL HACKING

OE – I: VI Semester: ECE / EEE OE –II: VII Semester: AERO / MECH / CIVIL								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AITC20	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

This course will provide fundamentals of the tools and techniques used by hackers and information security professionals alike to break into an organization. This course will immerse you into the Hacker Mindset so that you will be able to defend against future attacks. It puts you in the driver's seat of a hands-on environment with a systematic ethical hacking process. It will give an overview of how to scan, test, hack and secure own systems thought the different phases of ethical hacking include reconnaissance, gaining access, enumeration, maintaining access, and covering various tracks.

II.COURSE OBJECTIVES:

The students will try to learn:

I. The concepts of security testing and the knowledge required to protect against the hacker and attackers.
II. The reconnaissance and the publicly available tools used to gather information on potential targets.
III. The scanning techniques used to identify network systems open ports.
IV. The network system vulnerabilities and confirm their exploitability.
V. The techniques for identifying web application vulnerabilities and attacks.

III. SYLLABUS:

MODULE – I: INTRODUCTION TO HACKING (09)

Introduction to hacking, important terminologies, penetration test, vulnerability assessments versus penetration test, pre-engagement, rules of engagement, penetration testing methodologies, osstmm, nist, owasp, categories of penetration test, types of penetration tests, vulnerability assessment summary, reports.

MODULE – II: INFORMATION GATHERING AND SCANNING (09)

information gathering techniques, active information gathering, passive information gathering, sources of information gathering, tracing the location, traceroute, icmp traceroute, tcp traceroute, usage, udp traceroute, enumerating and fingerprinting the webservers, google hacking, dns enumeration, enumerating snmp, smtp enumeration, target enumeration and port scanning techniques, advanced firewall/ids evading techniques.

MODULE – III: NETWORK ATTACKS (09)

Vulnerability data resources, exploit databases, network sniffing, types of sniffing, promiscuous versus nonpromiscuous mode, mitm attacks, arp attacks, denial of service attacks.

Stripping https, traffic dns spoofing, arp spoofing attack manipulating the dns records, dhcp spoofing, remote exploitation, attacking network remote services, overview of brute force attacks, traditional brute force.

MODULE – IV: EXPLOITATION (09)

Introduction to metasploit, reconnaissance with metasploit, port scanning with metasploit, compromising a windows host with metasploit, client side exploitation methods, e–mails with malicious attachments, creating a custom executable, creating a backdoor with set, pdf hacking, social engineering toolkit, browser exploitation, post, exploitation, acquiring situation awareness, hashing algorithms, windows hashing methods.

MODULE – V: WIRELESS AND WEB HACKING (09)

Wireless hacking, introducing aircrack, cracking the wep, cracking a wpa/wpa2 wireless network using aircrack, ng – evil twin attack, causing denial of service on the original ap, web hacking, attacking the authentication, brute force and dictionary attacks, types of authentication.

IV. TEXT BOOKS:

1. Rafay Baloch, “Ethical Hacking and Penetration Testing Guide”, CRC Press, 2014.

V. REFERENCE BOOKS:

1. Kevin Beaver, “Ethical Hacking for Dummies”, Wiley, 6th Edition, 2018.
2. Jon Erickson , “Hacking: The Art of Exploitation”, Rogunix, 2nd Edition, 2007.

MOBILE COMPUTING

OE – I: VI Semester: ECE / EEE OE –II: VII Semester: AERO / MECH / CIVIL								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AITC21	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURE OVERVIEW:

With the increasing popularity of mobile devices, mobile computing has become part of our daily life. This course will cover the nomenclature and implementation of mobile computing and mobile communication. It also provide a systematic explanation of mobile computing as a discrete discipline and will provide an in-depth coverage of mobile systems and devices used for application development, mobile databases, client-server computing agents, application servers, security protocols, mobile Internet, and ad-hoc and sensor networks.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The concept of wireless transmission protocols.

II. The typical mobile networking infrastructure through a popular GSM protocol architecture.

III. The various layers of mobile networks for location management.

IV. The database issues in mobile environments and data delivery models.

V. The platforms and protocols used in mobile environment.

III. SYLLABUS:

MODULE-I: INTRODUCTION (08)

Mobile computing – Paradigm, promises/Novel applications and impediments and architecture; Mobile and handheld devices, limitations of mobile and handheld devices. GSM – Services, system architecture, radio interfaces, protocols, localization, calling, handover, security, new data services, GPRS.

MODULE-II: MEDIA ACCESS LAYER AND MOBILE NETWORK LAYER (08)

Motivation for a specialized MAC (Hidden and exposed terminals. Near and far terminals), SDMA, FDMA, TDMA, CDMA, wireless LAN (IEEE802.11) system and protocol architecture; Mobile network layer: Packet delivery and handover management, location management, registration, tunneling and encapsulation, route optimization, DHCP.

MODULE-III: MOBILE TRANSPORT LAYER (08)

Conventional TCP/IP protocols, indirect TCP, snooping TCP, mobile TCP, other transport layers protocols for mobile networks;

Database issues: Database hoarding & caching techniques, C-S computing and adaptation, transactional models, query processing, data recovery process and QoS issues.

MODULE-IV: DATA DISSEMINATION AND SYNCHRONIZATION (10)

Communications asymmetry, classification of data delivery mechanisms, data dissemination, broadcast models, selective tuning and indexing methods.

MODULE-V: MOBILE ADHOC NETWORKS(MANET'S) (09)

Introduction, applications and challenges of a MANET, routing, classification of routing algorithms, algorithms such as DSR, AODV, DSDV; Mobile Agents, Service Discovery.

IV. TEXT BOOKS:

1. Jochen Schiller, “Mobile Communications”, Pearson Education, 2nd Edition, 2009.
2. Raj Kamal, “Mobile Computing”, Oxford University Press, Illustrated, 2nd Edition, 2012.

V. REFERENCE BOOKS:

1. Adelstein, Frank, Gupta, Sandeep KS, Richard III, Golden, Schwiebert, Loren, “Fundamentals of Mobile and Pervasive Computing”, McGraw-Hill Professional, 2005.
2. Hansmann, Merk, Nicklous, Stober, “Principles of Mobile Computing”, Springer, 2nd Edition, 2003.
3. Martyn Mallick, “Mobile and Wireless Design Essentials”, Wiley Dream Tech, 1st Edition, 2003.

VI. WEB REFERENCE:

1. https://en.wikipedia.org/wiki/Mobile_computing
2. https://www.tutorialspoint.com/mobile_computing/mobile_computing_quick_guide.h
3. https://media.techtarget.com/searchMobileComputing/downloads/Mobile_and_pervasive_computing_Ch06 pdf

VII. E-TEXT BOOKS:

1. https://books.google.co.in/books?id=HoFdSmH77wsC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&false
2. https://books.google.co.in/books?id=LSqPLwEACAAJ&source=gbs_book_other_versions

EXPERIENTIAL ENGINEERING EDUCATION (ExEEd) – RESEARCH BASED LEARNING

VI Semester: **Common for all branches**								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC27	**Foundation**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		2	-	-	1	30	70	100
Contact Classes: 36	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 36**		
Prerequisite: **There are no prerequisites to take this course**								

I. COURSE OVERVIEW:

Research-based learning (RBL) presents as an alternative learning model that can develop the critical thinking skills. The research-based learning is conducted under constructivism which covers four aspects: learning which constructs student's understanding, learning through developing prior knowledge, learning which involves social interaction process, and meaningful learning which is achieved through real-world experience. The major focus is to engage students in the inquiry process where they formulate questions, conduct investigations, apply information and media to learning, and generate products that illustrate learning. The 5E learning cycle adopted for RBL leads students through five phases: Engage, Explore, Explain, Elaborate, and Evaluate which results in greater benefits concerning student's ability for scientific inquiry.

II. COURSE OBJECTIVES:

The students will try to learn:

I. To provide an opportunity for the students to engage in solving the real-world problems.
II. To introduce the overall process of research from its inception to the report.
III. To create the environment for multi-disciplinary research.
IV. Comprehend the role of ethics in research

III. COURSE SYLLABUS

I. What is Research?
II. Identifying Problem Statement
III. Overview of research-literature
IV. Planning activities, clarifying methods/methodologies
V. Experimentation
VI. Hypothesis testing
VII. Undertaking investigation and analyzing the data
VIII. Interpretation and consideration of results
IX. Presentation of replication studies

ANTENNAS AND MICROWAVE ENGINEERING LABORATORY

VI Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC41	**Core**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	-	4	2	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes: 36**		
Prerequisites: Electromagnetic Waves and Transmission Lines								

I. COURSE OVERVIEW:

The Antennas and Microwave Engineering Laboratory supports intermediate and advanced courses in Electromagnetics and Microwave Engineering. Students experiment with transmission line propagation, antennas and microwave circuit components. The microwave laboratories provide the necessary hardware software support for training the students in the area of RF and Microwave Engineering. It offers design, analysis and simulation of various components and devices to understand the basics of RF and microwave engineering, to boost the quality of engineering education, deepen understanding, and provide the necessary practical skills to young mind.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The experiments on microwave test equipment to make measurements of microwave parameters and devices.

II. The measurement of S-Parameters of microwave components to gain the practical hands on experience on the microwave test bench

III. The simulation to plot the radiation pattern for an antenna using High Frequency Software Simulator.

III. COURSE SYLLABUS:

Week-1: STUDY OF MICROWAVE COMPLONENTS

To study the different wave guide components in the microwave bench setup.

Week-2: MODE CHARACTERISTICS OF REFLEX KLYSTRON

To study the characteristics of Reflex Klystron oscillator, finding the mode numbers and efficiencies of different modes.

Week-3: GUNN DIODE CHARACTERISTICS

To study the characteristics of Gunn diode oscillator.

Week-4: DIRECTIONAL COUPLER CHARACTERISTICS

To measure coupling factor, insertion loss, isolation and directivity of a Directional coupler.

Week-5: MEASUREMENT OF VSWR

To measure the low and high VSWR‘s of matched terminals

Week-6: CIRCULATOR CHARACTERISTICS

To measure the isolation and insertion loss of a three port circulator

Week-7: MEASURMENT OF SCATTERING PARAMETERS OF MAGIC TEE

To find the scattering parameters of a four port Magic Tee.

Week-8: INTRODUCTION TO HFSS

Introduction To HFSS Tool

Week-9: MONOPOLE ANTENNA DESIGN

To find the gain of Monopole Antenna.

Week-10: DIPOLE ANTENNA DESIGN

To draw the Radiation Pattern of Dipole Antenna Design.

Week-11: MICROSTRIP FEED ANTENNA DESIGN

To find the gain and radiation pattern of Microstrip Feed Antenna Design.

Week-12: PROBE FEED PATCH ANTENNA DESIGN
To draw the 3D polar plot of Probe Feed Patch Antenna Design.

Week-13: SLOT COUPLED PATCH ANTENNA
To draw the 3D rectangular plot of Slot Coupled Patch Antenna.

Week-14: MICROSTRIP LINE DESIGN
To find the gain of Microstrip Line Design.

IV. REFERENCE BOOKS:

1. Samuel Y. Liao, “Microwave Devices and Circuits”, Pearson, 3rd Edition, 2003.
2. Herbert J. Reich, J.G. Skalnik, P.F. Ordung and H.L. Krauss, “Microwave Principles” ,CBS Publishers and Distributors, New Delhi, 1st Edition, 2004.
3. F.E. Terman, “Electronic and Radio Engineering”, Tata McGraw-Hill Publications, 4th Edition, 1955.
4. Warren L. Stutzman, Gary A. Thiele, “Antenna Theory and Design”, 3rd Edition, 2012.

V. WEB REFERENCES:

1. http://www.ee.iitkgp.ac.in
2. http://www.citchennai.edu.in

DIGITAL SIGNAL PROCESSING LABORATORY

VI Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
AECC42	**Core**	-	-	4	2	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes: 36**		
Prerequisites: **Signals and Systems**								

I. COURSE OVERVIEW:

This course is concerned with the implementation of digital signal processing algorithms using different computational platforms such as MATLAB and DSP tools that give core knowledge to develop the real time applications in the area of DSP. It focuses on the convolution, discrete Fourier transform, fast Fourier transform algorithms, digital filter design and multi rate signal processing. Digital signal processing applications are used in speech processing, image processing, audio and video data compression, communication systems.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The behavior of discrete time signals and systems in time and frequency domain.
II. The analysis of IIR, FIR digital filters and multi rate signal processing systems.
III. The implementation of real time digital signal processing algorithms using MATLAB tool and TI TMSC67XX target board.

III. COURSE SYLLABUS:

Week-1: LINEAR CONVOLUTION VS CIRCULAR CONVOLUTION

Generation of linear convolution without using built in function and the function conv in MATLAB Generation of circular convolution without using built in function in MATLAB.

Week-2: DFT AND IDFT

Compute the Discrete Fourier Transform and IDFT with and without FFT and IFFT in MATLAB.

Week-3: OVERLAPADD AND OVERLAP-SAVE METHODS

Implementation of Linear convolution using DFT (Overlapadd and Overlap-Save methods).

Week-4: DIT-FFT ALGORITHM

Implementation of Decimation-in-time radix-2 FFT algorithm.

Week-5: DIF-FFT ALGORITHM

Implementation of Decimation-in-frequency radix-2 FFT algorithm.

Week-6: IIR DIGITAL FILTERUSING BUTTERWORTH METHOD AND BILINEAR TRANSFORMATION

Implementation of IIR digital filter using Butterworth method and bilinear transformation.

Week -7: IIR Digital Filter Using Chebyshev (Type I And II) Method

Implementation of IIR digital filter using Chebyshev (Type I and II) method.

Week -8: FIR DIGITAL FILTER USING WINDOWS

Implementation of FIR digital filter using window (Rectangular, Hamming, Hanning, Bartlett) methods.

Week -9: FIR DIGITAL FILTER USING FREQUENCY SAMPLING METHOD

Implementation of FIR digital filter using frequency sampling method.

Week-10: OPTIMUM EQUIRIPPLE FIR DIGITAL FILTER

Implementation of optimum equiripple FIR digital filter using window methods.

Week-11: DTMF TONE GENERATION AND DETECTION

DTMF Tone Generation and Detection Using Goertzel Algorithm.

Week-12: SAMPLING RATE CONVERSION

Implementation of sampling rate conversion by decimation, interpolation and a rational factor using MATLAB.

Week-13: SINE WAVE GENERATION

a) Implementation of DFT b) Sine wave generation using lookup table with values generated from MATLAB..

Week-14: IIR AND FIR FILTERS USING DSP KITS

IIR and FIR Filter Implementation using DSP Kits.

IV. TEXTBOOKS:

1. John G. Proakis, Dimitris G. Manolakis, "Digital Signal Processing, Principles, Algorithms and Applications", Prentice Hall, 4th Edition, 2007.
2. Sanjit K Mitra, "Digital Signal Processing, A Computer Base Approach", McGraw Hill Higher Education, 4th Edition, 2011.
3. Emmanuel C, Ifeacher, Barrie. W. Jervis, "DSP-A Practical Approach", Pearson Education, 2nd Edition, 2002.
4. A.V. Oppenheim, R.W. Schaffer, "Discrete Time Signal Processing", PHI, 2nd Edition, 2006.

V. REFERENCE BOOKS:

1. RobertJ.schilling,Sandra.L.harris, "Fundamentals of Digital Signal Processing using MATlab" , Thomson Engineering, 2nd Edition,2005.
2. Vinay K. Ingle , John G. Proakis, "Digital Signal Processing Using MATlab", Cengage 4th Edition, 2009.
3. DSK Donald Reay, Rulph Chassaing, "Digital Signal Processing and Applications with the TMS 320C6713 and TMS 320C6416", Wiley 2nd Edition.

DESIGN OF ALGORITHMS

VI Semester: **AE / ECE / EEE / ME / CE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACSC29	**Skill**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	-	-	-	-	-	-
Contact Classes: Nil	**Total Tutorials: Nil**	**Total Practical Classes: Nil**				**Total Classes: Nil**		

I. COURSE OVERVIEW:

Design and analysis of algorithms is the process of finding the computational complexity of algorithms. It helps to design and analyze the logic on how the algorithm will work before developing the actual code for a program. It focuses on introduction to algorithm, asymptotic complexity, sorting and searching using divide and conquer, greedy method, dynamic programming, backtracking, branch and bound. NP-hard and NP-complete problems. The applications of algorithm design are used for information storage, retrieval, transportation through networks, and presentation to users.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Assess how the choice of data structures and algorithm design methods impacts the performance of programs.
II. Solve problems using data structures such as binary search trees, and graphs and writing programs for these solutions.
III. Choose the appropriate data structure and algorithm design method for a specified application.
IV. Solve problems using algorithm design methods such as the greedy method, divide and conquer, dynamic programming, backtracking, and branch and bound and writing programs for these solutions.

III. SYLLABUS

MODULE –IINTRODUCTION

Algorithm: Pseudo code for expressing algorithms; Performance analysis: Space complexity, time complexity; Asymptotic notations: Big O notation, omega notation, theta notation and little o notation, amortized complexity; Divide and Conquer: General method, binary search, quick sort, merge sort, Strassen‘s matrix multiplication.

MODULE -IISEARCHING AND TRAVERSAL TECHNIQUES

Disjoint set operations, union and find algorithms; Efficient non recursive binary tree traversal algorithms, spanning trees; Graph traversals: Breadth first search, depth first search, connected components, biconnected components.

MODULE -IIIGREEDY METHOD AND DYNAMIC PROGRAMMING

Greedy method: The general method, job sequencing with deadlines, knapsack problem, minimum cost spanning trees, single source shortest paths.

Dynamic programming: The general method, matrix chain multiplication optimal binary search trees, 0/1 knapsack problem, single source shortest paths, all pairs shortest paths problem, the travelling salesperson problem.

MODULE -IVBACKTRACKING AND BRANCH AND BOUND

Backtracking: The general method, the 8 queens problem, sum of subsets problem, graph coloring, Hamiltonian cycles; Branch and bound: The general method, 0/1 knapsack problem, least cost branch and bound solution, first in first out branch and bound solution, travelling salesperson problem.

MODULE -VNP-HARD AND NP-COMPLETE PROBLEMS

Basic Concepts: Non-deterministic algorithms, the classes NP-Hard and NP-NP Hard problems, clique decision problem, chromatic number decision problem, Cook's theorem.

IV. TEXT BOOKS:

1. Ellis Horowitz, SatrajSahni, SanguthevarRajasekharan, “Fundamentals of Computer Algorithms”, Universities Press, 2nd Edition, 2015.
2. Alfred V. Aho, John E. Hopcroft, Jeffrey D, “The Design And Analysis Of Computer Algorithms”, Pearson India, 1st Edition, 2013.

V. REFERENCE BOOKS:

1. Levitin A, “Introduction to the Design and Analysis of Algorithms”, Pearson Education, 3rd Edition, 2012.
2. Goodrich, M. T. R Tamassia, “Algorithm Design Foundations Analysis and Internet Examples”, John Wileyn and Sons, 1st Edition, 2001.
3. Base Sara Allen Vangelder, “Computer Algorithms Introduction to Design and Analysis”, Pearson, 3rd Edition, 1999.

