Course Title	**Design & Analysis of Algorithms**		
Course Code	**CSC-311**		
Credit Hours	3 (3,0)		
Category	Computer Science Core		
Prerequisite	None		
Co-Requisite	None		
Follow Up	None		
Course Introduction	This course aims to cultivate students' thinking abilities and proficiency in algorithm design and analysis. It teaches the process of algorithm design, emphasizing efficiency in terms of time and space, providing strategies for optimization, guiding the selection of appropriate data structures, fostering a systematic problem-solving approach, and last but not least, an awareness that not all problems are solvable with the introduction to P vs NP problems.		
Course Learning Outcomes (CLOs)	At the end of the course, the students will be able to:	**BT**	**PLO**
	CLO1: Design efficient algorithms for common computational problems.	C6 (Create)	2,4
	CLO2: Analyze algorithm complexity using Big-O notation.	C4 (Analyze)	2,3
	CLO3: Solve problems involving recursion, divide-and-conquer, and dynamic programming.	C3 (Apply)	3,4
	CLO4: Compare different algorithmic approaches to problem-solving.	C5 (Evaluate)	3
	CLO5: Demonstrate correctness and optimality of algorithms.	C3 (Evaluate)	3,4
Text Book(s)	1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, Introduction to Algorithms, 4th Edition, The MIT Press, 2022, ISBN-10: 026204630X, ISBN-13: 978-0262046305		
Reference Material	1. Algorithm Design, (1st edition, 2013/2014), Jon Kleinberg, Eva Tardos 2. Algorithms, (4th edition, 2011), Robert Sedgewick, Kevin Wayne		

Weekly Lesson Plan

Week No.	Course Content	Assignments/Readings
Week 1	Introduction to Course, Review of Data Structure and some Discrete Mathematics concepts, Role of Algorithms in Computing, Insertion Sort and its line-by-line analysis, and its correctness	Chapter 1, Chapter 2
Week 2	Asymptotic Notations: Big O, Little O, Big Omega, Little Omega, etc. Introduction to the Divide and Conquer Approach, Merge Sort, and its analysis.	Chapter 3 Quiz 1
Week 3	Recurrences and methods to solve recurrence equations: The Substitution Method, Recursion Tree Method, The Master Method, Description of Quick sort, Performance of Quick sort	Chapter 4, Chapter 7 Assignment 1
Week 4	Heaps, Maintaining a Heap Property, Building a Heap, Heap Sort Algorithm, Priority Queues	Chapter 6 Quiz 2
Week 5	Lower Bound on comparison-based sorting, Counting Sort, Radix Sort, Bucket Sort	Chapter 8
Week 6	Minimum and maximum, Selection in expected linear time, Selection in worst-case linear time, Introduction to Dynamic Programming, Matrix Chain Multiplication	Chapter 9, Chapter 14 Quiz 3

Week 7	0-1 Knapsack problem, Coin-change problem, Assembly-line Scheduling	Chapter 14 Assignment 2
Week 8	Longest common subsequence, Optimal binary search trees	Chapter 14 Quiz 4
Week 9	Greedy Algorithm, An activity-selection problem, Elements of Greedy Strategy, Knapsack Problem, Variant of Knapsack problem, Huffman Code	Chapter 15
Week 10	Graph Review, Representation of Graph	Chapter 20 Quiz 5
Week 11	Breadth First Search (BFS), Applications of BFS, Depth First Search (DFS), Applications of DFS	Chapter 20 Assignment 3
Week 12	Topological Sort, Strongly Connected Components, Minimum Spanning Tree, Kruskal Algorithm	Chapter 20, Chapter 21 Quiz 6
Week 13	Prim's Algorithm, Shortest Path Algorithm, Bellman-Ford Algorithm	Chapter 21,
Week 14	Single Source Shortest Path in a directed acyclic graph, Dijkstra's Algorithm, Floyd Warshall Algorithm	Chapter 22, Chapter 23 Quiz 7
Week 15	String Matching with Finite Automata, Knuth-Morris-Pratt Algorithm	Chapter 32 Assignment 4
Week 16	NP-completeness and the classes P and NP, Decision problems vs. optimization problems, NP-completeness and reducibility, NP-complete problems	Chapter 34