Semester: 4					
Analysis and Design of Algorithms **Category : MAJOR COMPUTER SCIENCE CORE** **(Theory and Lab)**					
Course Code	**:**	**CS2050**	**CIE**	**:**	**70 Marks**
Credits **L:T:P Hours**	**:**	**3** **2:0:2**	**SEE**	**:**	**30 Marks**
Total Hours	**:**	**30L+30P**	**SEE Duration**	**:**	**2:00 Hours**

Course Overview

Algorithm analysis is an important part of computational complexity theory, which provides theoretical estimation for the required resources of an algorithm to solve a specific computational problem. This course provides a comprehensive understanding of algorithmic thinking, emphasizing its definition, significance, and the characteristics of a good algorithm. It covers foundational concepts such as pseudocode, flowcharts, and the systematic steps in algorithm development. Students will delve into analyzing algorithm efficiency using asymptotic notations (Big O, Big Omega, and Big Theta) and explore search algorithms like linear and binary search. The course also focuses on mastering sorting techniques, including bubble sort, selection sort, insertion sort, merge sort, and quick sort, with an emphasis on time complexity and optimization. Advanced design paradigms, such as greedy algorithms (e.g., fractional knapsack, Huffman coding) and dynamic programming, are introduced to tackle complex problems. Essential graph algorithms like minimum spanning trees (Kruskal's and Prim's) and shortest path algorithms (Dijkstra's, Bellman-Ford, and Floyd-Warshall) are also covered, equipping students with critical tools for problem-solving and algorithm design in real-world scenarios.

Unit- 1	2 Hours
Algorithmic Thinking: Definition and Significance Definition and importance of algorithmic thinking, Characteristics of a good algorithm, Steps in algorithm development, Pseudocode and flowcharting.	
Unit-2	**3 Hours**
ComprehensiveAnalysis of Algorithm Efficiency:Asymptotic Notations and Search Algorithms Analyzing time and space complexity, Big O notation, Big Omega, and Big Theta. Best, worst, and average-case analysis, linear search algorithm, binary search algorithm.	

Unit-3	8 Hours
Mastering Algorithms: Analysis and Optimization of Sorting Techniques Importance of sorting, Different types of sorting algorithms, Stability and in-place sorting, Step-by-step explanation Analysis of time complexity (best, worst, and average cases), Optimization techniques of the following sorting algorithms - bubble sort , selection sort , insertion sort, merge sort and quick sort algorithms.	
Unit-4	**8 Hours**
Advanced Algorithm Design Paradigms: Greedy Algorithms and Dynamic Programming Design paradigms: greedy algorithms - greedy choice, fractional knapsack, Huffman coding. Dynamic programming - optimal substructure and overlapping sub-problems.	
Unit-5	**9 Hours**
Essential Graph Algorithms: Minimum Spanning Trees and Shortest Paths Graph Algorithms: minimum spanning tree algorithms - cut property, Kruskal's algorithm using disjoint sets, Prim's algorithm. Shortest path algorithms - Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshall algorithm.	

Course Outcomes: After completing the course, the students will be able to: -	
CO1	Describe basic algorithm design strategies and their applications to solve problems.
CO2	Estimate computational complexity of algorithms.
CO3	Select appropriate design techniques for real-world problem-solving.
CO4	Implement algorithmic design techniques using greedy method.
CO5	Apply dynamic programming approaches for problem-solving.

Textbooks (With ISBN No)

1. Gilles Brassard, Paul Bratley, Pearson, Fundamentals of Algorithmics, 1st Edition, 2015,ISBN: 9789332549999
2. Steven S. Skiena, The Algorithm Design Manual, ISBN: 978-3030542559, 3030542556, 3rd Edition, Springer, 2020, ISBN: 978-3030542559
3. Fundamentals of Computer Algorithms · Ellis Horowitz;Sartaj Sahni;Sanguthevar Rajasekaran · ISBN: 9788173716126
4. Introduction to the Design and Analysis of Algorithms, Anany Levitin:, 2rd Edition, 2009. Pearson.ISBN: 978-9332585485

Reference Material (With ISBN No)

1. Algorithm Efficiency Cases-Course-Swayam-NPTEL
   https://youtu.be/Q0pWMzPAdYM?si=J1AbDi0smb7u37mp
2. Asymptotic Notation-Course-Swayam-NPTEL
   https://youtu.be/msj_2TEzRP0?si=gs4QDzCRAYhQchxW
3. Examples for Calculating Algorithm's Efficiency
   https://youtu.be/xFRea6xyPgM?si=36-DWPDepqyuIYrh
4. DR Stinston, Techniques for Designing and Analyzing Algorithms,2021, ISBN:9780429277412

LABORATORY COMPONENT **30 Hours**

1	Design an algorithm, construct a flowchart,write pseudocode for a program that takes two integers as input and performs the following operation: 1. Addition 2. Subtraction 3. Multiplication 4. Division(handle division by zero)
2	Design an algorithm and write pseudocode for a program that finds the maximum element in a given array of integers.
3	Design an algorithm and write pseudocode for a program that checks whether a given number is prime or not.
4	Implement an algorithm to compute the largest element in an array and analyze its time complexity.
5	Implement a recursive algorithm to compute the nth Fibonacci number and analyze its time complexity.
6	Implement linear search to find an element in an array. Provide a detailed analysis of the time complexity.
7	Write programs to implement Bubble Sort, sort an array of integers and analyze the number of comparisons and swaps performed.
8	Write a program to implement Merge Sort. Analyze and explain the time complexity of the algorithm in terms of the number of comparisons and recursive calls.
9	Write a program to implement Quick Sort. Analyze and explain the time complexity of the algorithm in different cases (best, average, and worst). Count and print the number of comparisons and partitioning steps performed during sorting.
10	Write a program to implement Kruskal's algorithm using disjoint set data structures(union-find) to find the minimum spanning tree of a given weighted undirected graph.