VI. WEB REFERENCES:

1. http://www.personal.kent.edu/~rmuhamma/Algorithms/algorithm.html
2. http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=IntroToAlgorithms
3. http://www.facweb.iitkgp.ernet.in/~sourav/daa.html

VII. E-TEXT BOOKS:

1. https://kailash392.files.wordpress.com/2019/02/fundamentalsof-computer-algorithms-by-ellis-horowitz.pdf.

EMBEDDED SYSTEM DESIGN

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC43	Core	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Microprocessors and Microcontrollers

I. COURSE OVERVIEW:

This course allows students to learn the fundamentals of embedded system hardware and firmware design. It focus on embedded system design process, embedded C, interfacing modules, software development tools for debugging and testing of embedded applications, ARM & SHARC processor architectures and memory organization. It provides hands-on experience on implementation of embedded application prototype design using embedded C.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of embedded computing, embedded C, RTOS and embedded software tools for implementing embedded systems.

II. Embedded software development tools for debugging and testing of embedded applications, architectures of ARM and SHARC processors.

III. Interfacing with external environments using sensors, actuators and communication in distributed embedded systems.

III. COURSE SYLLABUS:

MODULE –I: EMBEDDED COMPUTING (08)

\Definition of embedded system, embedded systems vs. general computing systems, history of embedded systems, complex systems and microprocessor, classification, major application areas, the embedded system design process, characteristics and quality attributes of embedded systems, formalisms for system design,design examples.

MODULE –II: TYPICAL EMBEDDED SYSTEMS AND ITS APPLICATIONS (09)

Typical Embedded System: Core of the Embedded System, General Purpose and Domain Specific Processors, ASICs, PLDs, Commercial Off-The-Shelf Components (COTS).

Memory: ROM, RAM, Memory according to the type of Interface, Memory Shadowing, Memory selection for Embedded Systems, Communication Interface: Onboard and External Communication Interfaces.

Applications: LED interfacing, LCD display, Seven segment display, DAC and ADC converters interfacing with 8051 Microcontroller.

MODULE –III: RTOS FUNDAMENTALS AND PROGRAMMING (09)

Operating system basics, types of operating systems, tasks and task states, process and threads, multiprocessing and multitasking, How to choose an RTOS ,task scheduling, semaphores and queues, hard real-time scheduling considerations, saving memory and power.

Task communication: Shared memory, message passing, remote procedure call and sockets; Task synchronization: Task communication synchronization issues, task synchronization techniques, Device Drivers.

MODULE –IV: EMBEDDED SOFTWARE DEVELOPMENT TOOLS (09)

Host and target machines, linker/locators for embedded software, getting embedded software into the target system; Debugging techniques: Testing on host machine, using laboratory tools, an example system.

MODULE –V: INTRODUCTION TO ADVANCED PROCESSORS (10)

Introduction to advanced architectures: ARM and SHARC, processor and memory organization and Instruction level parallelism; Networked embedded systems: Bus protocols, I2C bus and CAN bus; Internet-Enabled systems, Design Example-Elevator Controller.

IV. TEXT BOOKS:

1. Shibu K.V, “Introduction to Embedded Systems”, Tata McGraw Hill Education Private Limited, 2nd Edition, 2009.
2. Raj Kamal, “Embedded Systems: Architecture, Programming and Design”, Tata McGraw-Hill Education, 2nd Edition, 2011.
3. Andrew Sloss, Dominic Symes, Wright, “ARM System Developer's Guide Designing and Optimizing System Software”, 1st Edition, 2004.

V. REFERENCE BOOKS:

1. Wayne Wolf, “Computers as Components, Principles of Embedded Computing Systems Design”, Elsevier, 2nd Edition, 2009.
2. Dr. K. V. K. K. Prasad, “Embedded / Real-Time Systems: Concepts, Design & Programming”, Dreamtech publishers, 1st Edition, 2003.
3. Frank Vahid, Tony Givargis, “Embedded System Design”, John Wiley & Sons, 3rd Edition, 2006.
4. Lyla B Das, “Embedded Systems”, Pearson Education, 1st Edition, 2012.
5. David E. Simon, “An Embedded Software Primer”, Addison-Wesley, 1st Edition, 1999.
6. Michael J.Pont, “Embedded C”, Pearson Education, 2nd Edition, 2008.

VI. WEB REFERENCES:

1. http://www.ee.iitkgp.ac.in
2. http://www.citchennai.edu.in

VLSI DESIGN

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC44	Core	L	T	P	C	CIA	SEE	Total
		3	1	-	4	30	70	100
Contact Classes: 45	Tutorial Classes: 15	Practical Classes: Nil				Total Classes: 60		

Prerequisites: Electronic Devices and Circuits

I. COURSE OVERVIEW:

This course introduces the students to fabrication techniques, rapid design and implementation of very large scale (VLSI) circuits. Specific topics include: CMOS logic, MOSFET theory, selection of technology and logic, design process, design rules and layout procedure, design aidfor layout, rule checking, logic and circuit simulation, timing and testability are the main aspects ofthis course. The course further gives information on data path subsystems, several PLD's performance parameters and testing approaches for the circuits.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The aspects of hierarchical VLSI design from the metal oxide semiconductor transistor up to the system level, fabrication and testing.

II. The subsystem design incorporating into a VLSI chip with contemporary techniques for achieving high-speed, low-power and low area overhead.

III. Advanced modern tools such as Vivado and Cadence for front end and back end for chip design through a practical approach.

III. COURSE SYLLABUS:

MODULE – I: BASICS OF MOSFETS (09)

Introduction to IC Technology: MOS, PMOS, NMOS, CMOS & BiCMOS; Fabrication Flow; Basic Electrical Properties of MOS and BiCMOS Circuits: Ids-Vds relationships in saturation and ohmic regions, Weak & strong inversion conditions, Threshold voltage concept in MOSFETs, gm, gds, Figure of merit; Pass transistor; NMOS Inverter; Various pull ups; CMOS Inverter analysis and design; Simple form of Bi-CMOS Inverters and it's alternative forms.

MODULE – II: MOS CIRCUIT DESIGN PROCESSES (09)

VLSI Design Flow; MOS Layers; Stick Diagrams; Physical design rules: 2 µm and lambda CMOS design rules for wires, contacts and transistors; Euler's rule for physical design and Layout; Transistors Layout Diagrams for NMOS and CMOS Inverters; Scaling of MOS circuits; Trends & projections in VLSI design & technology CMOS nanotechnology; FINFET; CNTFET.

MODULE – III: BASIC CIRCUIT CONCEPTS AND GATE LEVEL DESIGN (10)

Sheet Resistance and area capacitance of layers; Inverter Time delays; Driving large capacitive loads; Propagation Delays; Wiring capacitances; Fan-in and Fan-out; Choice of layers . VLSI Interconnects; Reliability issues in CMOS VLSI; Latching in VLSI, Electro migration.

Gate Level Design: Series and Parallel equivalent circuits, Complex gates; Switch logic; Transmission gates; Other forms of CMOS logic such as Pseudo –nMOS, dynamic CMOS, clocked CMOS, CMOS domino, n-p CMOS and their comparisons.

MODULE – IV: SUBSYSTEM DESIGN (09)

Data Path Sub Systems: Sub system design; Shifters, Ripple carry, Carry Look Ahead; Carry select Adders; Manchester carry chain; ALUs; Multipliers; Parity generators; Comparators; Zero/one detectors; Asynchronous and Synchronous; Counters Array Subsystems: SRAM, DRAM, ROM, Floating gate concepts and Flash Memories, Serial access Memories, Content Addressable Memories.

MODULE – V:PROGRAMMABLE LOGIC DEVICES AND CMOS TESTING (08)

Programmable Logic Devices: Design Approach – PROM, PLA and PAL; FPGAs; CPLDs; FPGA building block architectures; FPGA interconnect routing procedures; Speed and area tradeoff. Implementation strategies full custom and semi custom design; CMOS Testing; Built-in Self –Test Strategies; Test pattern generation using LFSR.

IV. TEXTBOOKS:

1. A. Pucknell; Kamran Eshraghian; “BASIC VLSI Design;” , Prentice Hall of India; 3rd Edition, 2007, ISBN: 978-81-203-0986-9.
2. R. Jacob Baker; Harry W.LI.; David E.Boyee; “CMOS Circuit Design; Layout and Simulation”, Wiley-IEEE Press; USA; 2005. ISBN: 978-0-470-88132-3.
3. Jan Rabaey; Anantha Chandrakasan; B.Nikolic; “Digital Integrated Circuits: A Design Perspective;” Second Phi Learning; 2009. ISBN: 9788120322578.

V. REFERENCE BOOKS:

1. N. Weste; K. Eshraghian; “Principles of CMOS VLSI Design”; Addision Wesley; 2nd Edition, 1993, ISBN: 978-81-317-1942-8.
2. M.J. Smith; “Application Specific Integrated Circuits”; Addisson Wesley; 1st Edition; 1997, ISBN-13: 978-0321602756.
3. John P. Uyemura; “CMOS Logic Circuit Design”, Springer; USA; 2007. ISBN: 0-7923-8452-0.

VI. WEB REFERENCES:

1. http://www.ee.iitkgp.ac.in
2. http://www.citchennai.edu.in

ADHOC WIRELESS SENSOR NETWORKS

VII Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC45	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Wireless Communications and Networks								

I. COURSE OVERVIEW:

Ad hoc networks are created between two or more wireless PCs together, without the use of a wireless router or an access point. The basic knowledge of the MAC protocols, routing protocols, transport layer protocols, sensor network networks, and wireless LANs and PANs. It focuses on disaster relief, conference, hospital, campus, and battlefield environments, with laptops, palmtops, cellular phones, or other devices serving as nodes.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The basic concepts of wireless LANs and comparison of wired and wireless LANs.
II. The nature and applications of Ad-hoc and sensor networks.
III. The function of security practices and protocols of Ad-hoc and Sensor Networks.
IV. The performance and characteristics of basic protocols involved in wired/wireless communication process.

III. COURSE SYLLABUS

MODULE-I: WIRELESS LANs AND PANs (09)

Introduction, Fundamentals of WLANS, IEEE 802.11 Standards, HIPERLAN Standard, Bluetooth, Home RF. ADHOC Wireless Networks: Introduction, Issues in Ad Hoc Wireless Networks.

MODULE -II: MAC PROTOCOLS (09)

Introduction, Issues in Designing a MAC protocol for Ad Hoc Wireless Networks, Design goals of a MAC Protocol for Ad Hoc Wireless Networks, Classifications of MAC Protocols, Contention – Based Protocols, Contention - Based Protocols with reservation Mechanisms, Contention – Based MAC Protocols with Scheduling Mechanisms, MAC Protocols that use Directional Antennas, Other MAC Protocols.

MODULE -III: ROUTING PROTOCOLS (09)

Introduction, Issues in Designing a Routing Protocol for ADHOC Wireless Networks, Classification of Routing Protocols, Table –Driven Routing Protocols, On – Demand Routing Protocols,

Hybrid Routing Protocols, Routing Protocols with Efficient Flooding Mechanisms, Hierarchical Routing Protocols, Power – Aware Routing Protocols.

MODULE -IV: TRANSPORT LAYER PROTOCOLS (10)

Introduction, Issues in Designing a Transport Layer Protocol for ADHOC Wireless Networks, Design Goals of a Transport Layer Protocol for ADHOC Wireless Networks, Classification of Transport Layer Solutions, TCPOver ADHOC Wireless Networks, Other Transport Layer Protocol for ADHOC Wireless Networks.

MODULE -V: SENSOR NETWORK NETWORKS (08)

Introduction, Sensor Network Architecture, Data Dissemination, Data Gathering, MAC Protocols for Sensor Networks, Location Discovery, Quality of a Sensor Network, Evolving Standards, Other Issues.

IV. TEXT BOOKS:

1. C. Siva Ram Murthy and B.S.Manoj, “Ad Hoc Wireless Networks: Architectures and Protocols”, PHI, 2004.
2. Jagannathan Sarangapani, “Wireless Ad- Hoc and Sensor Networks: Protocols, Performance and Control”, CRC Press.

V. REFERENCE BOOKS:

1. C.K. Toh,, “Ad- Hoc Mobile Wireless Networks: Protocols & Systems”, Pearson Education, 1st Edition, 2004.

2. C. S. Raghavendra, Krishna M.Sivalingam, “Wireless Sensor Networks”, Springer, 2004.

VI. WEB REFERENCES:

1. https://www.worldscientific.com/worldscibooks/10.1142/8066
2. https://onlinelibrary.wiley.com/doi/book/10.1002/9780470610893
3. http://www.oldcitypublishing.com/journals/ahswn-home/

VII. E-TEXT BOOKS:

1. https://ebooks.benthamscience.com/book/9781608050185
2. https://www.springer.com/la/book/9780387685656
3. https://onlinecourses.nptel.ac.in/noc17_cs07 4. textofvideo.nptel.ac.in/106105160/lec1.pdf 5.
4. https://nptel.ac.in/noc/individual_course.php?id=noc17-cs07 6. https://publons.com/journal/334/ad-hoc-sensor-wireless-networks

ARTIFICIAL NEURAL NETWORKS

VII Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC46	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		

Prerequisites: Probability Theory and Stochastic Processes

I. COURSE OVERVIEW:

This course introduces and relates the basic concept of neural networks. The course will provide a current and coherent view of artificial neural networks. The neural net algorithms will be discussed to understand contemporary neurocomputing technology. It emphasizes mathematical analysis of neural networks, methods for training networks, and application of networks to practical problems. Neural network implementation will be discussed to understand contemporary neurocomputing and soft-computing techniques. Students in computer science, engineering, and behavioral science can obtain immediate benefits to visualize new approaches and interdisciplinary perspectives.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The field of artificial neural networks and machine learning.
II. How to solve practical problems via implementation of these techniques via simulation.
III. The comprehend architecture& algorithms for Adaptive Resonance Theory.
IV. The back propagation for multilayer neural networks.

III.COURSE SYLLABUS

MODULE -I: INTRODUCTION TO NEURAL NETWORKS (10)

Neural Network, Human Brain, Models of Neuron, Neural networks viewed as directed graphs, Biological Neural Network, Artificial neuron, Artificial Neural Network architecture, ANN learning, analysis and applications, Historical notes. structure of biological neurons relevant to ANNs. Models of ANNs; Feedforward & feedback networks; learning rules; Hebbian learning rule, perception learning rule, delta learning rule, Widrow-Hoff learning rule, correction learning rule, Winner –lake all elarning rule, etc.

MODULE -II: SINGLE LAYER PERCEPTION CLASSIFIER (08)

Classification model, Features & Decision regions; training & classification using discrete perceptron, algorithm, single layer continuous perceptron networks for linearly seperable classifications.

MODULE -III: FEED NETWORKS (10)

Multi-layer Feed forward Networks: Linearly non-separable pattern classification, Delta learning rule for multi-perceptron layer, Generalized delta learning rule, Error back-propagation training, learning factors, Examples.

Single layer feedback Networks: Basic Concepts, Hopfield networks, Training & Examples.

MODULE -IV: ASSOCIATIVE MEMORIES & SELF ORGANIZING NETWORKS (09)

Linear Association, Basic Concepts of recurrent Auto associative memory: rentrieval algorithm, storage algorithm; By directional associative memory, Architecture, Association encoding & decoding, Stability.
UN supervised learning of clusters, winner-take-all learning, recall mode, Initializations of weights, seperability limitations.

MODULE -V: NEURO DYNAMICS (08)

Neuro Dynamics: Dynamical Systems, Stability of Equilibrium States, Attractors, Neuro Dynamical Models, Manipulation of Attractors as a Recurrent Network Paradigm, Hopfield Models – Hopfield Models, Computer Experiment.

IV. TEXT BOOKS:

1. M T Hagan, H B Demoth, M Beale, “Neural Networks Design”, Thomson Learning, 2002. ISBN-10: 0- 9717321-1-6/ ISBN-13: 978-0-9717321-1-7.

2. Simon Haykins, “Neural Network- A Comprehensive Foundation”, Pearson Prentice Hall, 2nd Edition, 1999, ISBN-13: 978-0-13-147139-9/ISBN-10: 0-13-147139-2.
3. Simon Haykin,, “Neural Networks a Comprehensive Foundations”, PHI Edition.

V. REFERENCE BOOKS:

1. Simon Haykins, “Neural Network- A Comprehensive Foundation”, Pearson Prentice Hall, 2nd Edition, 1999. ISBN-13: 978-0-13-147139-9/ISBN-10: 0-13-147139-2
2. B.P.Lathi, “Modern Analog and Digital Communication”, Oxford reprint, 3rd Edition, 2004.
3. Zurada and Jacek M, “Introduction to Artificial Neural Systems”, West Publishing Company, 1992, ISBN:9780534954604.
4. Vojislav Kecman,” Learning & Soft Computing”, Pearson Education, 1st Edition, 2004, ISBN:0-262- 11255-8.

VI. WEB REFERENCES:

1. http://www.igniteengineers.com
2. http://www.ocw.nthu.edu.tw
3. http://www.uotechnology.edu.iq

VII. E-TEXT BOOKS:

1. https://www.inf.ed.ac.uk › reading › Gurney_et_al
2. http://www.everythingvtu.wordpress.com

WIRELESS SENSOR NETWORKS

VII Semester: ECE

Course Code	Category	Hours /Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC47	Elective	3	-	-	3	30	70	100
Contact Classes:45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Wireless Communications and Networks								

I. COURSE OVERVIEW:

WSNs are beginning to be organized in an enhanced step. It is not awkward to expect that in 10 to 15 years that the world will be protected with WSNs with entree to them via the Internet. This can be measured as the Internet becoming a physical n/w. This technology is thrilling with infinite potential for many application areas like medical, environmental, transportation, military, entertainment, homeland defense, crisis management and also smart spaces. The most common WSN architecture follows the OSI architecture Model. The architecture of the WSN includes five layers and three cross layers. Mostly in sensor n/w we require five layers, namely application, transport, n/w, data link & physical layer.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. The basic WSN technology and supporting protocols, with emphasis placed on standardization basic sensor systems and provide a survey of sensor technology.
II. The medium access control protocols and address physical layer issues.
III. The different routing protocols for sensor networks and main design issues.
IV. The transport layer protocols for sensor networks, and design requirements.
V. The sensor management, sensor network middleware, operating systems.

III.COURSE SYLLABUS

MODULE-I: OVERVIEW OF WIRELESS SENSOR NETWORKS (10)

Introduction: Components of a wireless sensor node, Motivation for a Network of Wireless Sensor Nodes, Classification of sensor networks, Characteristics of wireless sensor networks, Challenges of wireless sensor networks, Comparison between wireless sensor networks and wireless mesh networks, Limitations in wireless sensor networks, Design challenges, Hardware architecture, Applications : Structural Health Monitoring, Traffic Control, Health Care, .Pipeline Monitoring, Precision Agriculture, Active Volcano, Underground Mining Node Architecture: The Sensing Subsystem, the Processor Subsystem, Communication Interfaces, Prototypes. Operating Systems: Functional Aspects, Nonfunctional Aspects, Prototypes, Evaluation.

MODULE -II: BASIC ARCHITECTURAL FRAMEWORK (09)

Physical Layer, Basic Components, Source Encoding, Channel Encoding, Modulation Medium Access Control: Wireless MAC Protocols, Characteristics of MAC Protocols in Sensor Networks, Contention-Free MAC Protocols, Contention-Based MAC Protocols, Hybrid MAC Protocols.

