Course Curriculam

Course Code:	CSE303	**Credit Units**	
Course Level	UG		
Course Title	Analysis and Design of Algorithms		
Course Description :			

L	T	P/S	SW	AS/DS	FW	No. of PSDA	Total Credit Unit
3	0	2	2	0	0	0	5

Course Objectives :

SN.	Objectives
1	The designing of algorithm is an important component of computer science. The objective of this course is to make students aware of various techniques used to evaluate the efficiency of a particular algorithm. Students eventually should learn to design efficient algorithm for a particular program.

Pre-Requisites : General

SN.	Course Code	Course Name

Course Contents / Syllabus :

SN.	Module	Descriptors / Topics	Weightage
1	Module I :	Algorithm Design paradigms - motivation, concept of algorithmic efficiency, run time analysis of algorithms, Asymptotic Notations. Recurrences- substitution method, recursion tree method, master method	20.00
2	Module II: Divide and Conquer	Structure of divide-and-conquer algorithms: examples; Binary search, quick sort, Merge sort, Strassen matrix multiplication; Run time analysis of divide and conquer and recurrence relations. Greedy Method: Overview of the greedy paradigm examples of exact optimization solution (minimum cost spanning tree), approximate solution (Knapsack problem), Single source shortest paths problems, traveling salesman problem	20.00
3	Module III : Dynamic Programming	Overview, difference between dynamic programming and divide and conquer technique, Applications: Shortest path in graph, chain matrix multiplication, Traveling salesman Problem, longest Common sequence problem, knapsack problem	20.00
4	Module IV :	Overview, Representation of graphs, strongly connected components, Traversal methods (depth first and breadth first search) and its analysis Back tracking: Overview, 8-queen problem, and Knapsack problem Brach and bound: LC searching Bounding, FIFO branch and bound, LC branch and bound application: 0/1 Knapsack problem, Traveling Salesman Problem	25.00
5	Module V: Computation Complexity	: Complexity measures, Polynomial Vs non-polynomial time complexity; NP-hard and NP-complete classes, examples.	15.00

Course Learning Outcomes :

SN.	Course Learning Outcomes
1	• Ability to apply knowledge of mathematics, science, engineering and computing appropriate to the discipline.
2	• Ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
3	• Ability to design, implement, and evaluate a computer-based system, process, component, or programme to meet desired needs
4	• Ability to use current techniques, skills, and tools necessary for computing practice.

Pedagogy for Course Delivery :

SN.	Pedagogy Methods
1	The class will be held through remote teaching and learning methodology. Instructional material will be uploaded on LMS using four quadrant approach. In addition to giving the assignments and tutorials, the course instructor will spend considerable time in transforming theoretical concepts in practical oriented approach. The lab programs will be solve by using C/C++.

Theory /VAC / Architecture Assessment (L,T & Self Work): 80.00 Max : 100

Attendance+CE+EE : 5+35+60

SN.	Type	Component Name	Marks
1	Attendance		5.00
2	End Term Examination (OMR)		60.00
3	Internal	CLASS TEST	15.00
4	Internal	CLASS QUIZ	10.00
5	Internal	HOME ASSIGNMENT	4.00
6	Internal	VIVA VOCE	3.00
7	Internal	GROUP PRESENTATION	3.00

Lab/ Practical/ Studio/Arch. Studio/ Field Work Assessment : 20.00 Max : 100

Attendance+CE+EE : 5+35+60

SN.	Type	Component Name	Marks
1	Attendance		5.00
2	External	EXPERIMENT	30.00
3	External	VIVA VOCE	30.00
4	Internal	PERFORMANCE	15.00
5	Internal	VIVA VOCE	10.00
6	Internal	PRACTICAL	10.00

Lab/ Practical details, if applicable :

SN.	Lab / Practical Details
1	1. Implement Recursive Binary search and determine the time taken to search an element. Repeat the experiment for different values of n, the number of elements in the list to be searched and plot a graph of the time taken versus n.
2	2. Sort a given set of elements using Quick sort method and determine the time taken 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.
3	3. Implement Knapsack Problem using Greedy Approach.
4	4. Implement 0/1 Knapsack Problem using Dynamic Programming method.
5	5. From a given starting node in a digraph, print all the nodes reachable by using BFS/DFS method.
6	6. Find Minimum Cost Spanning Tree of a given undirected graph using Prim's/Kruskal's algorithm.
7	7. From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.

8	8. Consider the problem of N queen on an (NxN) chessboard. Two queens are said to attack each other if they are on the same row, column, or diagonal. Implements backtracking algorithm to solve the problem i.e. place N non-attacking queens on the board.
9	9. Implement Knapsack Problem based on Backtracking algorithm.
10	10. Implement Traveling Salesman problem based on Branch and Bound technique.
11	1. Develop a Hamiltonian Path in an undirected graph is a path that visits each vertex exactly once. A Hamiltonian cycle (or Hamiltonian circuit) is a Hamiltonian Path such that there is an edge (in graph) from the last vertex to the first vertex of the Hamiltonian Path. Develop a program to implement the solution of Travelling Salesman Problem by considering the Hamiltonian cycle approach.
12	2. Given two sequences, a subsequence is a sequence that appears in the same relative order, but not necessarily contiguous. For example, “abc”, “abg”, “bdf”, “aeg”, ‘”acefg”, .. etc are subsequences of “abcdefg”. So a string of length n has 2^n different possible subsequences. It is a classic computer science problem, the basis of file comparison programs and has applications in bioinformatics. Develop a program to implement the solution of Longest Common Sub-sequence problem.
13	3. A road network can be considered as a graph with positive weights. The nodes represent road junctions and each edge of the graph is associated with a road segment between two junctions. The weight of an edge may correspond to the length of the associated road segment, the time needed to traverse the segment or the cost of traversing the segment. Using directed edges it is also possible to model one-way streets. Such graphs are special in the sense that some edges are more important than other
14	4. There are a number of agents and a number of tasks. Any agent can be assigned to perform any task, incurring some cost that may vary depending on the agent-task assignment. It is required to perform all tasks by assigning exactly one agent to each task and exactly one task to each agent in such a way that the total cost of the assignment is minimized. If the numbers of agents and tasks are equal and the total cost of the assignment for all tasks is equal to the sum of the costs for each agent
15	Suppose that we are designing a program to simulate the storage and search in a dictionary. Words appear with different frequencies, however, and it may be the case that a frequently used word such as "the" appears far from the root while a rarely used word such as "conscientiousness" appears near the root. We want words that occur frequently in the text to be placed nearer to the root. Moreover, there may be words in the dictionary for which there is no definition. Write a program to organiz

List of Professional skill development activities :

No.of PSDA : 3

SN.	PSDA Point
1	Quiz
2	Group Presentation
3	Case Study

Text & References :

SN.	Type	Title/Name	Description	ISBN/ URL
1	Book	1. T. H. Cormen, Leiserson, Rivest and Stein, “Introduction of Computer algorithm,” PHI Publication		
2	Book	2. E. Horowitz, S. Sahni, and S. Rajsekaran, “Funadmentals of Computer Algorithms,” Galgotia Pub		
3	Book	1. Sara Basse, A. V. Gelder, “Computer Algorithms,” Addison Willey Publication		
4	Book	2. J.E Hopcroft, J.D Ullman, “Design and analysis of algorithms” TMH Publication		
5	Book	3. D. E. Knuth, “ The art of Computer Program:, PHI Publication		