

DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY

III Semester: CSE / IT									
Course Code	Category	Hours / Week			Credits	Maximum Marks			
AIT101	Core	L	T	P	C	CIA	SEE	Total	
		-	-	3	2	30	70	100	
Contact Classes: Nil		Tutorial Classes: Nil		Practical Classes: 39			Total Classes: 39		
I. COURSE OVERVIEW:									
<p>Design and analysis of algorithm lab provides hands on experience in implementing different algorithmic paradigms and develops competence in choosing appropriate data structure to improve efficiency of technique used. This laboratory implements sorting techniques using divide and conquer strategy, shortest distance algorithms based on Greedy, Dynamic programming techniques, Minimum spanning tree construction and applications of Back tracking , Branch and Bound. This is essential for developing software in areas Information storage and retrieval, Transportation through networks, Graph theory and Optimization problems.</p>									
II. OBJECTIVES:									
The course should enable the students to:									
<p>I The selection of Algorithmic technique and Data structures required for efficient development of technical and engineering applications.</p> <p>II The algorithmic design paradigms and methods for identifying solutions of optimization problems..</p> <p>III Implementation of different algorithms for the similar problems to compare their performance.</p>									
III. COURSE OUTCOMES:									
After successful completion of the course, students should be able to:									
CO 1	Apply Divide and conquer strategy to organize the data in ascending or descending order. .							Apply	
CO 2	Make use of Algorithmic Design paradigms to determine shortest distance and transitive closure of Directed or Undirected Graphs.							Apply	
CO 3	Utilize Greedy Technique or principle of Optimality for finding solutions to optimization problems.							Analyze	
CO 4	Compare the efficiencies of traversal problems using different Tree and Graph traversal algorithms.							Apply	
CO 5	Utilize Backtracking method for solving Puzzles involving building solutions incrementally.							Analyze	
CO 6	Examine Branch and Bound Approach for solving Combinatorial optimization problems.							Apply	
IV. SYLLABUS:									
LIST OF EXPERIMENTS									
Week-1	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 1 st 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.									
Week-2	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									

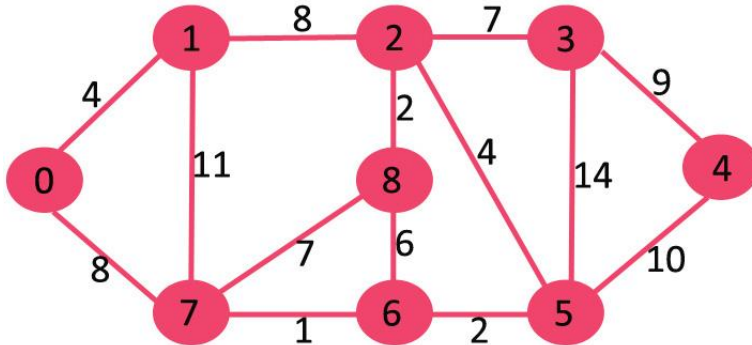
and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

Week-3 **KNAPSACK PROBLEM**

Implement 0/1 Knapsack problem using Dynamic Programming.

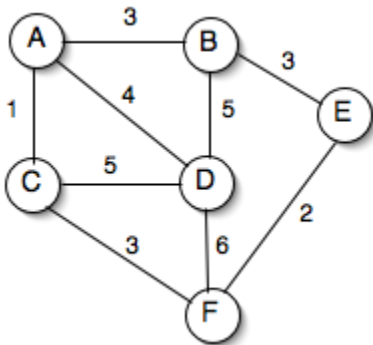
Week-4 **SHORTEST PATHS ALGORITHM**

From a given vertex in a weighted connected graph, find shortest paths from 0 to other vertices using Dijkstra's algorithm.



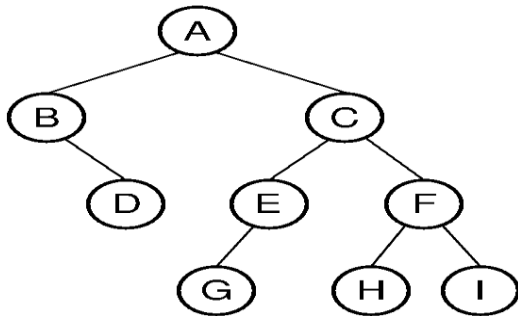
Week-5 **MINIMUM COST SPANNING TREE**

Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.