MODULE -III: NETWORK LAYER (09)

Network Layer: Routing Metrics, Flooding and Gossiping, Data-Centric Routing, Proactive Routing, On-Demand Routing, Hierarchical Routing, Location-Based Routing, QoS-Based Routing Protocols Node.

Network Management: Power Management, Local Power Management aspects, Dynamic Power Management, Conceptual Architecture

MODULE -IV: TIME SYNCHRONIZATION (09)

Time Synchronization: Clocks and the Synchronization Problem, Time Synchronization in Wireless Sensor Networks, Basics of Time Synchronization, Time Synchronization Protocols Localization: Ranging Techniques, Range-Based Localization, Range-Free Localization, Event Driven Localization.

MODULE -V: SECURITY (8)

Fundamentals of Network Security, Challenges of Security in Wireless Sensor Networks , Security Attacks in Sensor

Networks, Protocols and Mechanisms for Security, IEEE 802.15.4 and Zig Bee Security.

IV. TEXTBOOKS:

1. Waltenegus Dargie, Christian Poellabauer, “Fundamentals of Wireless Sensor Networks: Theory and Practice”, Wiley 2010.
2. Mohammad S. Obaidat, Sudip Misra, “Principles of Wireless Sensor Networks”, Cambridge, 2014.

V. REFERENCE BOOKS:

1. Ian F. Akyildiz, Mehmet Can Vuran , “Wireless Sensor Networks”, Wiley 2010.
2. C S Raghavendra, K M Sivalingam, Taieb Znati, “Wireless Sensor Networks”, Springer, 2010.
3. C. Sivarm murthy & B.S. Manoj, “Adhoc Wireless Networks”, PHI-2004.
4. FEI HU., XIAOJUN CAO, “Wireless Sensor Networks”, CRC Press, 2013.
5. Feng ZHAO, Leonidas GUIBAS, “Wireless Sensor Networks”, ELSEVIER , 2004.

VI. WEB REFERENCES:

1. https://www.geeksforgeeks.org/wireless-sensor-network-wsn/
2. https://www.intechopen.com/chapters/38793
3. https://www.elprocus.com/introduction-to-wireless-sensor-networks-types-and-applications/

RF CIRCUIT DESIGN

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC48	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Antennas and Wave Propagation

I. COURSE OVERVIEW:

This course provides in-depth knowledge experience on RF and millimeter circuit design. This course covers the topics on how to derive the RF wireless system specifications and how to design the main building blocks of transceiver i.e. low noise amplifier, power amplifier, RF mixers, oscillators, PLL frequency synthesis. An understanding of the technologies behind transmitter and receiver design including small-signal RF amplifiers is obtained.

II. COURSE OBJECTIVES:

The Students will try to learn:

I. RF systems and circuit design concepts in the existing and future radio technologies.
II. RF circuit devices with active non-linear components and integration of them into systems.
III. Transceiver architecture with matching and biasing networks.

III. COURSE SYLLABUS

MODULE -I: INTRODUCTION (10)

Importance of RF Design-Dimensions and Units-Frequency Spectrum-RF Behaviour of Passive Components: High Frequency Resistors, High Frequency Capacitors, High Frequency Inductors.-Chip Components, and Circuit Board Considerations: Chip Resistors, Chip Capacitors, and Surface Mount Inductors. Review of Transmission Lines: Types of Transmission Lines-Equivalent Circuit representation- R, L, C, G parameters of Different Line configurations-Terminated Lossless Transmission Lines-Special Terminations: Short Circuit, Open Circuit and Quarter Wave Transmission Lines-Sourced and Loaded.
Transmission Lines: Power Considerations, Input Impedance Matching, Return Loss and Insertion Loss.

MODULE -II: SINGLE AND MULTI-PORT NETWORKS (09)

The Smith Chart: Reflection Coefficient, Normalized Impedance-Impedance Transformation: Standing wave Ratio, Special Transformation Conditions-Admittance Transformation-Parallel and Series RL & RC Connections-Basic Definitions of Single and Multi-Port Networks-Interconnecting Networks.

RF Filter Design: Scattering Parameters: Definition, Meaning, Chain Scattering Matrix, Conversion Between S- and Z-parameters, Signal Flow Chart Modeling, Generalization Basic Resonator and Filter Configurations: Low Pass, High Pass, Band Pass and Band Stop type Filters-Filter Implementation using Unit Element and Kuroda's Identities Transformations-Coupled Filters.

MODULE -III: ACTIVE RF COMPONENT MODELLING (08)

RF Diode Models: Nonlinear and Linear Models Transistor Models: Large Signal and Small Signal BJT Models.

Large Signal and Small Signal FET Models- Scattering Parameter, Device Characterization.

MODULE -IV:MATCHING AND BIASING NETWORKS (08)

Impedance Matching Using Discrete Components: Two Component Matching Networks, Forbidden Regions, Frequency Response and Quality Factor, T and Pi Matching Networks-Amplifier Classes of Operation and Biasing Networks: Classes of Operation and Efficiency of Amplifiers, Biasing Networks for BJT, Biasing Networks for FET.

MODULE -V: RF TRANSISTOR AMPLIFIER DESIGN (10)

Characteristics of Amplifiers- Amplifier Power Relations: RF Source, Transducer Power Gain, Additional Power Relations-Stability Considerations: Stability Circles, Unconditional Stability, And Stabilization Methods-Unilateral and Bilateral Design for Constant Gain- Noise Figure Circles- Constant VSWR Circles. RF Oscillators and Mixers: Basic Oscillator Model: Negative Resistance Oscillator, Feedback Oscillator Design, Design steps, Quartz Oscillators- Fixed Frequency High Frequency Oscillator -Basic Characteristics of Mixers: Concepts, Frequency Domain Considerations, Single Ended

Mixer Design, Single, and Double Balanced Mixers.

IV. TEXT BOOKS:

1. Reinhold Ludwig, Pavel Bsetchko, “RF Circuit Design – Theory and Applications”, Pearson Education India, 2000.
2. Devendra K. Misra, “Radio Frequency and Microwave Communication Circuits – Analysis and Design”, Wiley Student Edition – John Wiley & Sons, Inc.

V. REFERENCE BOOKS:

1. Matthew M. Radmanesh, “Radio Frequency and Microwave Electronics”, Illustrated by– PEI.
2. Christopher Bowick, Cheryl Aljuni and John Biyler, “RF Circuit Design”, Elsevier Science, 2008.
3. Joseph J.Carr, “Secrets of RF Circuit Design”, McGraw Hill Education, 2000.
4. Peter L.D. Abrif, “Design of RF and Microwave Amplifiers and Oscillators”, Artech House, 2000.
5. Thomas H.Lee, “The Design of CMOS Radio Frequency Integrated Circuits”, Cambridge University Press, 2nd Edition, 2004.

VI. WEB REFERENCES:

1. http://twanclik.free.fr/electricity/electronic/pdfdone12/Radio%20Frequency%20Circuit%20Design. pdf
2. https://www.highfrequencyelectronics.com/index.php?...rf-circuit-design-references..
3. eecs.oregonstate.edu/~karti/ece621/ece621.pdf

VII. E-TEXT BOOKS:

1. https://ieeexplore.ieee.org/book/5628344
2. https://onlinelibrary.wiley.com/doi/book/10.1002/9781118309940
3. https://www.amazon.in/Radio-Frequency-Integrated-Circuits-Systems/.../0521190797

DIGITAL DESIGN THROUGH VERILOG

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC49	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Digital System Design								

I. COURSE OVERVIEW:

This course introduces the hardware description language for design and development of digital integrated circuits and field programmable devices. Provides hardware description language elements, synthesizable register transfer logic models in gate level, dataflow, behavioral, switch level modeling of combinational and sequential circuits. Allows to use computer aided design tools at the levels of system design, logic design and IC design.

II. COURSE OBJECTIVE:

The students will try to learn:

I. The fundamental principles of the Verilog hardware descriptive language and its constructs used in synthesizable Register Transfer Level (RTL) design implementation of digital logic systems.

II. The concepts of gate level, behavioral, dataflow and switch level modeling of fundamental digital logic circuits using Verilog hardware description Language.

III. The exposure to various stages of a typical ‘state of the art’ CAD VLSI tool for simulation, synthesis, place and route, layout and power and clock routing modules.

IV. The analytical skills needed to model finite state machines using Field Programmable Gate Arrays, fault-tolerant high-speed computer arithmetic circuits, built-in self-circuit (BIST).

III. COURSE SYLLABUS

MODULE - I: INTRODUCTION TO VERILOG HDL (09)

Introduction to Verilog, Popularity of Verilog HDL, Module Concept, Module Modeling Styles, Language Elements: Comments, Identifiers, Keywords, Value Set, Data Types, Memory Element, Constant, Parameter, Operators. Dataflow Modeling: Continuous Assignment, Implicit Continuous Assignment, Delays, Design examples using data flow modeling.

MODULE - II: GATE-LEVEL MODELING (09)

Multiple-Input Gates, Gate Delays, Design Examples, User-Defined Primitives: UDP Basics Combinational User-Defined Primitives, Sequential User-Defined Primitives, Combinational Logic Modules: Decoders, Encoders, Multiplexers, Demultiplexers, Magnitude Comparators.

MODULE - III: BEHAVIORAL MODELING (10)

Procedural Constructs, Procedural Assignments, Timing Control, Conditional Statements, Case Statement Design examples using behavioral modeling.

Loop Statements: For Loop, While Loop, Repeat Loop, Forever Loop, Block Statements Procedural Continuous Assignment, Design examples using behavioral modeling.

MODULE - IV: SWITCH LEVEL MODELLING (09)

Basic Transistor Switches, CMOS Switch, Bi – directional Gates, Time Delays with Switch Primitives, Instantiations with Strengths and Delays, Strength Contention with Trireg Nets.

MODULE - V: SEQUENTIAL LOGIC (08)

Analysis of Synchronous Sequential Machines, Synthesis of Synchronous Sequential Machines, Analysis of Asynchronous Sequential Machines, Synthesis of Asynchronous Sequential Machines, and Synthesis: Design flow of ASICs and FPGA-Based Systems, Design Environment and Constraints, Logic Synthesis.

IV. TEXT BOOKS:

1. Joseph Cavanagh, “Verilog HDL: Digital Design and Modeling”, CRC Press, 1st Edition, 2007.
2. Michael D. Ciletti, “Advanced Digital Design with Verilog HDL”, PHI, 2005.

3. Joseph Cavanagh, “Digital Design and Verilog HDL Fundamentals”, CRC Press, 1st Edition, 2008.

V. REFERENCE BOOKS:

1. Stephen Brown and Zvonko Vranesic, “Fundamentals of Digital Logic design with Verilog Design”, TMH, 2nd Edition, 2010.
2. Sunggu Lee “Advanced Digital Logic Design using Verilog, State Machine & Synthesis for FPGA”, Cengage Learning, 2012.
3. Samir Palnitkar, “Verilog HDL”, Pearson Education, 2nd Edition, 2009.
4. T. R. Padmanabhan and B. Bala Tripura Sundari, “Design through Verilog HDL”, Wiley, 2009.
5. Zainalabdien Navabi, “Verilog Digital System Design”, TMH, 2nd Edition, 2009.

VI. WEB REFERENCES:

1. https://www.crcpress.com/Verilog-HDL-Digital-Design an Modeling/ Cavanag h/p/book/ 9781420051544
2. https://www.uotechnology.edu.iq
3. https://www.iare.ac.in

VII. E-Text Books:

1. https://www.www.jntubook.com
2. https://www.allaboutcircuits.com
3. https://www.archive.org

SCRIPTING LANGUAGES FOR VLSI DESIGN

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC50	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil			Total Classes: 45			

Prerequisites: Python Programming

I. COURSE OVERVIEW:

The course is intended to provide Linux utilities to perform specific tasks such as file handling, text processing. Tool command language (TCL) is a simple textual programming language, intended for issuing commands to interactive programs such as text editors, debuggers and shells. While this course deals largely with the dynamic scripting language such as Java script, PERL and Python we will use these scripting languages are key sources to create electronic design automation (EDA) tool for VLSI design.

II. COURSE OBJECTIVE:

The students will try to learn:

I. The utilities used in Linux environment for file handling, text processing and TCL command for interacting text editors.
II. The scripting language fundamentals, libraries and packages for IC design.
III. Scripting language such as Java script, PERL and Python for interpreting VLSI designs in the EDA tool.

III. COURSE SYLLABUS

MODULE -I: INTRODUCTION TO LINUX (10)

Linux utilities: A brief history of LINUX, architecture and features of LINUX, introduction to vi editor. General purpose utilities, file handling utilities, security by file permissions, process utilities, disk utilities, networking commands; Text processing and backup utilities: Text processing utilities and backup utilities; SED: Scripts, operation, addresses, commands; AWK: Execution, fields and records, scripts, operation, patterns, actions, associative arrays, string and mathematical functions, system commands in awk, applications, Subroutines, Scripts with arguments.

MODULE -II: TCL9 (09)

The TCL phenomena, Philosophy, Structure, Syntax, Parser, Variables and data in TCL, Control flow, Data structures, Simple input/output, Procedures, Working with Strings, Patterns, Files and Pipes, Example code, EDA tool flows, DML-statements.

MODULE -III: ADVANCED TCL9 (08)

The eval, source, exec and up-level commands, Libraries and packages, Namespaces, trapping errors, Event- driven programs.

Making applications 'Internet-aware', 'Nuts-and-bolts' internet programming, Global ISP, Regional ISP Security issues, running un trusted code, The C interface.

MODULE -IV: TK AND JAVA SCRIPTS (08)

Visual tool kits, Fundamental concepts of TK, TK by example, Events and bindings, Geometry managers, Extensive Exercises for Programming in PERL. PERL-TK, Debugger Internal & Externals Portable Functions JavaScript – Object models, Design Philosophy, Versions of JavaScript, The Java Script core language

MODULE -V: INTRODUCTION TO PYTHON (10)

Basic concepts of Python. Object Oriented Programming Concepts (Qualitative Concepts Only): Objects, Classes, Encapsulation, Data Hierarchy. libraries and modules, Files for Scripts.
Logical design using Python: Installing Python, Python data types and data structures, control flow, functions, modules, packages, file handling. Useful of PEP8 for Python.

IV. TEXT BOOKS:

1. Guido Van Rossum, Fred L. Drake Jr., “Python Tutorial” by editor, Release 2.6.4
2. Brent Welch, “Practical Programming in Tcl and Tk”, Updated for Tcl 7.4 and Tk4.0.

V. REFERENCE BOOKS:

1. Brent Welch, “Practical Programming in Tcl and Tk”, 4th Edition, 2003.
2. David Barron, “The World of Scripting Languages”, Wiley Publications, 2000.
3. Guido van Rossum, and Fred L. Drake, “Python Tutorial”, Jr., editor, Release 2.6.4.
4. Neil Mathew, Richard Stones, “Beginning Linux Programming”, Wrox, Wiley India, 4th Edition, 2011.

VI. WEB REFERENCES:

1. https://doc.uments.com/s-vlsi-technology.pdf
2. https://www.quora.com/Why-are-Perl-and-TCL-scripting-languages-used-in-the-VLSI...
3. https://www.jntubook.com
4. https://www.reddit.com/r/Python/comments/37xs5j/python_in_vlsi_scripting

VII. E-TEXT BOOKS:

1. http://vic.gedris.org/Manual-ShellIntro/1.2/ShellIntro.pdf
2. http://www.freeos.com/guides/lsst/ https://technicalpublications.org/.../books/
3. https://python-textbok.readthedocs.io/en/1.0/Object_Oriented_Programming.html
4. https://www.programiz.com/python-programming/

MICROELECTRONICS

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC51	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: VLSI Design

I. COURSE OVERVIEW:

The objective of this course is to develop the ability to analyze and design electronic circuits both analog and digital, discrete and integrated. The course uses incremental and large-signal techniques to analyze and design bipolar and field effect transistor circuits, with examples chosen from digital circuits, single-ended and differential linear amplifiers, micro fluidic technology. Greater emphasis is placed on the fundamentals of photonic circuits and other integrated circuits.

II. COURSE OBJECTIVE:

The students will try to learn:

I. The operational principles, characteristics of bipolar junction transistor and its applications using linear and non-linear analysis.

II. The principles of operating operational amplifier for rectification, amplification, conditioning and voltage regularization of signals.

III. The analytical skills needed to model crystal growth and wafer preparation based on conventional silicon process.

III.COURSE SYLLABUS

MODULE –I:

Bipolar Junction Transistor; Physical Structure and Modes of operation, Operation in Active Mode, circuit symbols and conventions, BJT as an Amplifier, small circuit model, BJT as a switch and Ebers Moll Model, Simple BJT inverter and Second Order Effects, MOS Transistor Basic, MOS Parasitic & SPICE Model; CMOS Inverter Basics, Biasing of MOS Amplifier and its behavior as an analog switch, CMOS CS/CG/SF Amplifier Configuration, Internal cap models and high frequency modeling, JFET, structure and operation.

MODULE –II:

Multistage and Differential Amplifier, Small Signal Operation and Differential Amplifier, MOS Differential Amplier, BiCMOS Amplifier with Active Load, Multistage Amplifier with SPICE Simulation, s-domain analysis, transfer function, poles and zeros, High Frequency Response of CS and CE Amplifier, Frequency Response of CC and SF Configuration, Frequency Response of the Differential Amplifier, Cascode Connection and its Operation,General Feedback structure and properties of negative feedback, Basic Feedback Topologies, Design of Feedback Amplifier for all configuration, Stability and Amplifier poles, Bode Plots and Frequency Compensation.

MODULE –III:

Ideal Operational Amplifier and its terminals, Inverting and Non- Inverting Configuration, As an integrator and Differentiator, Introduction to Analog Computer, Large Signal Operation of Op Amp and Second order offset, Butterworth and Chebyshev Filters, First and Second Order Filter Functions, Switched Capacitor based filters, Single-Amplifier Biquadratic Filters, Second Order LCR Resonator, Combinational Logic Design, Sequential Logic Design.

MODULE –IV:

CRYSTAL GROWTH AND WAFER PREPARATION: Crystal growth techniques: czochralski and gradient freeze techniques, physics involved in CZ growth, Energy flow balance, pull rate considerations, problems and solutions, defects involved in CZ method, DIFFUSION: The nature of diffusion, diffusion mechanisms – interstitial, substitution, interstitial substitution combined, interstitially and grain boundary, Fick’s law of diffusion, limited and constant source diffusion, models of diffusion in solid, diffusion equation, atomic diffusion mechanisms, diffusion system for silicon and gallium arsenide.

