Educate Elevate Enlighten

JSS Mahavidyapeetha

JSS Science And Technology University

(Established Under JSS Science and Technology University Act No. 43 of 2013)

JSS SCIENCE AND TECHNOLOGY UNIVERSITY MYSURU

Course Title: Design and Analysis of Algorithms Lab	***Course Code: 20CS47L***
Credits (L: T:P): 0:0:1.5	***Contact Hours (L: T: P): 0:0:39***
Type of Course: Practical	***Category: Professional Core Course***
CIE Marks: 50	***SEE Marks: 50***

Pre-requisite: Data Structures

Course Outcomes: After completing this course, students should be able to:

CO-1	Analyze the problem domain; Choose the appropriate data structures and design technique based on the problem domain.
CO-2	Implement algorithms and perform analysis with empirical method.
CO-3	Evaluate the performance of different algorithms using different design techniques for solving the same problem.

Note: While demonstrating the results, students are required to show the correctness of the algorithms followed by analysis.

Unit No.	**List of Programs**
1	Implement Euclid’s, Consecutive integer checking and Modified Euclid’s algorithms to find GCD of two nonnegative integers and perform comparative analysis by generating best case and worst case data.
2	Implement the following searching algorithms and perform their analysis by generating best case and worst case data. a) Sequential Search b) Binary Search(Recursive)
3	Implement the following elementary sorting algorithms and perform their analysis by generating best case and worst case data. (Note: Any two may be asked in the test/exam) a) Selection Sort b) Bubble Sort c) Insertion Sort

4	Implement Brute force string matching algorithm to search for a pattern of length ‘M’ in a text of length ‘N’ (M<=N) and perform its analysis by generating best case and worst case data.
5	Implement Merge Sort algorithm and perform its analysis by generating best case and worst case data.
6	Implement Quick Sort algorithm and perform its by generating best case and worst case data.
7	Implement DFS algorithm to check for connectivity and acyclicity of a graph. If not connected, display the connected components. Perform its analysis by generating best case and worst case data. **Note: while showing correctness, input should be given for both connected/disconnected and cyclic/acyclic graphs.**

8	Implement BFS algorithm to check for connectivity and acyclicity of a graph. If not connected, display the connected components. Perform its analysis by generating best case and worst case data. **Note: while showing correctness, Input should be given for both connected/disconnected and cyclic/acyclic graphs.**
9	Implement DFS based algorithm to list the vertices of a directed graph in Topological ordering. Perform its analysis giving minimum 5 graphs with different number of vertices and edges. (starting with 4 vertices). **Note: while showing correctness, input should be given for with and without solution.**
10	Implement source removal algorithm to list the vertices of a directed graph in Topological ordering. Perform its analysis giving minimum 5 graphs with different number of vertices and edges. (starting with 4 vertices). **Note: Use efficient method to identify the source vertex.** **While showing correctness, Input should be given for with and without solution.**
11	Implement heap sort algorithm with bottom-up heap construction. Perform its analysis by generating best case and worst case data.

12	a) Implement Warshall’s Algorithm to find the transitive closure of a directed graph and perform its analysis giving minimum 5 graphs with different number of vertices and edges. (starting with 4 vertices). b) Implement Floyd’s Algorithm to find All-pair shortest paths for a graph and perform its analysis giving minimum 5 graphs with different number of vertices and edges(starting with 4 vertices).
13	a) Implement bottom up Dynamic Programming algorithm to solve Knapsack problem and perform its analysis with different instances (different number of items and Capacity, starting with 4 items) b) Implement a Dynamic Programming algorithm with Memory function to solve Knapsack problem and perform its analysis with different instances (different number of items and Capacity, starting with 4 items).
14	Implement Prim’s algorithm to find Minimum Spanning Tree of a graph and perform its analysis giving minimum 5 graphs with different number of vertices and edges (starting with 4 vertices).
15	Implement Dijkstra's algorithm to find the shortest path from a given source to all other vertices and perform its analysis giving minimum 5 graphs with different number of vertices and edges(starting with 4 vertices).