INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad – 500043

COMPUTER SCIENCE AND ENGINEERING

List of Laboratory Experiments

DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
ACSC15	**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**		
Branch: CSE	**Semester: IV**	**Academic Year: 2021-22**				**Regulation: UG20**		

Course overview:

Design and analysis of algorithms is the process of finding the computational complexity of algorithms. It helps to design and analyse the logic on how the algorithm will work before developing the actual code for a program. It focuses on implementation of sorting and searching algorithms, problem solving approaches using divide and conquer, greedy method, dynamic programming, backtracking, branch and bound.

Course objectives:

The students will try to learn:

1. The problem analysis and design the solution for the given problem.
2. The suitable algorithm for the given real-world problem.

Course outcomes:

After successful completion of the course, students will be able to:

CO1 Apply Divide and conquer strategy to organize the data in ascending or descending order.

CO2 Make use of Algorithmic Design paradigms to determine shortest distance and transitive closure of Directed or Undirected Graphs

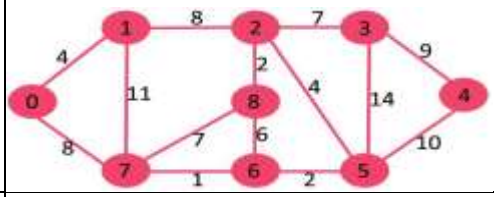
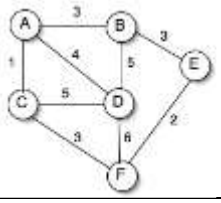
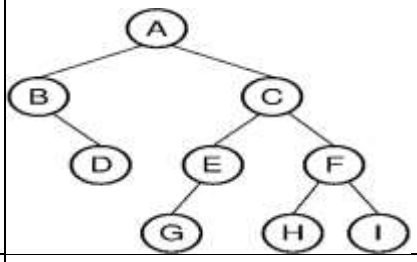
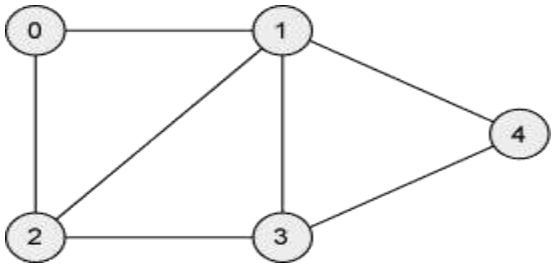
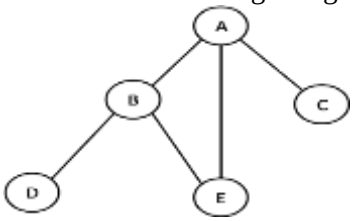
CO3 Utilize Greedy Technique or principle of optimality for finding solutions to optimization problems.

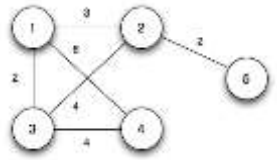
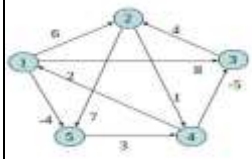
CO4 Compare the efficiencies of traversal problems using different Tree and Graph traversal algorithms.

CO5 Utilize Backtracking method for solving puzzles involving building solutions incrementally

CO6 Examine Branch and Bound Approach for solving Combinatorial optimization problems.

WEEK NO	EXPERIMENT NAME	CO
WEEK – I	**QUICK SORT** Sort a given set of elements using the quick sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the 1st to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.	**CO1**
WEEK – II	**MERGE SORT** Implement merge sort algorithm to sort a given set of elements and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.	**CO1**
WEEK – III	**KNAPSACK PROBLEM** Implement 0/1 Knapsack problem using Dynamic Programming.	**CO3**
WEEK – IV	**SHORTEST PATHS ALGORITHM** From a given vertex in a weighted connected graph, find shortest paths from 0 to other vertices using Dijkstra's algorithm.	**CO2** **CO3**

		
WEEK – V	**MINIMUM COST SPANNING TREE** Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm. 	**CO3**
WEEK – VI	**TREE TRAVESRSALS** Perform various tree traversal algorithms for a given tree. 	**CO4**
WEEK – VII	**GRAPH TRAVERSALS** a. Print all the nodes reachable from a given starting node in a digraph using BFS method.  b. Check whether a given graph is connected or not using DFS method. 	**CO4**
WEEK –VIII	**SUM OF SUB SETS PROBLEM** Find a subset of a given set S = {sl, s2,....., sn} of n positive integers whose sum is equal to a given positive integer d. For example, if S= {1, 2, 5, 6, 8} and d = 9 there are two solutions {1, 2, 6} and {1,8}.A suitable message is to be displayed if the given problem instance doesn't have a solution.	**CO3**
WEEK - IX	**TRAVELLING SALES PERSON PROBLEM**	**CO3**

	Implement any scheme to find the optimal solution for the Traveling Sales Person problem and then solve the same problem instance using any approximation algorithm and determine the error in the approximation.	**CO5**
WEEK - X	**MINIMUM COST SPANNING TREE**	**CO3**
	Find Minimum Cost Spanning Tree of a given undirected graph using Prim‘s algorithm. 	
WEEK – XI	**ALL PAIRS SHORTEST PATHS**	**CO3**
	Implement All Pairs Shortest Paths Problem using Floyd's algorithm.  |	1	2	3	4	5
---	---	---	---	---	---
1	0	6	8	∞	-4
2	∞	0	∞	1	7
3	∞	4	0	∞	∞
4	2	∞	-5	0	∞
5	∞	∞	∞	3	0	
WEEK – XII	**N QUEENS PROBLEM**	**CO5**
	Implement N Queen's problem using Back Tracking.	