MODULE –V:

ION IMPLANTATION: Introduction, physics of implantation, range theory, projected range, ion stopping mechanisms-LITHOGRAPHY: Pattern generation and mask making, exposure sources, photolithography, photoresists, optical lithography, electron lithography, X-ray lithography, ion lithography ,DEPOSITION: Need for film deposition, film deposition methods- physical and chemical, deposition processes, ETCHING: wet chemical etching, wet etchants, dry physical etching, dry etchants, plasma etching, METALLIZATION: Introduction, metallization applications, metallization choices, physical vapour deposition, patterning, metallization problems

IV. TEXT BOOKS:

1. Howe, R. T., and C. G. Sodini, “Microelectronics: An Integrated Approach”, Upper Saddle River, NJ: Prentice Hall, 1996. ISBN: 0135885183.
2. Fonstad, C. G, “Microelectronic Devices and Circuits” New York, NY: McGraw-Hill, 1994. ISBN: 0070214964.
3. Rabaey, Chandrakasan and Nikolic, “Digital Integrated Circuit A Design Perspective”, PHI Latest Edition.
4. Weste and Eshraghian, “Principles of CMOS VLSI Design” Addison Wesley, Latest Edition.

V. REFERENCE BOOKS:

1. Sedra, A. S., and K. C. Smith, “Microelectronic Circuits” , New York, NY: Oxford University Press, 4th Edition, 1998. ISBN: 0195116631.
2. Pierret, R. F, “Semiconductor Device Fundamentals” Upper Saddle River, NJ: Prentice Hall, 1995. ISBN: 0201543931.
3. B.G.Streetman and S. Banerjee, “Solid State Electronic Devices”, Prentice Hall.

VI. WEB REFERENCES:

1. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-012-microelectronic-devices-and-circuits-fall-2005/index.htm
2. https://onlinecourses.nptel.ac.in/noc19_ee54/preview

VII. E-TEXT BOOKS:

1. https://www.wiley.com/en-sg/Microelectronics%2C+2nd+Edition+International+Student+Version-p-9781118165065
2. https://www.sciencedirect.com/book/9780080445533/nanotechnology-for-microelectronics-and-optoelectronics

ADVANCED PROGRAMMABLE LOGIC DEVICE ARCHITECTURES

VII Semester: **ECE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC52	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: **Digital System Design**

I. COURSE OVERVIEW:

An application-specific integrated circuit (ASIC) is an integrated circuit (IC) chip customized for a particular use, rather than intended for general-purpose use. Provides the concepts, structure and programming characteristics of programmable logic devices such as PLDs and FPGAs. Hardware Description Languages (HDLs) are used to create designs that are tested on FPGA devices.

II. COURSE OBJECTIVE:

The students will try to learn:

I. The use of hardware description language in ASIC‘s & FPGA.
II. Various types of ASICs its design how & various programmable logic device.
III. The implementation of digital logic on programmable logic devices.
IV. The architecture specifications and applications of various types of ROMs and RAMs.

III.COURSE SYLLABUS

MODULE-I: INTRODUCTION TO ASICS (09)

ASIC design flow, types of ASICs, full custom ASIC‟s, standard cell based ASIC‟s, Gate array based ASIC‟s, programmable logic devices, introduction to programmable logic, fixed versus programmable logic,programmable logic devices, types of programmable logic devices, PROMs, PLA, PAL, CPLD & FPGA.

MODULE-II: MEMORY AND PROGRAMMABLE LOGIC (09)

Random Access Memory, Programmable Logic, PLD‟S, ROM, Programmable Logic Array, Programmable Array Logic.

MODULE-III: DIGITAL DESIGN WITH SM CHARTS (10)

State Machine charts, Derivation of SM Charts, Realization of SM Charts.

Implementation of Dice Game, Alternative realization for SM charts using microprogramming, Linked StateMachines.

MODULE-IV: DESIGN WITH FIELD PROGRAMMABLE GATE ARRAYS (09)

Field Programmable Gate Arrays – Logic blocks, routing architecture, Design flow. Xilinx 3000 Series, 4000series FPGAs, Designing with FPGAs, Using a One-Hot State Assignment.

MODULE-V: MEMORIES (08)

ROMs: Internal structure, 2D-decoding commercial types, timing and applications. Static RAM: Internal structure, SRAM timing, standard SRAMS, synchronous SRAMS. Dynamic RAM: Internal structure, timing, synchronous DRAMs.

IV.TEXT BOOKS:

1. Stephen. Trimberger , “Field Programmable Gate Array Technology”, Kluwer Academic Publications, 1st Edition, 1994
2. Charles H Roth, Jr. “Digital System Design using VHDL”, Cengage Learning, 2006.
3. John F.Wakerly, “Digital Design Principles & Practices”, PHI/ Pearson Education Asia, 3rd Edition, 2005,

V. REFERENCE BOOKS:

1. Parag.K.Lala, “Digital System Design using Programmable Logic Devices”, BS Publications, 1st Edition, 2003.
2. Stephen Brown and Zvonko Vranesic, “Fundamentals of Digital Logic Design with Verilog Design”, TMH, 2nd Edition, 2010.
3. Charles.H.Roth,Jr., Lizy Kurian John, “Digital System Design using VHD”, Thomson, 2nd Edition, 2008.
4. Zainalabdien Navabi, “Verilog Digital System Design”, TMH, 2nd Edition, 2008.
5. John V.Oldfield, Richard C Dore, “Field Programmable Gate Arrays”, Wiley Publications, 1st Edition, 1995.

VI. WEB REFERENCES:

1. http://www.igniteengineers.com
2. http://www.eecg.toronto.edu
3. http://www.ece.uic.edu
4. http://www.iare.ac.in

VII. E-Text Books:

1. https://books.google.co.in
2. http://www.www.jntubook.com
3. http://www.allaboutcircuits.com
4. http://www.archive.org

SOFT SKILLS AND INTERPERSONAL COMMUNICATION

OE – I: VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT
OE –II: VII Semester: ECE / EEE
OE – III: VIII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC15	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

The objectives of the Soft Skills and Interpersonal Communication are to give each student a realistic perspective of work and work expectations, to help formulate problem solving skills, to guide students in making appropriate and responsible decisions, to create a desire to fulfill individual goals, and to educate students about unproductive thinking, self-defeating emotional impulses, and self- defeating behaviors.

II. COURSE OBJECTIVES:

The students will try to learn:

I. How to communicate in a comprehensible English accent and pronunciation.
II. The four language skills i.e., Listening, Speaking, Reading and Writing effectively.
III. The art of interpersonal communication skills to avail the global opportunities.
IV. The understanding of soft skills resulting in an overall grooming of the skills.

III. SYLLABUS

MODULE-I: SOFT SKILLS (09)

Soft Skills: An Introduction – Definition and Significance of Soft Skills; Process, Importance and Application of Soft Skills, Discovering the Self; Setting Goals; Positivity and Motivation: Developing Positive Thinking and Attitude.

MODULE –II: EFFECTIVENESS OF SOFT SKILLS (09)

Developing interpersonal relationships through effective soft skills; Define Listening, Speaking, Reading and Writng skills; Barriers to Listening, Speaking, Reading and Writing; Essential formal writing skills; Public Speaking: Skills, Methods, Strategies and Essential tips for effective public speaking.

MODULE-III: ORAL AND AURAL SKILLS (09)

Vocabulary:
Sounds of English vowels sounds and constant sounds, Word Accent and connected speech- contractions, questions tags, Listening for information, Taking notes while listening to lectures (use of Dictionary).

Group Discussion: Importance, Planning, Elements, Skills, Effectively disagreeing, Initiating.

MODULE-IV: VERBAL AND NON-VERBAL COMMUNICATION (09)

Interpersonal communication-verbal and nonverbal etiquette; Body language, grapevine, Postures, Gestures, Facial expressions, Proximity; Conversation skills, Critical thinking, Teamwork, Group Discussion, Impact of Stress; Measurement and Management of Stress.

MODULE-V: INTERPERSONAL COMMUNICATION (09)

Significance; Effectiveness of writing; Organizing principles of Paragraphs in documents; Writing introduction and conclusion; Techniques for writing precisely; Letter writing; Formal and Informal letter writing; E-mail writing, Report Writing.

IV. TEXT BOOKS:

Handbook of English for Communication (Prepared by Faculty of English, IARE)

V. REFERENCE BOOKS:

1. Dorch, Patricia. What Are Soft Skills? New York: Execu Dress Publisher, 2013.

2. Kamin, Maxine. Soft Skills Revolution: A Guide for Connecting with Compassion for Trainers, Teams, and Leaders. Washington, DC: Pfeiffer & Company, 2013.
3. Klaus, Peggy, Jane Rohman & Molly Hamaker. “The Hard Truth about Soft Skills”, London: HarperCollins E-books, 2007.
4. Stein, Steven J. & Howard E. Book. “The EQ Edge: Emotional Intelligence and Your Success” Canada: Wiley & Sons, 2006
5. Suresh Kumar. English for Success. Cambridge University Press IndiaPvt.Ltd.2010.
6. Dorling Kindersley. Communication Skills & Soft Skills - An Integrated Approach. India Pvt. Ltd. 2013.

VI. WEB REFERENCES:

1. www.edufind.com
2. www.myenglishpages.com
3. http://grammar.ccc.comment.edu
4. http://owl.english.prudue.edu

VII. E-Text Books:

1. http://bookboon.com/en/communication-ebooks-zip
2. http://www.bloomsbury-international.com/images/ezone/ebook/writing-skills-pdf.pdf
3. https://americanenglish.state.gov/files/ae/resource_files/developing_writing.pdf
4. http://learningenglishvocabularygrammar.com/files/idiomsandphraseswithmeaningsandexamplespdf.pdf
5. http://www.robinwood.com/Democracy/General Essays/CriticalThinking.pdf

CYBER LAW AND ETHICS

OE – I: VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT
OE –II: VII Semester: ECE / EEE
OE –III: VIII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AHSC16	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COUSRE OVERVIEW

This course consists of a sustained study of ethical and legal issues that arise in relation to employment in the public and private sectors, including allocation of resources, corporate and social responsibility, relationships, and discrimination. The main focus of this course will be on the ethical and legal standards governing information technology. New technology creates ethical challenges for individuals around the globe, and applies to most persons regardless of whether they are employed in the information technology field or a more traditional occupation. The study of this course provides a framework for making ethical decisions that professionals are likely to encounter in the workplace. This course will not only focus on ethics but on the legal, economic, social, cultural and global impacts of decisions that are made in the context of professional occupations.

II. COUSRE OBJECTIVES:

The students will try to learn:

I. The key terms and concepts in cyber society, cyber ethics.
II. The fundamentals of Cyber Law
III. The importance of nine P's in ethics.
IV. The artificial intelligence and Blockchain ethics.

III. SYLLABUS

MODULE-I: CYBER SOCIETY (09)

Definitions, Specificities of the Cyberspace, Dimensions of Cyber Ethics in Cyber Society, Fourth Industrial Revolution, Users' Motivations in Cyber-Space, Core Values and Virtues, Old Values or Eschatological Vision?, Cyber Ethics by Norms, Laws and Relations Artificial Intelligence Ethics: "AI for Good", Cyber-Capitalism: Cyber-Ethics as Business Ethics.

MODULE-II: CYBER LAW AND CYBER ETHICS (09)

Cyber Law and Cyber Ethics

The importance of cyber law, the significance of cyber ethics, cyber crime is unethical and illegal, ethics education has positive impact, the need for cyber regulation based on cyber ethics, very dangerous times.

MODULE-III: ETHICS IN THE INFORMATION SOCIETY, THE NINE P'S (09)

Principles: ethical values, participation: access to knowledge for all, people: community, identity, gender, generation, education, profession: ethics of information professions, privacy: dignity, data mining, security.

Piracy: intellectual property, cybercrime, protection: children and young people, power: Economic power of technology, media and consumers, policy: ethics of regulation and freedom.

MODULE-IV: DISRUPTIVE CYBER TECHNOLOGIES AND AI ETHICS (09)

Disruptive Cyber Technologies and Ethics -I

Artificial: negative moral judgment?, artificial: ethically positive innovation?, intelligence: action-oriented ability, creation story: human beings responsibility, the commandment to love and artificial intelligence;

Artificial Intelligence Ethics: Top nine ethical issues in artificial intelligence, five core principles to keep AI ethical, ethics should inform AI, but which ethics?

MODULE-V: DISRUPTIVE CYBER TECHNOLOGIES AND ETHICS –II (09)
Disruptive Cyber Technologies and Ethics -II
BLOCKCHAIN ETHICS:
Blockchain definition and description, Blockchain anonymity and privacy: ethical, no possibility to be forgotten, Blockchain for voting, Blockchain for transparent trade tracing, Blockchain energy: environmental impact, decentralized or majority-owned, ethically more benefits or dangers, future jobs in cyber society.

IV. TEXT BOOKS:

1. Christoph Stuckelberger, Pavan Duggal, "Cyber Ethics 4.0 Serving Humanity with Values", Globethics.net Global Series, 2018.

V. REFERENCE BOOKS:

1. Dr. Farooq Ahmad, Cyber Law in India, Allahbad Law Agency- Faridabad.
2. J.P. Sharma, SunainaKanojia, Cyber Laws
3. Harish Chander , Cyber Laws and IT Protection.

VI. E-REFERENCE:

https://www.globethics.net/documents/4289936/13403236/Ge_Global_17_web_isbn9782889312641.pdf/

ECONOMIC POLICIES IN INDIA

OE – I: VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT
OE –II: VII Semester: ECE / EEE
OE –III: VIII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AHSC17	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW

The objective of this course is to provide a broad sweep of the concept, structure and trends in the Indian economy in a roughly chronological manner. It begins with a review of the evolution of the Indian economy during colonial rule and introduces the roots of Indian underdevelopment. This course is designed to acquaint the students in a comprehensive manner with different aspects of Indian economy. The policy issues and measure to understand economic initiatives for improving economic development and growth, agriculture and industry, planning of the different sectors of the economy and the place of Indian economy in the international level particularly after economic reforms and covered.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The economic development elements and its measures
II. The inside knowledge on monetary policy and its importance in economic development
III. The importance of fiscal policies in promoting the economy
IV. The policies and practices in resource base infrastructure
V. The industrial and exit policies related to the industries

III. SYLLABUS

MODULE-I: INTRODUCTION ECONOMIC DEVELOPMENT AND ITS DETERMINANTS (09)

Approaches to economic development and its measurement – sustainable development; Role of State, market and other institutions; Indicators of development – PQLI, Human Development Index (HDI), gender development indices.

MODULE-II: MONEY, BANKING AND PRICES (09)

Analysis of price behavior in India; Financial sector reforms; Interest rate policy; Review of monetary policy of RBI; Money and capital markets; Working of SEBI in India.

MODULE-III: FISCAL POLICY AND PUBLIC FINANCES (09)

Fiscal federalism – Centre-State financial relations; Finances of central government; Finances of state governments; Parallel economy; Problems relating to fiscal policy; Fiscal sector reforms in India.

MODULE-IV: RESOURCE BASE AND INFRASTRUCTURE (09)

Energy; social infrastructure – education and health; Environment; Regional imbalance; Issues and policies in financing infrastructure development. Policies and Performance in Industry Growth; productivity; diversification; small scale industries; public sector; competition policy; foreign investment.

MODULE-V: THE INDUSTRIAL AND EXIT POLICIES (09)

Industrial policy; Public Sector enterprises and their performance; Problem of sick units in India; Privatization and disinvestment debate; Growth and pattern of industrialization; Small-scale sector; Productivity in industrial sector; Exit policy – issues in labour market reforms; approaches for employment generation.

IV. TEXT BOOKS:

1. The Wealth of Nations-Adam Smith, introduction by Alan B Krueger.
2. The Strength of Economic Development by Albert Hirschman.
3. Money, Banking and Public Finance by Dr. V.C.Sinha
4. Government of India, Economic Survey (Annual), Ministry of Finance, New Delhi.
5. Jain, a. K. (1986), Economic Planning in India, Ashish Publishing House, New Delhi.

V. REFERENCE BOOKS:

1. Ahluwalia, I. J. and I. M. D Little (Eds.) (1999), India's Economic Reforms and Development (Essays in honour of Manmohan Singh), Oxford University Press, New Delhi.
2. Bardhan, P. K. (9th Edition) (1999), The Political Economy of Development in India, Oxford University Press, New Delhi.
3. Bawa, R. s. and P. S. Raikhy (Ed.) (1997), Structural Changes in Indian Economy, Guru Nanak Dev University Press, Amritsar.
4. Brahmananda, P. R. and V. R. Panchmukhi (Eds.) (2001), Development Experience in the Indian Economy: Inter-State Perspectives, Book well, Delhi.
5. Chakravarty, S. (1987), Development Planning: The Indian Experience, Oxford University Press, New Delhi.
6. Dantwala, M. L. (1996), Dilemmas of Growth: The Indian Experience, Sage Publications, New Delhi.
7. Datt, R. (Ed.) (2001), Second Generation Economic Reforms in India, Deep &amp; Deep Publications, New Delhi.

VI. WEB REFERENCE:

1. Parikh, K. S. (1999), India Development Report – 1999-2000, Oxford University Press, New Delhi8.
2. Reserve Bank of India, Report on Currency and Finance, (Annual).
3. Sandesara, J. c. (1992), Industrial Policy and Planning, 1947-19919 : Tendencies, Interpretations and Issues, Sage Publications, New Delhi.

GLOBAL WARMING AND CLIMATE CHANGE

OE – I: **VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT**
OE –II: **VII Semester: ECE / EEE**
OE –III: **VIII Semester: AERO / MECH / CIVIL**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AHSC18	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW

This course aims to address the whole complexity of climate change as an issue, by bringing together the science, impacts, economics, abatement technologies, and policy solutions. The course will address several important questions like what is the scientific basis for our understanding of climate change, and in what ways is that scientific basis uncertain. What changes in climate might we expect over the coming centuries? What would be the impacts of these changes in climate for human well-being and the natural world? What are the sources of emissions of greenhouse gases? What technologies exist or might be developed to allow us to slow climate change, and what international policy solutions might be necessary or preferred?

II. COURSE OBJECTIVES:

The students will try to learn:

I. The importance of Ozone layer in the atmosphere.
II. The comprehend composition of atmosphere.
III. The impacts of climate change on ecosystem.
IV. The initiatives taken by different countries to reduce emission of greenhouse gases.

III. SYLLABUS:

MODULE – I: EARTH'S CLIMATE SYSTEM (09)

Role of ozone in environment, Ozone layer – Ozone depleting gases, Green House Effect – Radioactive effects of Greenhouse gases, The Hydrological cycle, Green House Gases and Global Warming, Carbon Cycle.

MODULE –II: ATMOSPHERE AND ITS COMPONENTS (09)

Importance of Atmosphere – Physical and chemical characteristics of Atmosphere, Vertical structure of the atmosphere, Composition of the atmosphere, Atmospheric stability, Temperature profile of the atmosphere, Lapse rates, Temperature inversion, Effects of inversion on pollution dispersion.

MODULE – III: IMPACTS OF CLIMATE CHANGE (09)

Causes of Climate change: Changes of Temperature in the environment, Melting of ice pole, sea level rise, Impacts of Climate Change on various sectors – Agriculture, Forestry and Ecosystem, Water Resources, Human Health, Industry, Settlement and Society.

Methods and Scenarios, Projected Impacts for different regions, Uncertainties in the projected impacts of Climate Change, Risk of Irreversible Changes.

MODULE – IV: OBSERVED CHANGES AND ITS CAUSES (09)

Climate change and Carbon credits, CDM – Initiatives in India-Kyoto Protocol, Paris Convention - Intergovernmental Panel on Climate change, Climate Sensitivity and Feedbacks. The Montreal Protocol – UNFCCC – IPCC – Global Climate Models (GCM) - Evidences of Changes in Climate and Environment- on a Global scale and in India.

