Course Title	Course Structure			Pre-Requisite
Algorithm Design and Analysis	L	T	P	Fundamentals of Programming
	3	1	0	

Course Objective:
To introduce the concept of algorithmic efficiency by analyzing various algorithms such as Searching, Sorting, Divide-and-Conquer algorithms and to know details about the Greedy Paradigm, Principle of Dynamic Programming, Back Tracking, Branch and Bound, and Computational Complexity.

Course Outcome (CO):

1. To learn the Algorithm and Design Concepts of linear and non-linear structures and complexity.
2. To understand the concept of searching and sorting
3. To learn concepts of searching and sorting.
4. To learn concepts of the Greedy method.
5. To understand concepts of Dynamic programming.
6. To understand the concepts of Branch and Bound.
7. To understand computational complexity.

S.No.	Content	Contact Hours
Unit 1	**Introduction:** Concept of algorithmic efficiency, run time analysis of algorithms, Asymptotic Notations. Growth of Functions, Master's Theorem.	6
Unit 2	**Searching and Sorting:** Structure of divide-and-conquer algorithms; examples: binary search, quick sort, Stassen Multiplication; merge sort, heap sort, and Analysis of divide and conquer run time recurrence relations.	7
Unit 3	**Greedy Method**: Overview of the greedy paradigm examples of exact optimization solution: minimum cost spanning tree, approximate solutions: Knapsack problem, Kruskal’s algorithm and Prim’s algorithm for finding Minimum cost Spanning Trees, Dijkstra’s and Bellman Ford Algorithm for finding Single source shortest paths, Huffman coding, Activity Selection Problem.	8
Unit 4	**Dynamic programming**: Principles of dynamic programming. Applications: Rod cutting problem, Floyd-Warshall algorithm for all pair shortest paths. Matrix multiplication, Travelling salesman Problem, Longest Common sequence,	7

	Back tracking: Overview, 8-queen problem, and Knapsack problem, Traveling Salesman problem	
Unit 5	**Branch and bound**: LC searching Bounding, FIFO branch and bound, LC branch and bound application: 0/1 Knapsack problem. **Computational Complexity**: Complexity measures, Polynomial Vs non-polynomial time complexity; NP-hard and NP-complete classes, examples: Circuit Satisfiability, Vertex cover, Subset Sum problem, Randomized Algorithms, String Matching, NP-Hard and NP-Completeness, Approximation Algorithms, Sorting Network, Matrix Operations, Polynomials and FFT, Number Theoretic Algorithms.	14
	Total	42

Books:-

S.No.	Name of Books/Authors/Publisher
1	T .H . Cormen, C . E . Leiserson, R. L. Rivest “Introduction to Algorithms”, 3rd Ed., PHI.
2	E. Horowitz, S. Sahni, and S. Rajsekaran, “Fundamentals of Computer Algorithms,” Galgotia Publication, 2008.
3	Sara Basse, A. V. Gelder, “Computer Algorithms,” Addison Wesley, 1999.
4	Aho, Hopcroft, Ullman: The Design and Analysis of Algorithms, Addison Wesley, 1974.