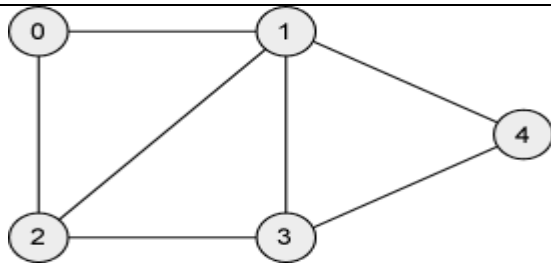
Week-6 **TREE TRAVERSALS**

Perform various tree traversal algorithms for a given tree.

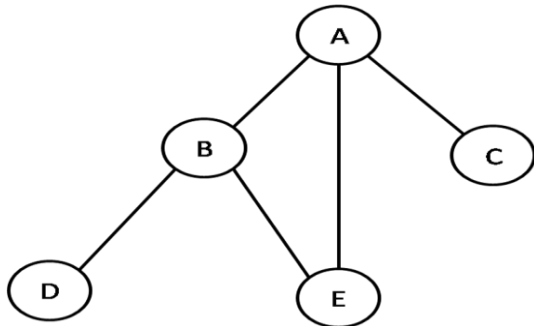


Week-7 **GRAPH TRAVERSALS**

a. Print all the nodes reachable from a given starting node in a digraph using BFS method.

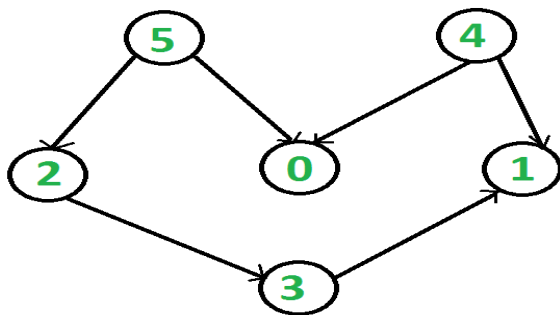


b. Check whether a given graph is connected or not using DFS method.



Week-8 | **WARSHALL'S ALGORITHM**

a. Obtain the all pair shortest path



b. Compute the transitive closure of a given directed graph using Warshall's algorithm.

Week-9 | **SUM OF SUB SETS PROBLEM**

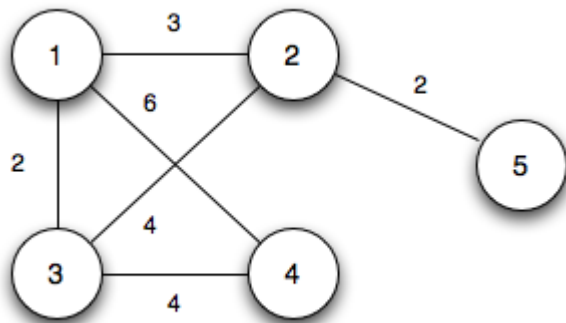
Find a subset of a given set $S = \{s_1, s_2, \dots, s_n\}$ of n positive integers whose sum is equal to a given positive integer d . For example, if $S = \{1, 2, 5, 6, 8\}$ and $d = 9$ there are two solutions $\{1, 2, 6\}$ and $\{1, 8\}$. A suitable message is to be displayed if the given problem instance doesn't have a solution.

Week-10 | **TRAVELLING SALES PERSON PROBLEM**

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.

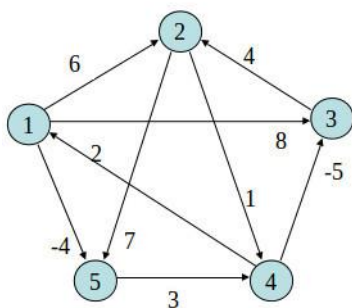
Week-11 | **MINIMUM COST SPANNING TREE**

Find Minimum Cost Spanning Tree of a given undirected graph using Prim's algorithm.



Week-12 | **ALL PAIRS SHORTEST PATHS**

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-13 | **N QUEENS PROBLEM**

Implement N Queen's problem using Back Tracking.

Reference Books:

1. Levitin A, "Introduction to the Design and Analysis of Algorithms", Pearson Education, 2008.
2. Goodrich, M.T. R Tomassia, "Algorithm Design foundations Analysis and Internet Examples", John Wiley and Sons, 2006.
3. Base Sara, Allen Van Gelder, "Computer Algorithms Introduction to Design and Analysis", Pearson, 3rd Edition, 1999.

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>

Course Home Page:

SOFTWARE AND HARDWARE REQUIREMENTS FOR A BATCH OF 36 STUDENTS:

HARDWARE: Desktop Computer Systems: 36 nos
SOFTWARE: C programming Compiler (Open source)