MODULE – V: CLIMATE CHANGE AND MITIGATION MEASURES (09)

Clean Development Mechanism, Carbon Trading – Examples of future clean technology, Biodiesel – Natural Compost, Eco-friendly plastic, Alternate Energy –Hydrogen, Bio-fules, Solar Energy, Wind and Hydroelectric Power. Mitigation Efforts in India and Adaptation funding. Key Mitigation Technologies and Practices –

Energy Supply, Transport, Buildings, Industry, Agriculture, Forestry – Carbon sequestration, Carbon capture and storage (CCS), Waste (MSW & Bio-waste, Biomedical, Industrial waste) – International and Regional cooperation.

IV. TEXT BOOKS:

1. Dr. Sushil Kumar Dash, "Climate Change: An Indian Perspective (Environment and Development)", Cambridge University Press India Pvt Ltd, 2007.
2. Adaptation and mitigation of climate change – Scientific Technical Analysis, Cambridge University Press, Cambridge, 2006.

V. REFERENCE BOOKS:

1. Atmospheric Science, J.M. Wallace and P.V Hobbs, Elsevier/ Academic Press, 2006.
2. "Climate Change and Climate Variability on Hydrological Regimes", Jan C. Van Dam, Cambridge University Press, 2003.

VI. E-TEXT BOOKS

1. https://www.worldcat.org/title/encyclopedia-of-global-warming-climate-change/oclc/805580328
2. https://libguides.nus.edu.sg/c.php?g=433566&p=2955835

INTELLECTUAL PROPERTY RIGHTS

OE – I: VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT **OE –II:** VII Semester: ECE / EEE **OE –III:** VIII Semester: AERO / MECH / CIVIL								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AHSC19	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		

I. COUSRE OVERVIEW:

The course will cover the philosophy of intellectual property rights, various technical and legal dimensions of IPR, and implications of IPR for growth and development of science, along with the various socio-economic and ethico-legal consequences of IPR on economic development. Students can also get disseminate knowledge on Design, Geographical Indication (GI), Plant Variety and Layout Design Protection and their registration aspects and also aware about current trends in IPR and Govt. steps in fostering IPR.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The knowledge in world trade organization and agreements between nations.
II. The intellectual property with international trade agreements.
III. The different types of intellectual property rights.
IV. The different laws in protection of intellectual property rights and its implementation.

III. SYLLABUS:

MODULE- I: INTRODUCTION (10)

General agreement on tariffs and trade (GATT) eight rounds: Uruguay round, world trade organization: structure, technology transfer, dispute resolution mechanism, Doha declaration world trade organization agreements including trade related intellectual properties rights and trade related investment measures.

MODULE- II: WORLD INTELLECTUAL PROPERTY ORGANIZATION (08)

Paris convention, Bern convention, Budapest treaty, Madrid agreement, huge agreement.

MODULE- III: PATENTS (09)

Historical background of intellectual property rights, introduction, definition and classification of intellectual property, patents, patentable and non-patentable inventions. Legal requirements for patents, types of patent applications.

Patent document: specification and claims, important procedural aspects, management of intellectual property rights assets and intellectual property portfolio, commercial exploitation of intellectual property.

MODULE- IV: DESIGNS AND GEOGRAPHICAL INDICATIONS (10)

Designs: basic requirements, procedure, convention application term, date, geographical indication: definition, what can be registered, who can apply, rights, term, restrictions.

MODULE- V: TRADEMARK AND COPYRIGHTS (08)

Definition, classification of trademarks, classifications of goods and services, Vienna classification, trademarks procedure, trademarks enforcement: infringement and passing off, remedies, copyrights, term of copyrights, and procedure of copyright assignment of copyright, copyright infringement remedies.

IV. TEXT BOOKS:

1. P. K. Vasudeva,World Trade Organization: Implications on Indian Economy, Pearson Education,2015.
2. P.KrishnaRao, WTO, Text and cases, Excel Books, 2015.
3. Carlos M.Correa- Intellectual property rights, The WTO and Developing countries-Zed books.

V. REFERENCE BOOKS:

1. Caves, Frankel, Jones, World Trade and Payments-An Introduction, Pearson4. Education, 2015.

2. Carlos M.Correa- Intellectual property rights, The WTO and Developing countries-Zed books.
3. Peter-Tobias stoll, Jan busche, Katrianarend- WTO- Trade –related aspects of IPR- Library of Congress.

VI. WEB REFERENCES:

1. http://www.ebooks directory.com
2. http://Campus guides.lib.utah.edu

VII. E-Text Books:

1. http://www.bookboon.com
2. http://www.freemagagement.com
3. http://www.emeraldinsight.com

ENTREPRENEURSHIP

OE – I: VI Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT
OE –II: VII Semester: ECE / EEE
OE –III: VIII Semester: AERO / MECH / CIVIL

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AHSC20	Elective	3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

This course aims to provide students with an understanding of the nature of enterprise and entrepreneurship and introduces the role of the entrepreneur, will inculcate the knowledge of government supporting programs. Apart from this, students learn about the women entrepreneurs and success stories of women entrepreneurs, gain the knowledge of project management and profitability appraisal, focus on importance of training the new entrepreneurs as well as existing entrepreneurs. The students can also acquire necessary knowledge and skills required for organizing and carrying out entrepreneurial activities, for analysing and understanding business situations in entrepreneurs act and to master the knowledge necessary to plan entrepreneurial activities. The objective of the course is, to develop the ability of analysing various aspects of entrepreneurship – especially of taking over the risk, and the specificities as well as the pattern of entrepreneurship development and, finally, to contribute to their entrepreneurial and managerial potentials.

II. COUSRE OBJECTIVES:

The students will try to learn:

I. The Entrepreneurial process and also inspire them to be Entrepreneurs.
II. The key steps in the elaboration of business idea.
III. The stages of the entrepreneurial process and the resources needed for the successful development of entrepreneurial ventures.

III. SYLLABUS:

MODULE-I: UNDERSTANDING ENTREPRENEURIAL MINDSET (09)

The revolution impact of entrepreneurship- The evolution of entrepreneurship - Functions of Entrepreneurs – types of entrepreneurs -Approaches to entrepreneurship- Process approach- Role of entrepreneurship in economic development- Twenty first century trends in entrepreneurship.

MODULE-II: INDIVIDUAL ENTREPRENEURIAL MIND-SET AND PERSONALITY (09)

The entrepreneurial journey Stress and the entrepreneur - the entrepreneurial ego - Entrepreneurial motivations- Motivational cycle – Entrepreneurial motivational behavior – Entrepreneurial competencies. Corporate Entrepreneurial Mindset, the nature of corporate entrepreneur- conceptualization of corporate entrepreneurship Strategy-sustaining corporate entrepreneurship.

MODULE-III: LAUNCHING ENTREPRENEURIAL VENTURES (09)

Opportunities identification- Finding gaps in the market place – techniques for generating ideas- entrepreneurial Imagination and Creativity- the nature of the creativity process - Innovation and entrepreneurship.

Methods to initiate Ventures- Creating new ventures-Acquiring an Established entrepreneurial venture- Franchising- advantage and disadvantages of Franchising.

MODULE-IV: LEGAL CHALLENGES OF ENTREPRENEURSHIP (09)

Intellectual property protection - Patents, Copyrights - Trademarks and Trade secrets - Avoiding trademark pitfalls. Feasibility Analysis - Industry and competitor analysis - Formulation of the entrepreneurial Plan- The challenges of new venture start-ups, developing an effective business model – Sources of finance - Critical factors for new venture development - The Evaluation process.

MODULE-V: STRATEGIC PERSPECTIVES IN ENTREPRENEURSHIP (09)

Strategic planning - Strategic actions strategic positioning- Business stabilization - Building the adaptive firms - Understanding the growth stage – Internal growth strategies and external growth strategies, Unique managerial concern of growing ventures. Initiatives by the Government of India to promote entrepreneurship, Social and women entrepreneurship.

IV. TEXT BOOKS:

1. D F Kuratko and T V Rao, "Entrepreneurship- A South-Asian Perspective", Cengage Learning, 2012.
2. Bruce R. Barringer/ R.Duane Ireland, "Entrepreneurship Successfully Launching New Ventures", Pearson, 4th Edition, 2015.
3. S.S.Khanka, Entrepreneurship Development, S. Chand Publications, 2015.

V. REFERENCE BOOKS:

1. Stuart Read, Effectual Entrepreneurship, Routledge, 2013.
2. Rajeev Roy, Entrepreneurship, Oxford publications, 2nd Edition, 2012.
3. Nandan .H, Fundamentals of Entrepreneurship, PHI, 2013.

EMBEDDED SYSTEMS DESIGN LABORATORY

VII Semester: **ECE**

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC53	Core	-	-	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes: 36		

Prerequisites: **Microprocessors and Microcontrollers**

I. COURSE OVERVIEW:

This laboratory course is intended to train the students on various embedded modules and embedded C language. This course provides hands-on experience of programming on input/output (I/O) devices and Keil µVision tool. The lab allows students to learn the interfacing of input/output (I/O) devices to increase student interest and develop skills to build embedded systems.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The demonstration of Keil IDE tool and 8051 Microcontroller Development Kit for the implementation of embedded systems.

II. The interfacing of I/O devices with 8051 microcontroller using embedded C language.

III. The interfacing of analog to digital converters (ADC) and digital to analog converters (ADC) with 8051 microcontroller to convert signals from one form to another form.

III. COURSE SYLLABUS:

Week-1: DEVELOP PROGRAM USING KEIL IDE TOOL

Design and develop a reprogrammable embedded computer using 8051 microcontrollers and to show thefollowing aspects.

a. Programming
b. Execution
c. Debugging

To Demonstrate the Tool Chain for Keil IDE (Embedded Systems Development Tool Chain) with the example of LED Blinking Program.

Week-2: INTERFACING LED WITH DIFFERENT PORT PINS

a) Program to toggle all the bits of port P1 continuously with 250 ms delay
b) Program to toggle only the bit P1.5 continuously with some delay

Week-3: INTERFACING BUZZER AND SWITCH

Program to interface a switch and a buzzer to two different pins of a port such that the buzzer should sound as long as the switch is pressed.

Week-4: INTERFACING LCD DISPLAY

Program to interface LCD data pins to port P1 and display a message on it using P89V51RD2

Week-5: INTERFACE HEXA KEYPAD

Program to 4*4 interface keypad. Whenever a key is pressed, it should be displayed on LCD

Week-6: INTERFACE SEVEN SEGMENT DISPLAY

Program to interface seven segment display using 89V51RD2

Week-7: SERIAL COMMUNICATION INTEFACING

Program for serial communication between Microcontroller to PC communication the data should betransfer from microcontroller to PC terminal window using 89V51RD2

Week-8: SERIAL COMMUNICATION INTEFACING

Program for serial communication between PC to Microcontroller communication the data should betransfer from PC to Microcontroller terminal window using 89V51RD2

Week-9: INTERFACING WITH TEMPERATURE SENSOR

Program to develop necessary interfacing circuit to read data from Temperature sensor and process using P89V51RD2, the data has to display terminal window

Week-10: INTERFACING STEPPER MOTOR

Program to interface Stepper Motor to rotate the motor in clockwise and anticlockwise directions

Week-11: INTERFACING MULTPLE DEVICES

Program to verify run 2 to 3 tasks simultaneously on P89V51RD2 SDK. Use LCD interface, LED interface, Serial communication.

Week-12: INTERFACE ADC DEVICE

Program to interface ADC device with P89V51RD2 and display value on LCD

Week-13: INTERFACE DAC DEVICE

Program to interface DAC device with P89V51RD2 and observer the analog output in CRO

Week-14: INTERFACE RELAY

Program to interface Relay with P89V51RD2 using transistor

Week-15: INTERRUPT

Program to toggle LEDS using simple INTERRUPT

IV. TEXT BOOKS

1. Shibu K.V, “Introduction to Embedded Systems”, Tata McGraw Hill Education Private Limited, 2nd Edition, 2009.
2. Raj Kamal, “Embedded Systems: Architecture, Programming and Design”, Tata McGraw-Hill Education, 2nd Edition, 2011.
3. Andrew Sloss, Dominic Symes,Wright, “ARM System Developer's Guide Designing and Optimizing System Software”, 1st Edition, 2004.

V. REFERENCE BOOKS

1. Lyla B Das, “Embedded Systems”, Pearson Education, 1st Edition, 2012.
2. Michael J. Pont, “Embedded C”, Pearson Education, 2nd Edition, 2008.
3. Raj Kamal, “Embedded Systems: Architecture, Programming and Design”, Tata McGraw-Hill Education, 2nd Edition, Tata McGraw Hill, 2011.

VI. WEB REFERENCES:

1. https://www.intorobotics.com/8051-microcontroller
2. https://electrosome.com/led-blinking-8051-microcontroller-keil-c-tutorial-at89c51/
3. http://www.8051projects.net/wiki/Keil_Embedded_C_Tutorial

VLSI DESIGN LABORATORY

VII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC54	Core	L	T	P	C	CIA	SEE	Total
		-	-	3	1.5	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 36				Total Classes: 36		

Prerequisites: Digital System Design

I. COURSE OVERVIEW:

The art of VLSI circuit design is dynamic with advances in process technology and innovations in the electronic design automation (EDA) industry. The objective of this laboratory course is to demonstrate the various stages in VLSI design flow using cadence software. Hands on training on logic and circuit simulations of MOSFETS, ring oscillators, multiplexers, analog amplifiers etc are included. The course also covers physical layout of complex logic gates for chip design.

II. COURSE OBJECTIVES:

The students will try to learn:

I. Modern tools for functional level to physical layout with verification at intermediate stages in the VLSI design flow in top-down approach.

II. Design and simulations of analog, digital and mixed circuits for optimum values of area over head, power and time delay.

III. The Chip design through a practical approach using advanced modern tools such as vivado and cadence for front end and back end.

III. COURSE SYLLABUS:

Week - 1: MOSFET

To plot the (i) output characteristics

(ii) Transfer characteristics of an n-channel and p-channel MOSFET.

Week - 2: CMOS INVERTER

To design the static (VTC) and dynamic characteristics and layout of a digital CMOS inverter.

Week - 3: RING OSCILLATOR

To design and plot the output characteristics of a 3 stage ring oscillator using CMOS inverters.

Week - 4: LOGIC GATES

To design and plot the dynamic characteristics of 2-input NAND, NOR, XOR and XNOR logic gates using CMOS design style

Week - 5: 4X1 MULTIPLEXER

To design and plot the characteristics of a 4x1 digital multiplexer using pass transistor logic and transmission gate logics

Week - 6: LATCHES

To design and plot the characteristics of a positive and negative latch using multiplexers.

Week - 7: REGISTERS

To design and plot the characteristics of a master-slave positive and negative edge triggered registers based on multiplexers.

Week - 8: DIFFERENTIAL AMPLIFIER

Design and simulation of a simple 5 transistor differential amplifier. Measure gain and Common mode rejection ratio.

Week - 9: MOSFET COMMON SOURCE AMPLIFIER

Analysis of Frequency response of Common source amplifiers using n and p MOSFETs

Week - 10: MOSFET COMMON DRAIN AMPLIFIER
Analysis of Frequency response of Common drain amplifiers using n and p MOSFETs

Week - 11: SINGLE STAGE CASCODE AMPLIFIER
Design and Simulation of Single Stage Cascode Amplifier.

Week - 12: BASIC CURRENT MIRROR, CASCODE CURRENT MIRROR AMPLIFIER
Design and Simulation of Basic Current Mirror, Cascode Current Mirror Amplifier.

Week - 13: Design of NAND/NOR using CNTFET
Design and plot the dynamic characteristics of 2-input NAND/ NOR logic gates using CNTFET.

Week - 14: Design of DIFFERENTIAL AMPLIFIER using FinFET
Analysis of Frequency response of differential amplifiers using FinFET.

IV. TEXT BOOKS

1. Razavi, “Design of Analog CMOS Integrated Circuits”, Tata McGraw Hill Publications, 2002.
2. Allen Holberg, “CMOS Analog Circuit Design” Oxford Publications, 2002.
3. Baker, Li, Boyce, “CMOS Mixed Circuit Design”, Wiley Publications, 2002.

V. REFERENCE BOOKS

1. Mohammad Rashid, “Electronic Devices and Circuits”, Cengage learning, 1st Edition, 2014.
2. David A. Bell, “Electronic Devices and Circuits”, Oxford University Press, 5th Edition, 2009.

PROJECT WORK - I

VII Semester: **Common for all branches**								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC55	**Project**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		-	-	4	2	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 150**				**Total Classes: 150**		

The object of Project Work I is to enable the student to take up investigative study in the broad field of Electronics & Communication Engineering, either fully theoretical/practical or involving both theoretical and practical work to be assigned by the Department on an individual basis or two/three students in a group, under the guidance of a Supervisor. This is expected to provide a good initiation for the student(s) in R&D work. Theassignment to normally include:

1. Survey and study of published literature on the assigned topic;
2. Working out a preliminary Approach to the Problem relating to the assigned topic;
3. Conducting preliminary Analysis / Modeling / Simulation/Experiment/Design/Feasibility;
4. Preparing a Written Report on the Study conducted for presentation to the Department;
5. Final Seminar, as oral Presentation before a departmental committee.

DIGITAL IMAGE PROCESSING

VIII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC56	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Signals and Systems

I. COURSE OVERVIEW:

The course is intended to provide image processing fundamentals, representation, sampling, quan- tization, image acquisition and imaging geometry. Transform techniques including two dimensional Fourier transforms, Walsh, Hotelling, Haar and Slant transforms. Analyze image processing filters and techniques for the applications of enhancement, segmentation and compression.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of digital image processing methods and techniques.
II. The image enhancement, image segmentation and compression techniques in spatial and frequency domains.
III. The algorithms to solve image processing problems to meet design specifications of various applications of image processing in industry, medicine and defense.
IV. Fundamentals of image representation and processing in MATLAB.

III. COURSE SYLLABUS:

MODULE –I: INTRODUCTION

Digital image fundamentals and image transforms digital image fundamentals, sampling and quantization, relationship between pixels; Image transforms: 2-D FFT, properties, Walsh transform, Hadamard transform, discrete cosine transform, Haar transform, Slant transform, Hoteling transform.

MODULE –II: IMAGE ENHANCEMENT

Introduction, image enhancement in spatial domain, enhancement through point processing, types of point processing, histogram manipulation, linear and non-linear gray level transformation, local or neighborhood operation, median filter processing; Spatial domain high pass filtering, filtering in frequency domain, obtaining frequency domain filters from spatial filters, generating filters directly in the frequency domain, low pass (smoothing) and high pass (sharpening) filters in frequency domain.

MODULE –III: IMAGE SEGMENTATION AND MORPHOLOGICAL IMAGE PROCESSING

Image segmentation detection of discontinuities, edge linking and boundary detection, threshold, region oriented segmentation, Watershed transformation. Morphological image processing dilation and erosion, structuring element decomposition, the Strel function, erosion; Combining dilation and erosion: Opening and closing the hit and miss transformation, Boundary extraction ,Region filling, Extracted of connected components, convex hull ,skeletons, pruning, Thinning , Thickening.

