ADVANCED DATA STRUCTURES LABORATORY

I Semester: CSE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
BCSC11	**Core**	0	0	4	2	30	70	100
Contact Classes: Nil	**Tutorial Classes: Nil**	**Practical Classes: 36**				**Total Classes:36**		

I. COURSE OVERVIEW:

It covers the design and analysis of fundamental data structures and engages learners to use advanced data structures as tools to algorithmically design efficient computer programs that will cope with the complexity of actual applications. This course is essential for image viewer software, music players, multi-player game using data structures.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The linear and nonlinear data structures and their implementations.
II. Algorithms based on their time and space complexity.
III. Appropriate data structure and algorithm design method for a specific application.
IV. The graph traversals algorithms to solve real-world challenges such as finding shortest paths on huge maps and assembling genomes from millions of pieces.

III. COURSE OUTCOMES:

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

CO1	**Design** and analyze a divide and conquer algorithm using data structures and ADT/libraries	Apply
CO2	**Use** stack operations for evaluating mathematical expressions.	Understand
CO3	**Demonstrate** collision resolution techniques with hashing technique	Apply
CO4	**Implement** set operations using union operations	Analyze
CO5	**Use** tree traversal algorithms for solving graph applications.	Evaluate

IV. SYLLABUS:

Week-1: DIVIDE AND CONQUER – 1

a. Implement Quick Sort on 1D array of Student structure (contains student name, student_roll_no, total_marks), with key as student_roll_no and count the number of swapperformed.
b. Implement Merge Sort on 1D array of Student structure (contains student_name, student_roll_no, total_marks), with key as student_roll_no and count the number of swapperformed

[Week-2: DIVIDE AND CONQUER – 2

a. Design and analyze a divide and conquer algorithm for following maximum sub-array sum problem: given an array of integer's find a sub-array [a contagious portion of the array] which gives the maximum sum.
b. Design a binary search on 1D array of Employee structure (contains employee_name, emp_no, emp_salary), with key as emp_no and count the number of comparison happened.

Week-3: IMPLEMENTATION OF STACK AND QUEUE

a. Implement 3-stacks of size 'm' in an array of size 'n' with all the basic operations such as Is

Empty(i), Push(i), Pop(i), Is Full(i) where ‘i’ denotes the stack number (1,2,3), Stacks are not overlapping each other.

b. Design and implement Queue and its operations using Arrays.

Week-4: HASHING TECHNIQUES

Write a program to store k keys into an array of size n at the location computed using a hash function, loc = key % n, where k<=n and k takes values from [1 to m], m>n. To handle the collisions use the following collision resolution techniques

a. Linear probing
b. Quadratic probing
c. Random probing
d. Double hashing/rehashing

Week-5: APPLICATIONS OF STACK

Write C programs for the following:

a. Uses Stack operations to convert infix expression into post fix expression.
b. Uses Stack operations for evaluating the post fix expression.

Week-6: BINARY SEARCH TREE

Write a program for Binary Search Tree to implement following operations:

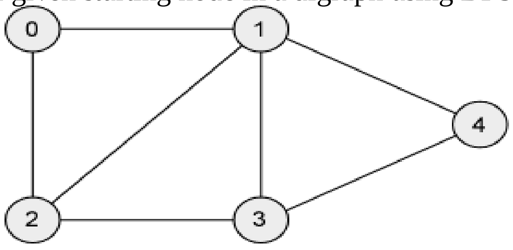
a. Insertion
b. Deletion
    i. Delete node with only child
    ii. Delete node with both children
c. Finding an element
d. Finding Min element
e. Finding Max element
f. Left child of the given node
g. Right child of the given node
h. Finding the number of nodes, leaves nodes, full nodes, ancestors, descendants.

Week-7: DISJOINT SET OPERATIONS

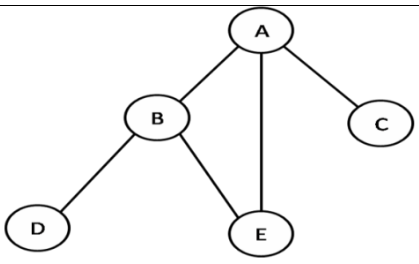
a. Write a program to implement Make_Set, Find_Set and Union functions for Disjoint Set Data Structure for a given undirected graph G(V,E) using the linked list representation with simple implementation of Unionoperation.
b. Write a program to implement Make_Set, Find_Set and Union functions for Disjoint Set Data Structure for a given undirected graph G(V,E) using the linked list representation with weighted-union heuristic approach.

Week-8: GRAPH TRAVERSAL TECHNIQUES

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

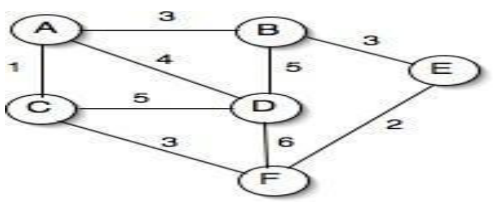


a. Check whether a given graph is connected or not using DFS method.



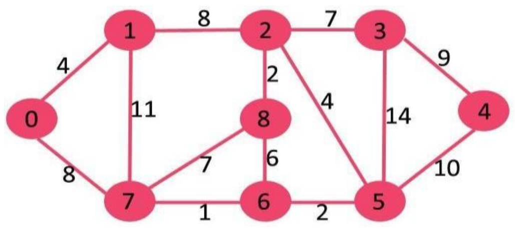
Week-9: SHORTEST PATHS ALGORITHM

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



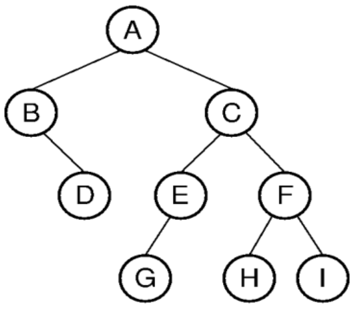
Week-10: MINIMUM COST SPANNING TREE

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



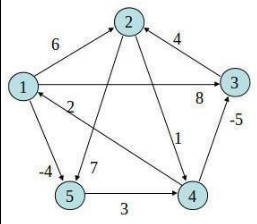
Week-11: TREE TRAVESRSALS

Perform various tree traversal algorithms for a given tree.



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

V. REFERENCE BOOKS:

1. Kernighan Brian W, Dennis M. Ritchie, “The C Programming Language”, Prentice Hall of India, Re- Print, 2008.
2. Balagurusamy E, “Programming in ANSIC”, Tata McGraw Hill, 6th Edition, 2008.
3. Gottfried Byron, “Schaum's Outline of Programming with C”, Tata McGraw Hill, 1st Edition, 2010.
4. Lipschutz Seymour, “Data Structures Schaum's Outlines Series”, Tata McGraw Hill, 3rd Edition, 2014.
5. HorowitzEllis, SatrajSahni,SusanAnderson,Freed,“FundamentalsofDataStructuresinC”,W.H. Freeman Company, 2nd Edition, 2011.

VI. WEB REFERENCES:

1. http://www.tutorialspoint.com/data_structures_algorithms
2. http://www.geeksforgeeks.org/data-structures/
3. http://www.studytonight.com/data-structures/
4. http://www.coursera.org/specializations/data-structures-algorithms