DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT	**Computer Science and Engineering**						
Course Code	***22CS47L***	***Total Credits***	***1.5***	***Course Type***	***Professional Core Course***		
Course Title	***Design and Analysis of Algorithms Lab***						
Teaching Learning Process		***Contact Hours***	***Credits***	***Assessment in Weightage and marks***			
	Lecture	**0**	**0**		**CIE**	**SEE**	**Total**
	Tutorial	**0**	**0**	***Weightage***	***40 %***	***60 %***	***100 %***
	Practical	**39**	**1.5**	***Maximum Marks***	***40 Marks***	***60 Marks***	***100 Marks***
	Total	**39**	**1.5**	***Minimum Marks***	***20 marks***	***25 marks***	***45 Marks***

Note: *For passing the student has to score a minimum of 45 Marks (CIE+SEE: 20 + 25 or 21 + 24)

Note: Students are required to show the correctness of the program by reading inputs. After showing correctness, analysis to be carried out by generating the data.

Program 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. Plot the Graphs.
2	Implement the following searching algorithms and perform their analysis for worst case, best-case and average inputs. Plot the Graphs a) Sequential Search b) Binary Search(Recursive)
3	Implement the following elementary sorting algorithms and perform their analysis for worst case, best-case and average inputs. Plot the Graphs. (Any two may be asked in the exam/test) 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. Plot the Graphs
5	Implement Merge Sort algorithm and perform its analysis by generating best case and worst case data. Plot the Graphs
6	Implement Quick Sort algorithm and perform its analysis by generating best case and worst case data. Plot the Graphs

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. Plot the Graphs for different cases.(Use array storage representation)
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. Plot the Graphs for different cases.(Use array storage representation)
9	Implement DFS based algorithm to list the vertices of a directed graph inTopological ordering. Perform its analysis by generating best case and worst case data. Plot the Graphs for different cases.(Use array storage representation)
10	Implement source removal algorithm to list the vertices of a directed graph inTopological ordering (Note: Use efficient method to identify the source vertex). Perform its analysis by generating best case and worst case data. Plot the Graphs for different cases. (Use array storage representation).
11	Implement heap sort algorithm with bottom-up heap construction. Perform its analysis by generating best case and worst case data. Plot the Graphs for different cases.
12	a) Implement Warshall's Algorithm to find the transitive closure of a directed graph and perform its analysis for different inputs. Plot the Graphs b) Implement Floyd's Algorithm to find shortest paths between every pair of vertices in a graph and perform its analysis for different inputs. Plot the Graphs
13	a) Implement Bottom up dynamic programming algorithm to solve Knapsack problem and perform its analysis for different inputs. Plot the Graph. b) Implement Memory Function algorithm to solve Knapsack problem and perform its analysis for different inputs. Plot the Graph.
14	Implement Prim's algorithm to find Minimum Spanning Tree of agraph and perform its analysis for different inputs. Plot the Graphs
15	Implement Dijkstra's algorithm to find shortest paths to other vertices in a graph and perform its analysis for different inputs. Plot the Graphs

Implement Bottom up dynamic programming algorithm to