MODULE –IV: IMAGE RESTORATION

Image restoration degradation model, Noise models, Restoration in the presence of noise only (Spatial Filtering), Estimating the degradation function, Inverse filtering, Least mean square filters, constrained least square restoration.

MODULE –V: IMAGE COMPRESSION AND WAVELET BASED IMAGE PROCESSING

Image compression: Redundancies and their removal methods, fidelity criteria, image compression models, source encoder and decoder, error free compression, lossy compression, Wavelet transform: Continuous wavelet transformation, 2D continuous wavelet transformation, Examples of wavelets, Wavelet based image compression.

V. TEXT BOOKS:

1. Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing”, Pearson, 3rd Edition, 2008.

2. S. Jayaraman, S. Esakkirajan, T. Veerakumar, “Digital Image Processing”, TMH, 3rd Edition, 2010.

VI. REFERENCE BOOKS:

1. Rafael, C. Gonzalez, Richard E woods, Stens L Eddings, “Digital Image Processing using MAT LAB”, Tata McGraw Hill, 2nd Edition, 2010.
2. A.K. Jain, “Fundamentals of Digital Image Processing”, PHI, 1st Edition, 1989.
3. Somka, Hlavac, Boyle, “Digital Image Processing and Computer Vision”, Cengage Learning, 1st Edition, 2008.
4. Adrain Low, “Introductory Computer vision Imaging Techniques and Solutions”, Tata McGraw-Hill, 2nd Edition, 2008.
5. John C. Russ, J. Christian Russ, “Introduction to Image Processing & Analysis”, CRC Press, 1st Edition, 2010.

VII. WEB REFERENCES:

1. https://imagingbook.com/
2. https://en.wikipedia.org/wiki/Digital_image_processing
3. http://www.tutorialspoint.com/dip/
4. http://www.imageprocessingplace.com

VIII. E-TEXT BOOKS:

1. http://bookboon.com/en/communication-ebooks-zip
2. http://www.bloomsbury-international.com/images/ezone/ebook/writing-skills-pdf.pdf
3. https://americanenglish.state.gov/files/ae/resource_files/developing_writing.pdf

SIGNAL PROCESSING FOR COMMUNICATION AND BIOMEDICAL APPLICATIONS

VIII Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC57	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Signals and Systems								

I. COURSE OVERVIEW:

In communications engineering and signal analysis methodology, the comprehensive treatment of advanced communication systems from theoretical and practical approaches. For original knowledge dissemination and communication of tangible research outcomes in the field of Biomedical Signal Processing. Signal processing theories for biomedical applications based on time-domain approaches, spectral domain approaches, joint time and frequency methods, signal decomposition methods, sparse signal representations. Signal modalities based on ECG, EEG, EMG, PPG, voice, bioacoustics, and other biomedical signals of electrical, mechanical or chemical in nature. The role of software systems, hardware-software co-design and user-centered design factors for biomedical signal processing.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The basic signal processing techniques in analyzing biological signals.

II. The mathematical, scientific & computational skills related to the field of biomedical signal processing

III. The awareness of the complexity of biological signal and the impact, promise of biomedical engineering in understanding these signals.

III. COURSE SYLLABUS:

MODULE – I: INTRODUCTION TO BIO SIGNALS (10)

Introduction to bio signals- generation of bio signals-action potential, resting membrane potential, ECG-data acquisition, lead system, arrhythmias; EEG signals and characteristics, evoked potential ,Computerized data acquisition system - basic requirements, Preprocessing of bio-signals removal of interferences due to power line & Electro Surgical Unit, Adaptive filtering fetal heart rate monitoring -a case study

MODULE – II: STATISTICAL SIGNAL PROCESSING (09)

Statistical Signal Processing - Introduction to random signals and its characteristics, properties of random signals, moments of signal, Concepts of PDF, autocorrelation, cross correlation, covariance, estimation of power spectral density-parametric& non-parametric, Wiener filter, implementation of algorithm for autocorrelation and PSD using MATLAB.

MODULE – III: ANALYSIS OF BIO SIGNALS (10)

Analysis of bio signals – ECG- continuous monitoring, arrhythmia detection- algorithms and methods, HRV signal. EEG- video EEG, analysis of epilepsy using EEG, prediction.

Modeling of bio signals –linear models- AR model, MA model and ARMA model, Modeling of ECG signal using MATLAB.

MODULE – IV: NON STATIONARY SIGNAL ANALYSIS (08)

Non stationary signal analysis: Time domain methods, frequency domain methods, time frequency methods, wavelets Classification - introduction to ANN and Fuzzy logic, algorithm for Arrhythmia classification.

MODULE – V: INTRODUCTION TO NONLINEAR ANALYSIS (08)

Introduction to nonlinear analysis of bio signals-chaos, Analysis of heart rate variability and blood pressure variability using measures based on chaotic theory.

IV. TEXT BOOKS:

1. Peyton Z. Peebles, “Probability, Random Variables & Random Signal Principles”, TMH, 4th Edition, 2009.
2. D. C. Reddy, “Biomedical Signal Processing- Principles and Techniques”, TMH, 2005.
3. Weitkunat R, “Digital Bio Signal Processing”, Elsevier, 1991.

V. REFERENCE BOOKS:

1. Varun Bajaj, G. R. Sinha, Chinmay Chakraborty, “Biomedical Signal Processing for Healthcare Applications”, CRC Press, 2021.
2. Cohen.A,, “Biomedical Signal Processing”, Vol. I Time & Frequency Analysis, CRC Press, 1986.

VI. WEB REFERENCES:

1. http://www.biomedicahelp.altervista.org/SecondoAnno/StatisticaSegnali/Segnali/Segnali_BiomedicalSignalAnalysisBook_Libro.pdf
2. https://www.ncl.ac.uk/postgraduate/degrees/module/?code=EEE8128
3. https://www.ncl.ac.uk/postgraduate/degrees/module/?code=EEE8129
4. https://nptel.ac.in/courses/108/105/108105101/
5. https://www.ncl.ac.uk/postgraduate/degrees/5066f/

WAVELETS AND APPLICATIONS

VIII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC58	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Signals and Systems								

I. COURSE OVERVIEW:

The course discusses about the fundamentals of wavelet theory and its applications. The balance between mathematical rigour and practical applications of wavelet theory are applied using the remarks and graphical representations to explain mathematical ideas in a conversational way. The course emphasizes on the continuous and discrete wavelet transform techniques, the filter bank and multi resolution analysis using signal processing concepts and applications of wavelet analysis in various aspects.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The mathematical introduction to the wavelet theory: Continuous and discrete wavelet transforms, wavelet base and wavelet packages, wavelets and singular integrals.

II. Explain the concepts, theory, and algorithms behind wavelets from an interdisciplinary perspective that unifies harmonic analysis (mathematics), filter banks (signal processing), and multi resolution analysis (computer vision).

III. Application of wavelets, filters banks, and multi resolution techniques to a problem at hand, and justify why wavelets provide the right tool.

III. COURSE SYLLABUS:

MODULE-I : INTRODUCTION

Stationary and non-stationary signals, Signal representation using basis and frames, Brief introduction to Fourier transform and Short time Fourier transform, Time frequency analysis, Bases of time frequency: orthogonal, Filter banks, Multi resolution formulation: Wavelets from filters, Classes of wavelets: Haar, Daubechies, bi-orthogonal.

MODULE-II: CONTINUOUS WAVELET TRANSFORM

Continuous wavelet transform (CWT), Time and frequency resolution of the continuous wavelet transform, Construction of continuous wavelets: Spline, orthonormal, bi-orthonormal, Inverse continuous wavelet transform, Redundancy of CWT, Zoom property of the continuous wavelet transform, Filtering in continuous wavelet transform domain.

MODULE-III: DISCRETE WAVELET TRANSFORM AND FILTER BANKS

Orthogonal and biorthogonal two-channel filter banks, Design of two-channel filter banks, Tree-structured filter banks, Discrete wavelet transform.

Non-linear approximation in the Wavelet domain, multi resolution analysis, Construction and Computation of the discrete wavelet transform, the redundant discrete wavelet transform.

MODULE-IV: MULTI RESOLUTION ANALYSIS

Multirate discrete time systems, Parameterization of discrete wavelets, Bi-orthogonal wavelet bases, Two dimensional, wavelet transforms and Extensions to higher dimensions, wave packets.

MODULE-V: APPLICATIONS

Signal and Image compression, Detection of signal changes, analysis and classification of audio signals using CWT, Wavelet based signal de-noising and energy compaction, Wavelets in adaptive filtering, Adaptive wavelet techniques in signal acquisition, coding and lossy transmission, Digital Communication and Multicarrier Modulation, Trans multiplexers, Image fusion, Edge Detection and object isolation.

IV. TEXT BOOKS:

1. S. Mallat, “A Wavelet Tour of Signal Processing”, Academic Press, 2nd Edition, 1999.
2. M. Vetterli and J. Kovacevic, “Wavelets and Sub band Coding”, Prentice Hall, 1995.
3. Raghuveer rao and Ajit S.Bopardikar, “Wavelet transforms: Introduction, Theory and applications”, Pearson Education Asia, 2000.

V. REFERENCE BOOKS:

1. J.C. Goswami and A.K. Chan, “Fundamentals of Wavelets: Theory, Algorithms, and Applications”, Wiley 2nd Edition, , 2011.
2. Michel Misiti, Yves Misiti, Georges Oppenheim, JeanMichel Poggi, “Wavelets and their Applications”, John Wiley & Sons, 2010.
3. J S Walker, “A Premier on Wavelets and their scientific applications”, CRC press, 2002.
4. Stark, “Wavelets and Signal Processing: An application based introduction”, Springer, 2005.

VI. WEB REFERENCES:

1. https://ocw.mit.edu/courses/mathematics/18-327-wavelets-filter-banks-and-applications-spring-2003/
2. http://math.sfsu.edu/shidong/wvlet_2020s.html
3. http://www.ifp.illinois.edu/~minhdo/teaching/wavelets.html
4. https://www.geneseo.edu/~haddad/WaveletsCourseDescriptionF09.html

VII. E-TEXT BOOKS:

1. https://waveletsandsubbandcoding.org/Repository/VetterliKovacevic95_Manuscript.pdf
2. https://www.wiley.com/en-gb/Fundamentals+of+Wavelets:+Theory,+Algorithms,+and+Applications,+2nd+Edition-p-9780470934647

DIGITAL SIGNAL PROCESSORS AND ARCHITECTURE

VIII Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC59	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Digital Signal Processing								

I. COURSE OVERVIEW:

This course enables the architecture, memory management of single instruction and multiple data, very large instruction word and TMS processors for the implementation of discrete fourier transform and fast fourier transform algorithms. It focuses on memory organization, external bus interfacing signals, parallel input/output interface, interrupts, direct memory access, finite impulse response, infinite impulse response filters. In built peripherals of TMS processor used in applications of communication equipment, image processing, control systems and consumer electronic devices.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The architectures of digital signal processors and design aspects of digital signal processing algorithms.

II. The memory and external input/output peripheral interface with programmable DSP processor.

III. The realization of digital filters and fast fourier transform algorithms of the signal spectrum on host DSP processor.

IV. The programming skills using code composer studio environment for TMS320C54XX processor.

III. COURSE SYLLABUS:

MODULE - I: INTRODUCTION TO DIGITAL SIGNAL PROCESSOR

Digital signal-processing and processors, Sampling process, Discrete time sequences, Discrete Fourier Transform (DFT) and fast Fourier transform (FFT), differences between DSP and other micro processor architectures; Computational accuracy in DSP implementation-Number formats: Fixed point, floating point and block floating point formats, IEEE-754 floating point, dynamic range and precision, relation between data word size and instruction word size; Sources of error in DSP implementations: A/D conversion errors, DSP computational errors, D/A conversion errors.

MODULE - II: ARCHITECTURE OF PROGRAMMABLE DSP DEVICES

DSP Computational building blocks-Multiplier and multiplier accumulator, modified bus structures and memory access in PDSPs, multiple access memory, multiport memory, SIMD, VLIW architectures, pipelining, special addressing modes in PDSPs, on-chip peripherals.

MODULE - III: OVERVIEW OF TMS320C54XX PROCESSOR

Architecture of TMS320C54XX DSPs, addressing modes, memory space of TMS320C54XX processors.

Program control, instruction set and programming, on-chip peripherals, interrupts of TMS320C54XX processors, pipeline operation.

MODULE - IV: INTERFACING MEMORY AND I/O PERIPHERALS TO PDSPs

Memory space organization, external bus interfacing signals, memory interface, parallel I/O interface, programmed I/O, interrupts and I/O, direct memory access (DMA).A Multi channel buffered serial port (McBSP) and CODEC - DSP Interface.

MODULE - V: IMPLEMENTATIONS OF BASIC DSP ALGORITHMS

The Q-notation, convolution, correlation, FIR filters, IIR filters, interpolation filters, decimation filters, an FFT algorithm for DFT filters computation of the signal spectrum. PID controller, Adaptive filters and 2-d signal processing.

IV. TEXT BOOKS:

1. Avatar Singh and S. Srinivasan, “Digital Signal Processing”, Thomson Publications, 1st Edition, 2004.
2. Lapsley et al., “DSP Processor Fundamentals Architectures & Features”, S. Chand & Co, 1st Edition, 2000.
3. B. Ventakaramani, M. Bhaskar, “Digital Signal Processors Architecture Programming and Applications”, Tata McGraw-Hill, 1st Edition, 2006.

V. REFERENCE BOOKS:

1. Jonatham Stein, “Digital Signal Processing”, John Wiley, 1st Edition, 2000.
2. Sen M. Kuo&WoonSergGan, “Digital Signal Processors Architectures, Implementation and Application”, Pearson Practice Hall, 1st Edition, 2013.
3. K Padmanabhan, R.Vijayarajeswaran, Ananthi. S, “A Practical Approach to Digital Signal Processing”, New Age International, 1st Edition, 2006.
4. Ifeachor E. C., Jervis B. W, “Digital Signal Processing: A practical approach”, Pearson Education, PHI/, 2nd Edition, 2002.
5. Peter Pirsch, “Architectures for Digital Signal Processing”, John Weily, 1st Edition, 2007.

VI. WEB REFERENCES:

1. https://books.google.co.in/books/about/DigitalSignalProcessors.html?id=2A2-v3raKEC
2. https://www.analog.com/en/design-center/landing-pages/001/beginners-guide-to-dsp.html
3. https://nptel.ac.in/noc/courses/noc19/SEM2/noc19-ee70/
4. https://onlinecourses.nptel.ac.in/noc21_ee20/preview

VII. E-TEXT BOOKS:

1. https://books.google.com.na/books?id=2A_2-v3raKEC&printsec=copyright#v=onepage&q&f=false
2. https://1lib.in/book/5472545/a3fcb6

MICROCONTROLLERS AND APPLICATIONS

VIII Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC60	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Microprocessors and Microcontrollers								

I. COURSE OVERVIEW:

Controller cores are the key components in most of the modern embedded and system on-chip designs. This course outlines the architecture and signal description of 8 bit and 16 bit microcontrollers. It also covers the I/O devices interfacing with microcontrollers for real time applications. This course will enable the students in development of embedded hardware projects and models for engineering and scientific applications.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The signal descriptions along with functional architecture and hardware interfacing skills using 8 bit and 16 bit microcontrollers.
II. The programming and interfacing of microcontrollers with I/O devices.
III. The essential concepts of development through a practical hands-on approach on advanced ARM processors and Internet of Things based systems.

III. COURSE SYLLABUS:

MODULE - I : OVERVIEW OF ARCHITECTURE AND MICROCONTROLLER RESOURCES

Architecture of a microcontroller, Microcontroller resources, Resources in advanced and next generation microcontrollers, 8051 microcontroller –Internal and External memories, Counters and Timers, Synchronous and asynchronous serial communication, Interrupts, Instruction set of 8051 microcontroller, Basic assembly language programming

MODULE - II:REAL TIME CONTROL

Interrupts: Interrupt handling structure of an MCU, Interrupt Latency and Interrupt deadline, Multiple sources of the interrupts, Non-maskable interrupt sources, Enabling or disabling of the sources, Polling to determine the interrupt source and assignment of the priorities among them, Interrupt structure in Intel 8051.
TIMERS : Programmable Timers in the MCU's, Free running counter and real time control, Interrupt interval and density constraints.

MODULE - III : SYSTEMS DESIGN

Digital And Analog Interfacing Methods: Switch, Keypad and Keyboard interfacings, LED and Array of LEDs, Keyboard/ Display controller (8279), Alphanumeric Devices, Display Systems and its interfaces, Printer interfaces, Programmable instruments interface using IEEE 488 Bus, Interfacing with the Flash Memory.

Interfaces – Interfacing to High Power Devices, Analog input interfacing, Analog output interfacing, Optical motor shaft encoders, Industrial control, Industrial process control system, Prototype MCU based Measuring instruments, Robotics and Embedded control, Digital Signal Processing and Digital Filters.

MODULE - IV: REAL TIME OPERATING SYSTEM FOR MICROCONTROLLERS

Real Time operating system, RTOS of Keil (RTX51), Use of RTOS in Design, Software development tools for Microcontrollers.

MODULE - V: 16-BIT MICROCONTROLLERS

Hardware – Memory map in Intel 80196 family MCU system, IO ports, Programmable Timers and High-speed outputs and input captures, Interrupts – instructions.
ARM 32 Bit MCUs : Introduction to 16/32 Bit processors, ARM architecture and organization, ARM / Thumb

programming model, ARM / Thumb instruction set, Development tools.

IV. TEXT BOOKS:

1. Raj Kamal, “Microcontrollers Architecture, Programming, Interfacing and System Design”, Pearson Education, 2005.
2. Mazidi and Mazidi, “The 8051 Microcontroller and Embedded Systems , PHI, 2000.

V. REFERENCE BOOKS:

1. A.V. Deshmuk, “Microcontrollers (Theory & Applications)”, WTMH, 2005.
2. John B. Peatman, “Design with PIC Microcontrollers”, Pearson Education, 2005.

VI. WEB REFERENCES:

1. https://www.vectorindia.org › 8051_microcontroller
2. https://www.digikey.in

EMBEDDING SENSORS AND MOTORS

VIII Semester: ECE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AECC61	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	-	-	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		
Prerequisites: **Electronic Measurements and Instrumentation**								

I. COURSE OVERVIEW:

This course will introduce to design of sensors and motors, and to methods that integrate them into embedded systems used in consumer and industrial products. An electronic or computer system that is designed to control, access the data in electronics based systems. The applications include industrial control and monitoring, IoT industries, artificial intelligence and logistics monitoring.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The embedding sensors and motors are to monitor processes and assets accurately and reliably to increase the performance and productivity.

II. The IOT based sensors are used for detection of changes in the physical and/or logical relationship of one object to another(s) and/or the environment.

III. To gain the hands-on experience with the technologies by building systems that take sensor or motor inputs, and then filter and evaluate the resulting data.

III. COURSE SYLLABUS:

MODULE - I: SENSORS AND SENSOR CIRCUIT DESIGN (08)

Thermal Sensors-Temperature sensors in embedded circuit, specifications and uses, types of sensors and actuators in smartphones and automobiles, specifications, high-level overview of analog and digital interfaces, thermistors, RTD’s, thermocouples, types of thermal sensors, Examples of commercial sensors.

MODULE - II: SENSOR CIRCUIT DESIGN (09)

Sensor development-design of complete temperature sensor system, internal and external components, interfacing a thermistor, Rotary and flow sensors-rotary sensors, optical encoders and resolvers, design intricacies of flow sensors and applications, Amplifiers and sensor noise-amplifiers and circuit noise, gain of inverting, non-inverting, summing, differential, and instrumentation amplifiers, noise in sensor circuits and its effects.

MODULE - III: MOTORS AND MOTOR CONTROL CIRCUITS (10)

AC motor design-operation of AC induction motors, single and 3-phase types, torque speed curves, types of single phase motors, split phase motor, applications for single phase motors, AC motor control-specifications and enclosures, design standards, AC motor control components and systems.

DC motors-principles of DC motors, traditional brushed motors, electronically driven brushless motors, shunt wound, series wound, compound wound, servo, stepper, and torque motors, DC motor speed measurement, DC motor control and stepper motors.

MODULE - IV: PRESSURE, FORCE, MOTION AND HUMIDITY SENSORS (09)

Pressure sensors-types of pressure sensors for an embedded circuit, piezoresistive, capacitive, and vacuum sensors, Wheatstone bridge, pressure transmitters, types of force and strain sensors, working of touch screens, magnetic detection sensors, capacitive detection sensors, use accelerometers in an embedded circuit, MEMS technology, position and motion detectors in an embedded circuit, pyroelectric effect, Passive Infrared motion detectors, ultrasonic distance detection, microwave detection sensors.

MODULE - V: SENSOR MANUFACTURING AND PROCESS CONTROL (09)

Process control, Implementation of plant-wide control systems, software protocols, PID control with PSOC system, sensor characterization, sensor calibration capability, sensor accuracy, closed loop motor control with PSOC system, Advanced sensors- radar level transmitters, components and design issues for LIDAR systems for self-driving vehicles, sensors in medical applications, Sensor manufacturing- MEMS construction, micro-electronic connections, designs of housings , Sensor testing.

IV. TEXT BOOKS:

1. “Sensors and Transducers”, Prentice Hall India Learning Private Limited; 2nd Edition, 2003.

V. REFERENCE BOOKS:

1. Dharma, Prakash Agrawal, “Embedded Sensor Systems”, Springer Singapore, 2017.

VI. WEB REFERENCES:

1. https://www.engineersgarage.com/sensors-different-types-of-sensors/

INTERNET OF THINGS

VIII Semester: ECE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC62	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		
Prerequisites: Electronic Measurements and Instrumentation								

I. COURSE OVERVIEW:

Internet of things (IoT) is a network of things that are embedded with software and sensors to process data. This course include physical and logical design of IoT systems, M2M systems, SDN, IoT Architecture components such as physical devices and endpoints, physical servers and cloud offerings. This is used in various applications such as Smart Refrigerator, Smart Homes and Smart environments.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The sensors, actuators and communication protocols used for establishing communication in M2M.
II. The significance of the internet of things to protect the privacy when using it.
III. The real time IoT applications related to smart environments and industrial applications.

III. COURSE SYLLABUS:

MODULE – I : INTRODUCTION TO INTERNET OF THINGS (IoT) (08)

Definition and characteristics of IoT, sensing, actuation,physical design of IoT, logical design of IoT, IoT enabling technologies, IoT levels and deployment.

MODULE – II : IoT AND M2M (09)

Introduction, M2M, difference between IoT and M2M, Sensor Networks, software defined networking (SDN) and network function virtualization (NFV) for IoT, basics of IoT system management with NETCONF-YANG
.

MODULE – III : IoT ARCHITECTURE AND PYTHON (10)

IoT Architecture: State of the art introduction, state of the art; Architecture reference model: Introduction, reference model and architecture, IoT reference model.

Logical design using Python: Installing Python, Python data types and data structures, control flow, functions, modules, packages, file handling

MODULE – IV : IoT PHYSICAL DEVICES AND ENDPOINTS (08)

Introduction to Raspberry Pi interfaces (Serial, SPI, I2C), programming Raspberry PI with Python, other IoT devices, Implementation of IoT with Raspberry Pi

MODULE – V : IoT PHYSICAL SERVERS AND CLOUD OFFERINGS (10)

Introduction to cloud storage models and communication APIs, Webserver – Web server for IoT, Cloud for IoT, Python web application framework Designing a RESTful web API , IoT Case studies: illustrating IoT design: Home automation, smart cities, smart environment.

IV. TEXT BOOKS:

1. Arshdeep Bahga, Vijay Madisetti, "Internet of Things: A Hands-on-Approach", VPT, 1st Edition, 2014.
2. Matt Richardson, Shawn Wallace, "Getting Started with Raspberry Pi", O"Reilly (SPD), 3rd Edition, 2014.

V. REFERENCE BOOKS:

1. Adrian McEwen, Hakim Cassimally, "Designing the Internet of Things", John Wiley and Sons, 1st Edition, 2014.
2. Francis Da Costa, "Rethinking the Internet of Things: A Scalable Approach to Connecting Everything", Apress Publications, 1st Edition, 2013.

VI. WEB REFERENCES:

1. https://www.upf.edu/pra/en/3376/22580.
2. https://www.coursera.org/learn/iot.
3. https://bcourses.berkeley.edu.
4. https://mitpress.mit.edu/books/internet-things

FOUNDATIONS OF ROBOT MOTION AND ROBOT DYNAMICS

VIII Semester: ECE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AECC63	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

Prerequisites: Electronic Measurements and Instrumentation

I. COURSE OVERVIEW:

The course emphasis on the design and developments of robot geometry, sensors and actuators to meet the kinematics requirements and trajectory planning of the manipulator. Robotics is recognized as one of the important aids of mechatronics systems and industrial automation. The applications include in manufacturing, health care and industrial automation is to minimal elimination of human intervention.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamental concepts of various configurations of the robot manipulators and their working principles used in the industries.

II. The path planning of a robot manipulator for given polynomial equation and how to avoid obstacles in its path.

III. The performance of various feedback components like sensors and actuators and how they can be used according to the specifications of the manipulator.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION TO ROBOTICS (09)

Introduction: Automation and robotic, an over view of robotics, classification by coordinate system and control systems; Components of the industrial robotics: Degrees of freedom, end effectors: Robot anatomy, Classification and usage, science and technology of robots, associated parameters: resolution, accuracy, repeatability, dexterity. general consideration on gripper selection and design.

MODULE -II: MOTION ANALYSIS AND KINEMATICS (09)

Motion analysis: Basic rotation matrices, composite rotation matrices, Euler angles, equivalent angle and axis, homogeneous transformation, problems; Manipulator kinematics: D-H notations, joint coordinates and world coordinates, forward and inverse kinematics, problems.

MODULE -III : KINEMATICS AND DYNAMICS (10)

Differential kinematics: Differential kinematics of planar and spherical manipulators, Jacobians,problems.

Robot dynamics: Lagrange, Euler formulations, Newton-Euler formulations, problems on planar two link manipulators.

MODULE -IV : TRAJECTORY PLANNING AND ACTUATORS (08)

Trajectory planning: Joint space scheme, cubic polynomial fit, avoidance of obstacles, types of motion**:** Slew motion, joint interpolated motion, straight line motion, problems; Robot actuators and feedback components; Actuators: pneumatic and hydraulic actuators.

MODULE -V : SENSORS FOR ROBOTS, ELECTRIC ACTUATORS (09)

Electric actuators: DC servo motors, stepper motors, feedback components: position sensors, Selections of sensors, Classification and applications of sensors. Types of Sensors, potentiometers, resolvers and encoders, velocity sensors, tactile sensors; Robot application in manufacturing: Material handling, working and control of a robot.

IV. TEXT BOOKS:

1. Groover M. P, “Industrial Robotics”, Tata McGraw-Hill, 1st Edition, 2013.
2. J. J Craig, “Introduction to Robotic Mechanics and Control”, Pearson, 3rd Edition, 2013.

V. REFERENCE BOOKS:

1. Adrian McEwen, Hakim Cassimally, “Designing the Internet of Things”, John Wiley and Sons, 1st Edition, 2014.
2. Francis Da Costa, “Rethinking the Internet of Things: A Scalable Approach to Connecting Everything”, Apress Publications, 1st Edition, 2013.

VI. WEB REFERENCES:

1. https://www.upf.edu/pra/en/3376/22580.
2. https://www.coursera.org/learn/iot.
3. https://bcourses.berkeley.edu.
4. https://mitpress.mit.edu/books/internet-things

FLIGHT CONTROL THEORY

OE –I: VI Semester: AERO / MECH / CIVIL								
OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AAEC30	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

Flight control system of an aircraft is instrumental in establishing stability of the aircraft through control surfaces. This course introduces the concepts of the control system theory such as transfer functions, step response and impulse response. This course covers stability, feedback and different techniques used for control systems analysis. The course emphasizes on the flight control systems, response analysis for control surface inputs and control augmentation systems such as autopilots.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The stability criteria to determine the stability of an aircraft, and specify the aircraft time-domain and frequency-domain response specifications.

II. The classical control theory in the frequency domain and modern control theory in the state- space are effectively mixed to provide the student with a modern view of systems theory.

III. The various control techniques for aircraft control systems, and study some feedback control applications.

IV. The controllability and observability of aerospace systems, and apply the modern control techniques to design enhanced flight control systems.

III. COURSE SYLLABUS:

MODULE-I: INTRODUCTION TO CONTROL SYSTEM (10)

Dynamical systems-principal constituents-input, output-process (plant)-block diagram representation. Inputs- control input, noise. Function of controls regulation (hold), tracking (command)-examples. Measure of effectiveness. Sensitivity of output to control input, noise and system parameters-robustness. Deterministic and stochastic control. Control in everyday life. The pervasiveness of control in nature, engineering and societal systems. The importance of study of control system. Need for stable, effective (responsive), robust control system. Modeling of dynamical systems by differential equations-system parameters. Examples from diverse fields. First and second order systems, higher order systems, single input single output systems, and multiple-input multiple-output.

MODULE –II: MATHEMATICAL MODELING OF DYNAMICAL SYSTEMS (10)

Control system performance- time domain description- output response to control inputs-- impulse and indicial response- characteristic parameters- significance- relation to system parameters- examples- first and second order linear systems, higher order systems. Synthesis of response to arbitrary input functions from impulse and indicial response. Review of Fourier transforms and Laplace transforms-inverse transforms- significance, applications to differential equations. 's' (Laplace) domain description of input- output relations- transfer function representation- system parameters- gain, poles and zeroes. Characteristic equation- significance- examples. Frequency and damping ratio of dominant poles. Relation of transfer functions to impulse response. Partial fraction decomposition of transfer functions-significance.

MODULE –III: STEADY STATE RESPONSE ANALYSIS (10)

System type, steady state error, error constants- overall system stability. Application of feedback in stability augmentation, control augmentation, automatic control-examples. Composition, reduction of block diagrams of complex systems-rules and conventions. Control system components - sensors, transducers, servomotors, actuators, filters-modeling, transfer functions. Single-input single-output systems. Multiple input-multiple output systems, matrix transfer functions-examples. Types of control problems- the problem of analysis, control synthesis, system synthesis- examples- static control of aircraft.

Extension to dynamic control. System identification from input output measurements importance. Flight path stabilization, longitudinal control law design using back stepping algorithm. Experimental determination of system transfer functions by frequency response measurements. Example. Frequency domain description- frequency response- gain and phase shift- significance- representation asymptotic (Bode) plots, polar (Nyquist) plots, frequency transfer functions. Characteristic parameters corner frequencies, resonant frequencies, peak gain, and bandwidth- significance. First and second order systems- extension to higher order systems.

MODULE –IV: AIRCRAFT RESPONSE TO CONTROL (07)

Approximations to aircraft transfer functions, control surface actuators-review. Response of aircraft to elevator input, Response of aircraft to rudder input and Response of aircraft to aileron input to atmosphere. Need for automatic control. Auto pilots Stability augmentation systems-pitch damper and yaw damper.

MODULE –V: FLYING QUALITIES OF AIRCRAFT (08)

Reversible and irreversible flight control systems. Flying qualities of aircraft-relation to airframe transfer function. Pilot's opinion ratings. Flying quality requirements- pole-zero, frequency response and time- response specifications. Displacement and rate feedback determination of gains conflict with pilot input s resolution-control augmentation systems- Full authority fly-by-wire. Auto Pilot-Normal acceleration, Turn rate, Pitch rate Commands-Applications.

IV. TEXT BOOKS:

1. Kuo, B.C., “Automatic control of Aircraft and Missiles”, John Wiley Sons, New York, 1990.
2. Stevens B.L & Lewis F.L, “Aircraft control & Simulation”, John Wiley Sons, New York, 1992.

V. REFERENCE BOOKS:

1. Mc Lean, D., “Automatic Flight Control Systems”, Prentice Hall, 1990.
2. Bryson, A.E., “Control of Aircraft and Spacecraft”, Princeton University Press, 1994.
3. E H J Pallett, Shawn Coyle, “Automatic Flight Control”, 4th Edition, 2002.

VI. WEB REFERENCES:

1. https://www.e-booksdirectory.com/
2. https://www.aerospaceengineering.es/book/

VII. E-TEXT BOOKS:

1. https://books.google.co.in/books?isbn=1118870972
2. https://books.google.co.in/books?isbn=0387007261

AIRFRAME STRUCTURAL DESIGN

OE –I: VI Semester: AERO / MECH / CIVIL
OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE

Course Code	Category	Hours / Week			Credits	Maximum Marks		
AAEC31	Elective	L	T	P	C	CIA	SEE	Total
		3	-	-	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

This course deals with fundamental aspects of an anatomy of aircraft and the current trends in airframe design. It includes the evolution of the aircraft and space industry, aerodynamics and performance of the aircraft with their applications. It compares and contrasts various thrust vector control mechanisms of different aircraft propulsion systems. It discusses various materials and its properties that are used for manufacturing different parts of an aircraft. This course enriches the knowledge of connection between theoretical and practical methods for performing the airframe design exercises

II. COURSE OBJECTIVES:

The student will try to learn:

I. The fundamental concepts of various airframe designs, aircraft propulsion systems and aerodynamic forces/moments acting on the aircraft and spacecraft under static and dynamic load conditions

II. The characteristics of stability and performance of an aircraft and the role of primary and secondary controls in longitudinal and lateral stability

III. The properties of different materials that are used in industries for manufacturing various components of an aircraft and spacecraft achieving specified stability requirements.

IV. The mathematical modeling of tailless aircraft, flapping wing aircraft and innovative designs in modern aircraft for future requirements.

II. COURSE SYLLABUS:

MODULE-I: HISTORY OF FLIGHT AND SPACE ENVIRONMENT (10)

Balloons and dirigibles, heavier than air aircraft, commercial air transport; Introduction of jet aircraft, helicopters, missiles; Conquest of space, commercial use of space; Different types of flight vehicles, classifications exploring solar system and beyond, a permanent presence of humans in space; Earth‘s atmosphere, the standard atmosphere; The temperature extremes of space, laws of gravitation, low earth orbit, microgravity, benefits of microgravity; Environmental impact on spacecraft, space debris; Planetary environments.

MODULE –II: INTRODUCTION TO AERODYNAMICS (10)

Anatomy of the airplane, helicopter; Understanding engineering models; Aerodynamic forces on a wing, force coefficients; Generating lift, moment coefficients; Aerodynamic forces on aircraft – classification of NACA airfoils, aspect ratio, wing loading, mach number, centre of pressure and aerodynamic centre, aerofoil characteristics-lift, drag curves; Different types of drag.

MODULE –III: FLIGHT VEHICLE PERFORMANCE AND STABILITY (09)

Performance parameters, performance in steady flight, cruise, climb, range, endurance, accelerated flight symmetric maneuvers, turns, sideslips, takeoff and landing.

Flight vehicle Stability, static stability, dynamic stability; Longitudinal and lateral stability; Handling qualities of the airplanes.

MODULE –IV: INTRODUCTION TO AIRPLANE STRUCTURES AND MATERIAL,POWERPLANT (08)

General types of construction, monocoque, semi-monocoque; Typical wing and fuselage structure; Metallic & non-metallic materials, use of aluminum alloy, titanium, stainless steel and composite materials; Basic ideas about engines, use of propellers and jets for thrust production; Principles of operation of rocket, types of rockets.

MODULE –V: SATELLITE SYSTEMS ENGINEERING HUMAN SPACE EXPLORATION (08)

Satellite missions, an operational satellite system, elements of satellite, satellite bus subsystems; Satellite structures, mechanisms and materials; Power systems; Communication and telemetry; Propulsion and station keeping; Space missions, mission objectives. Goals of human space flight missions, historical background, the Soviet and US missions; The mercury, Gemini, Apollo (manned flight to the moon), Skylab, apollo-soyuz, space Shuttle; International space station, extravehicular activity; The space suit; The US and Russian designs; Life support systems, flight safety; Indian effort in aviation, missile and space technology.

IV. TEXT BOOKS:

1. Newman D, “Interactive Aerospace Engineering and Design”, McGraw-Hill, 1st Edition, 2002.
2. Anderson J. D, “Introduction To Flight”, McGraw-Hill Education, 5th Edition, 2002.

V. REFERENCE BOOKS:

1. Kermode. A. C, “Flight without Formulae”, McGraw Hill, 4th Edition, 1997.
2. Barnard R.H and Philpot. D.R, “Aircraft Flight”, Pearson, 3rd Edition, 2004.
3. SwattonP.J, “Flight Planning”, Blackwell Publisher, 6th Edition, 2002.

VI. WEB REFERENCES:

1. http://ase.sbu.ac.ir/FA/Staff/abbasrahi/Lists/Dars/Attachments/11/Vibrations%20of%20Continuous%20Systems.pdf
2. http://arc-test.aiaa.org/doi/book/10.2514/4.862458
3. http://arc-test.aiaa.org/doi/abs/10.2514/5.9781600862373.0719.0728

VII. E-TEXT BOOKS:

1. http://www.gregorypaulblog.com/structural-dynamics-in-aeronautical-engineering-aiaa-education-series.pdf
2. https://aerocastle.files.wordpress.com/2012/10/mechanical_vibrations_5th-edition_s-s-rao.pdf

INDUSTRIAL MANAGEMENT

OE –I: VI Semester: AERO / MECH / CIVIL OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
AMEC34	**Elective**	**L**	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
		3	0	0	3	30	70	100
Contact Classes: 45	**Tutorial Classes: Nil**	**Practical Classes: Nil**				**Total Classes: 45**		

I.COURSE OVERVIEW:

The industrial management prepares engineers to design, improve, install, and operate the integrated systems of people, materials, and facilities needed by industry, commerce, and society. Industrial engineers solve problems that arise in the management of systems, applying the principles of engineering science, product/service and process design, work analysis, human factors principles, and operations research.The focus of this course is how to improve processes or design things that are more efficient and waste less money, time, raw resources, man-power and energy while following safety standards and regulations

II. COURSE OBJECTIVES:

The students will try to learn:

I. The production planning and control procedures to handle industrial disputes.
II. The Work study procedures and quality concepts to enhance more productivity
III. The significant exposure on some maintenance practices in industry for consistent productivity.

III.COURSE SYLLABUS:

MODULE-I: CONCEPTS OF INDUSTRIAL MANAGEMENT (9)

Principles of management- Growth of management thought, Functions of management, Principles of organization, Types of organization and committees.

MODULE –II: WORK STUDY (9)

Concept of productivity, Method Study - Basic steps in method study, Process charts, Diagrams, Principles of motion economy, Micro motionstudy, Therbligs, SIMO chart. Work Measurement - Stop watch procedure of time study, Performance rating, allowances, Work sampling, Simple problems.

MODULE –III: INVENTORY CONTROL (9)

Inventory Control: Inventory, Cost, Deterministic Models and Introduction to Supply Chain Management.

MODULE –IV: QUALITY CONTROL (9)

Quality Control: Process control, SQC, Control charts, Single, Double and Sequential Sampling, Introduction to TQM.

MODULE –V: DEMAND FORECASTING AND COST ESTIMATION (9)

Demand Forecasting and cost Estimation: Characteristics of Forecasts, Forecasting Horizons, Steps to Forecasting, Forecasting Methods, Seasonal Adjustments, Forecasting Performance Measures, Cost Estimation, Elements of cost, Computation of Material Variances Break-Even Analysis.

IV. TEXT BOOKS:

1. O.P. Khanna, "Industrial Engineering and Management", Khanna Publishers.
2. T.R. Banga and S.C.Sarma, "Industrial Engineering and Management Science", Khanna Publishers.

V. REFERENCE BOOKS:

1. Ralph M Barnes, “Motion and Time Study”, John Willey & Sons Work Study lLO.
2. Ernest J McCormick, “Human factors in Engineering & Design”, TMH.
3. Paneer Selvam, “Production & Operation Management”, PHI.
4. NVS Raju, “Industrial Engineering Management”, Cengage Learning.

VI. REFERENCE BOOKS:

1. https://nptel.ac.in/courses/112/107/112107142/#
2. https://nptel.ac.in/courses/112/107/112107143/#

ELEMENTS OF MECHANICAL ENGINEERING

OE –I: VI Semester: AERO / MECH / CIVIL OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
AMEC35	Elective	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Tutorial Classes: Nil	Practical Classes: Nil				Total Classes: 45		

I.COURSE OVERVIEW:

The main aim of this course to impart mechanical engineering fundamental basics to allied engineering students so that they have minimum understanding of mechanical system, equipment and process.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The fundamentals of mechanical systems.
II. The significance of mechanical engineering and apply in different fields of engineering.
III. The various applications of engineering materials for designing different engineering components.

III. COURSE SYLLABUS:

MODULE-I: SOURCES OF ENERGY, BASIC CONCEPTS OF THERMODYNAMICS (9)

Sources of Energy : Introduction and application of energy sources like fossil fuels, hydel, solar, wind, nuclear fuels and bio-fuels; environmental issues like global warming and ozone depletion.

Basic concepts of Thermodynamics: Introduction, states, concept of work, heat, temperature; Zeroth, 1st, 2nd and 3rd laws of thermodynamics. Concept of internal energy, enthalpy and entropy (simple numericals).

MODULE –II: BOILER AND TURBINES(9)

Boilers: Introduction to boilers, classification, Lancashire boiler, Babcock and Wilcox boiler. Introduction to boiler mountings and accessories (no sketches).

Turbines: Hydraulic Turbines-Classification and specification, Principles and operation of Pelton wheel turbine, Francis turbine and Kaplan turbine (elementary treatment only).

Hydraulic Pumps: Introduction, classification and specification of pumps, reciprocating pump and centrifugal pump, concept of cavitations and priming.

MODULE –III: PROPERTIES, COMPOSITION AND INDUSTRIAL APPLICATIONS OF ENGINEERING MATERIALS(9)

Metals-Ferrous: cast iron, tool steels and stainless steels and nonferrous: aluminum, brass, bronze. Polymers -Thermoplastics and thermosetting polymers. Ceramics -Glass, optical fiber glass, cermets. Composites -Fiber reinforced composites, Metal Matrix Composites, Smart materials -Piezoelectric materials, shape memory alloys, semiconductors and insulators.

Joining Processes: Soldering, Brazing and Welding Definitions. Classification and methods of soldering, brazing and welding. Brief description of arc welding, oxy-acetylene welding, TIG welding, and MIG welding.

MODULE –IV: MACHINE TOOLS(9)

Lathe -Principle of working of a center lathe. Parts of a lathe. Operations on lathe –Turning, Facing, Knurling, Thread Cutting, Drilling, Taper turning by Tailstock offset method and Compound slide swiveling method, Specification of Lathe.

Milling Machine-Principle of milling, types of milling machines. Working of horizontal and vertical milling machines. Milling processes -plane milling, end milling, slot milling, angular milling, form milling, straddle milling, and gang milling.

MODULE –V: INTRODUCTION TO ADVANCED MANUFACTURING SYSTEMS (9)

Computer Numerical Control (CNC): Introduction. Components of CNC, open loop and closed loop systems, advantages of CNC, CNC Machining centers and Turning centers.

Robots: Robot anatomy, joints and links, common robot configurations. Applications of Robots in material handling, processing and assembly and inspection

IV.TEXT BOOKS

1. V. K. Manglik, "Elements of Mechanical Engineering", Prentice Hall, 1st Edition, 2013.
2. Mikell P. Groover, "Automation, Production Systems and CIM", Prentice Hall, 4th Edition, 2013

V.REFERENCE BOOKS:

1. S. Trymbaka Murthy, "A Text Book of Elements of Mechanical Engineering", University Press, 4th Edition, 2006.
2. K. P. Roy, S. K. Hajra Choudary, Nirjhar Roy, "Element of Mechanical Engineering", Media Promoters & Publishers, 7th Edition, 2012.
3. Pravin Kumar, "Basic Mechanical Engineering", Pearson, 1st Edition, 2013

VI.WEB REFERENCES:

1. http://www.nptel.ac.in/courses/112107144/
2. http://www.nptel.ac.in/courses/112101098/download/lecture-37.pdf

MODERN CONSTRUCTION MATERIALS

OE –I: VI Semester: AERO / MECH / CIVIL OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE								
Course Code	Category	Hours / Week			Credits	Maximum Marks		
ACEC30	Elective	L	T	P	C	CIA	SEE	Total
		3	0	0	3	30	70	100
Contact Classes: 45	Total Tutorials: Nil	Total Practical Classes: Nil				Total Classes: 45		

I. COURSE OVERVIEW:

This course provides the scientific basis for the understanding and development of construction materials. It serves as a foundation course for post-graduate students interested in careers involving research, teaching and/or construction engineering, as well as marketing, decision making, innovation and specification related to construction materials.

II. COURSE OBJECTIVES:

The student will try to learn:

I. The concept of modern water proofing and insulating materials in constructions.
II. Importance of composites and chemicals in production of modern concrete.
III. The types of concrete and their constituents and properties.
IV. The impact of building construction on society and demonstrate awareness of contemporary issues.

III.COURSE SYLLABUS:

MODULE-I: STONES – BRICKS – CONCRETE BLOCKS (09)

Stone as building material, Criteria for selection, Tests on stones, Deterioration and Preservation of stone work, Bricks, Classification, Manufacturing of clay bricks, Tests on bricks Compressive Strength, Water Absorption, Efflorescence, Bricks for special use, Refractory bricks, Cement, Concrete blocks, Light-weight concrete blocks.

MODULE-II: LIME – CEMENT – AGGREGATES – MORTAR (09)

Lime, Preparation of lime mortar, Cement, Ingredients, Manufacturing process, Types and Grades, Properties of cement and Cement mortar, Hydration, Compressive strength, Tensile strength, Fineness, Soundness and consistency, Setting time, Industrial byproducts, Fly ash, Aggregates, Natural stone aggregates, Crushing strength, Impact strength, Flakiness Index, Elongation Index, Abrasion Resistance, Grading, Sand Bulking.

MODULE-III: CONCRETE (09)

Concrete, Ingredients, Manufacturing Process, Batching plants, RMC, Properties of fresh concrete, Slump, Flow and compaction Factor, Properties of hardened concrete, Compressive, Tensile and shear strength.

Modulus of rupture, Tests, Mix specification, Mix proportioning, BIS method, High Strength Concrete and HPC, Self compacting Concrete, Other types of Concrete, Durability of Concrete.

MODULE-IV: TIMBER AND OTHER MATERIALS (09)

Timber, Market forms, Industrial timber, Plywood, Veneer, Thermacole, Panels of Laminates, Steel, Aluminum and Other Metallic Materials, Composition, Aluminium composite panel, Uses: Market forms, Mechanical treatment, Paints, Varnishes, Distempers, Bitumens.

MODULE-V: MODERN MATERIALS (09)

Glass, Ceramics, Sealants for joints, Fibre glass reinforced plastic, Clay products, Refractories, Composite materials, Types, Applications of laminar composites, Fibre textiles, Geomembranes and Geotextiles for earth reinforcement.

IV.TEXT BOOKS:

1. W.D. Callister, John Wiley, “Materials Science and Engineering: An Introduction”, John Wiley & Sons, Inc. 1994.
2. P.C. Varghese, “Building Materials”, Prentice-Hall India, 2005.

V. REFERENCE BOOKS:

1. V. Raghavan, “Materials Science and Engineering”, Prentice Hall, 1990.
2. R.A. Higgins, “Properties of Engineering Materials”, Industrial Press, 1994.
3. Eds. J.M. Illston and P.L.J. Domone, “Construction Materials: Their nature and behaviour”, Spon Press, 3rd Edition, 2002

VI.WEB REFERENCES:

1. https://www.scribd.com/document/394619658/Material-Science-and-Engineering-V-Raghavan-pdf
2. https://files.isec.pt/DOCUMENTOS/SERVICOS/BIBLIO/INFORMA%C3%87%C3%95ES%20ADICIONAIS/Materials-for-engineers-5ed_Higgins.pdf

VII. E-TEXT BOOKS:

1. https://onlinecourses.nptel.ac.in/noc20_ce05/preview
2. http://kaizenha.com/wp-content/uploads/2016/04/Materials-Textbook-8th-Edition.pdf

DISASTER MANAGEMENT

OE –I: VI Semester: AERO / MECH / CIVIL **OE – III: VIII Semester: CSE / CSE (AI & ML) / CSE (DS) / CSE (CS) / CSIT / IT / ECE / EEE**								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
ACEC31	**Elective**	3	0	0	3	30	70	100
Contact Classes: 45	**Total Tutorials: Nil**	**Total Practical Classes: Nil**				**Total Classes: 45**		

I. COURSE OVERVIEW:

The Disaster management provides a fundamental understanding of different aspects. It deals with the concepts and functions of disaster management to build competencies of professionals and development practitioners. It provides effective supporting environment by the governmental locating substantial resources for effective mitigation of disasters. It helps learners to apply the disaster mitigation strategies, preparedness for reducing damage intensity, loss of life and property.

II. COURSE OBJECTIVES:

The student will try to learn:

I. The concept of environmental hazards, disasters and various approaches dealing with the mitigation of disasters.

II. The knowledge on various types of environmental disasters and their impacts on human beings and nature.

III. The Different types of endogenous and exogenous hazards and their influence on human life and nature.

IV. The immediate response and damage assessment with information reporting and monitoring tools.

III.COURSE SYLLABUS:

MODULE-I: ENVIRONMENTAL HAZARDS AND DISASTERS (09)

Environmental hazards and disasters: meaning of environmental hazards, environmental disasters and environmental stress; concept of environmental hazards, environmental stress and environmental disasters, different approaches and relation with human ecology, landscape approach, ecosystem approach, perception approach, human ecology and its application in geographical researches.

MODULE-II: TYPES OF ENVIRONMENTAL HAZARDS AND DISASTERS (09)

Types of environmental hazards and disasters: Natural hazards and disasters, man induced hazards and disasters, natural hazards, planetary hazards/ disasters, extra planetary hazards/ disasters, planetary hazards, endogenous hazards, exogenous hazards.

MODULE-III: ENDOGENOUS HAZARDS (09)

Endogenous hazards, volcanic eruption, earthquakes, landslides, volcanic hazards/ disasters, causes and distribution of volcanoes, hazardous effects of volcanic eruptions, environmental impacts of volcanic eruptions.

Earthquake hazards/ disasters, causes of earthquakes, distribution of earthquakes, hazardous effects of, earthquakes, earthquake hazards in India, human adjustment, perception and mitigation of earthquake

MODULE-IV: EXOGENOUS HAZARDS (09)

Exogenous hazards/ disasters, infrequent events, cumulative atmospheric hazards/ disasters; Infrequent events: Cyclones , lightning , hailstorms; Cyclones: Tropical cyclones and local storms, destruction by tropical cyclones and local storms (causes, distribution human adjustment, perception and mitigation);

Cumulative atmospheric hazards/ disasters: Floods, droughts, cold waves, heat waves floods; Causes of floods, flood hazards India, flood control measures (human adjustment, perception and mitigation); Droughts: Impacts of droughts, drought hazards in India, drought control measures, extra planetary hazards/ disasters, man induced hazards /disasters, physical hazards/ disasters, soil erosion, Soil erosion: Mechanics and forms of soil erosion, factors and causes of soil erosion, conservation measures of soil erosion; Chemical hazards/ disasters: Release of toxic chemicals, nuclear explosion, sedimentation processes; Sedimentation processes: Global sedimentation problems regional sedimentation problems, sedimentation and environmental problems, corrective measures of erosion and sedimentation, biological hazards/ disasters, population explosion.

MODULE-V: EMERGING APPROACHES IN DISASTER MANAGEMENT (09)

Emerging approaches in Disaster Management, Three Stages

1. Pre, disaster stage(preparedness)
2. Emergency Stage
3. Post Disaster stage, Rehabilitation.

IV.TEXT BOOKS:

1. Pardeep Sahni, "Disaster Mitigation: Experiences and Reflections", PHI Learning Pvt. Ltd., 1st Edition, 2001.
2. J.Glynn,Gary W.HeinKe, "Environmental Science and Engineering", Prentice Hall Publishers, 2nd Edition, 1996.

V. REFERENCE BOOKS:

1. B. C. Punmia, Ashok K Jain and Arun K Jain, "Mechanics of Materials", Laxmi Publications Pvt. Ltd., New Delhi, 12th Edition, 2007.
2. R. Subramanian, "Strength of Materials", Oxford University Press, 2nd Edition, 2010.
3. D. S. Prakash Rao, "Strength of Materials A Practical Approach Vol.1", Universities Press (India) Pvt. Ltd., India, 3rd Edition, 2007.
4. J. M. Gere, S.P. Timoshenko, "Mechanics of Materials, SI units edition", CL Engineering, USA, 5th Edition, 2000.

VI. Web References:

1. https://www.google.co.in/?gfe_rd=cr&ei=,iAwWLiDIazv8we8_5LADA#q=disater+mangement
2. http://ndma.gov.in/images/policyplan/dmplan/National%20Disaster%20Management%20Plan%20May%202016.pdf
3. http://www.eib.europa.eu/attachments/pipeline/20080021_eia_en.pdf
4. http://www.ndmindia.nic.in/

VII. E-Text Books:

1. http://cbse.nic.in/natural%20hazards%20&%20disaster%20management.pdf\
2. http://www.digitalbookindex.org/_search/search010emergencydisastera.asp
3. http://www.icbse.com/books/cbse,ebooks,download

PROJECT WORK - II

VIII Semester: Common for all branches

Course Code	Category	Hours / Week			Credits	Maximum Marks		
		L	T	P	C	CIA	SEE	Total
AECC64	Project	-	-	16	8	30	70	100
Contact Classes: Nil	Tutorial Classes: Nil	Practical Classes: 180				Total Classes: 180		

The object of Project Work II & Dissertation is to enable the student to extend further the investigative study taken up under EC P1, either fully theoretical/practical or involving both theoretical and practical work, under the guidance of a Supervisor from the Department alone or jointly with a Supervisor drawn from R&D laboratory/Industry. This is expected to provide a good training for the student(s) in R&D work and technical leadership. The assignment to normally include:

1. In depth study of the topic assigned in the light of the Report prepared under EC P1;
2. Review and finalization of the Approach to the Problem relating to the assigned topic;
3. Preparing an Action Plan for conducting the investigation, including team work;
4. Detailed Analysis / Modeling / Simulation / Design / Problem Solving / Experiment as needed;
5. Final development of product/process, testing, results, conclusions and future directions;
6. Preparing a paper for Conference presentation/Publication in Journals, if possible;
7. Preparing a Dissertation in the standard format for being evaluated by the Department.
8. Final Seminar Presentation before a Departmental Committee.

INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad - 500 043

UNDERTAKING BY STUDENT / PARENT

“To make the students attend the classes regularly from the first day of starting of classes and be aware of the College regulations, the following Undertaking Form is introduced which should be signed by both student and parent. The same should be submitted to the Dean of Academic”.

I, Mr. / Ms. -- joining I Semester / III Semester for the academic year 20 - 20 / 20 - 20 in Institute of Aeronautical Engineering, Hyderabad, do hereby undertake and abide by the following terms, and I will bring the ACKNOWLEDGEMENT duly signed by me and my parent and submit it to the Dean of Academic.

1. I will attend all the classes as per the timetable from the starting day of the semester specified in the institute Academic Calendar. In case, I do not turn up even after two weeks of starting of classes, I shall be ineligible to continue for the current academic year.
2. I will be regular and punctual to all the classes (theory/laboratory/project) and secure attendance of not less than 75% in every course as stipulated by Institute. I am fully aware that an attendance of less than 65% in more than 60% of theory courses in a semester will make me lose one year.
3. I will compulsorily follow the dress code prescribed by the college.
4. I will conduct myself in a highly disciplined and decent manner both inside the classroom and on campus, failing which suitable action may be taken against me as per the rules and regulations of the institute.
5. I will concentrate on my studies without wasting time in the Campus/Hostel/Residence and attend all the tests to secure more than the minimum prescribed Class/Sessional Marks in each course. I will submit the assignments given in time to improve my performance.
6. I will not use Mobile Phone in the institute premises and also, I will not involve in any form of ragging inside or outside the campus. I am fully aware that using mobile phone to the institute premises is not permissible and involving in Ragging is an offence and punishable as per JNTUH/UGC rules and the law.
7. I declare that I shall not indulge in ragging, eve-teasing, smoking, consuming alcohol drug abuse or any other anti-social activity in the college premises, hostel, on educational tours, industrial visits or elsewhere.
8. I will pay tuition fees, examination fees and any other dues within the stipulated time as required by the Institution / authorities, failing which I will not be permitted to attend the classes.
9. I will not cause or involve in any sort of violence or disturbance both within and outside the college campus.
10. If I absent myself continuously for 3 days, my parents will have to meet the HOD concerned/ Principal.
11. I hereby acknowledge that I have received a copy of IARE – UG20 Academic Rules and Regulations, Syllabus copy and hence, I shall abide by all the rules specified in it.

ACKNOWLEDGEMENT

I have carefully gone through the terms of the undertaking mentioned above and I understand that following these are for my/his/her own benefit and improvement. I also understand that if I/he/she fail to comply with these terms, shall be liable for suitable action as per Institute/JNTUH/AICTE/UGC rules and the law. I undertake that I/he/she will strictly follow the above terms.

Signature of Student with Date

Signature of Parent with Date
Name & Address with Phone Number