DESIGN AND ANALYSIS OF ALGORITHMS

SUB CODE: CE 404 / IT 404

Teaching Scheme (Credits and Hours)

Teaching scheme				Total Credit	Evaluation Scheme					
L	T	P	Total		Theory		Mid Sem Exam	CIA	Pract.	Total
Hrs	Hrs	Hrs	Hrs		Hrs	Marks	Marks	Marks	Marks	Marks
03	00	02	05	04	03	70	30	20	30	150

Learning Objectives:

The educational Objectives of this Course are:

- To Introduce various designing techniques and methods for algorithms
- Performance analysis of Algorithms using asymptotic and empirical approaches
- Demonstrate a familiarity with major algorithms and data structures.
- To give clear idea on algorithmic design paradigms like Divide-and-Conquer, Dynamic Programming, Greedy, Branch and Bound etc.

Outline Of the Course:

Sr. No	Title of the Unit	Minimum Hours
1	**Fundamentals of Algorithms& Mathematics**	05
2	**Analysis of Algorithms**	05
3	**Sorting and searching algorithms**	05
4	**Divide and conquer algorithms**	05
5	**Greedy algorithms**	06
6	**Dynamic programming**	06
7	**Graph Algorithms**	05
8	**String matching**	05
9	**Introduction to Complexity Theory**	03

Total hours (Theory): 45

Total hours (Lab): 30

Total hours: 75

Detailed Syllabus

Sr. No	Topic	Lecture Hours	Weight age(%)
1	**Fundamentals of Algorithms and mathematics** • Problem, algorithm definitions • Mathematics for algorithmic sets • Functions and relations • Combinations • Vectors and matrices • Linear inequalities and linear equations	05	10
2	**Analysis of Algorithms** • Orders of Magnitude (Asymptotic notations) • Growth rates, some common bounds (constant, logarithmic, linear, polynomial, exponential) • Average and worst case analysis • Analysing control statements • Recurrence Relations- substitution, change of variables, master's method	05	10
3	**Sorting and searching algorithms** • Selection sort, bubble sort, insertion sort • Sorting in linear time, count sort • Linear search	05	12
4	**Divide and conquer algorithms** • Introduction • Quick sort, worst and average case complexity • Merge sort • Matrix multiplication • Binary search • Binary search tree	05	12
5	**Greedy algorithms** • General Characteristics of greedy algorithms • Problem solving using Greedy Algorithm- Activity selection problem, Minimum Spanning trees (Kruskal'salgorithm, Prim'salgorithm), Graphs:Shortest paths, The Knapsack Problem	06	12
6	**Dynamic programming** • Introduction • The Principle of Optimality • Problem Solving using Dynamic Programming-Making Change Problem, Assembly Line Scheduling, Knapsack problem, Matrix chain multiplication, Longest Common	06	15

	Subsequence. • Dynamic Programming using Memoization.		
7	**Graph Algorithms:** • An introduction using graphs and games • Traversing Trees– Preconditioning, Depth First Search, Undirected Graph, Directed Graph, Breath First Search, Backtracking– The Knapsack Problem, The Eight queens problem	05	12
8	**String matching** • Introduction • The naive string matching algorithm • The Rabin-Karp algorithm • String Matching with finite automata	05	12
9	**Introduction to Complexity Theory** • The class P and NP • Polynomial reduction • NP- Complete Problems • NP-Hard Problems	03	5
	Total	**45**	**100**

Instructional Method and Pedagogy:

- At the start of course, the course delivery pattern, prerequisite of the subject will be discussed.
- Lectures will be conducted with the aid of multi-media projector, black board, OHP etc.
- Attendance is compulsory in lecture and laboratory which carries 10 marks in overall evaluation.
- One internal exam will be conducted as a part of internal theory evaluation.
- Assignments based on the course content will be given to the students for each unit and will be evaluated at regular interval evaluation.
- Surprise tests/Quizzes/Seminar/tutorial will be conducted having a share of five marks in the overall internal evaluation.
- The course includes a laboratory, where students have an opportunity to build an appreciation for the concepts being taught in lectures.
- Experiments shall be performed in the laboratory related to course contents.

Learning Outcome:

On successful completion of the course, the student will:

- Be able to check the correctness of algorithms using inductive proofs and loop invariants.

- Be able to compare functions using asymptotic analysis and describe the relative merits of worst-, average-, and best-case analysis.
- Be able to solve recurrences using the master, the iteration, and the substitution method.
- Become familiar with a variety of sorting algorithms and their performance characteristics (eg, running time, stability, space usage) and be able to choose the best one under a variety of requirements.
- Be able to understand and identify the performance characteristics of fundamental algorithms and data structures and be able to trace their operations for problems such as sorting, searching, selection, operations on numbers, polynomials and matrices, and graphs.
- Explain the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate. Synthesize new graph algorithms and algorithms that employ graph computations as key components, and analyze them.
- Be able to use the design techniques introduced i.e. dynamic programming, greedy algorithm etc. to design algorithms for more complex problems and analyze their performance.
- Become familiar with the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate.

Reference Books:

1. IntroductiontoAlgorithms,ThomasH.Cormen,CharlesE.Leiserson,RonaldL.Rivest and CliffordStein, PHI.
2. Fundamental of Algorithms by Gills Brassard, Paul Bratley, PHI.
3. Design and Analysis of Computer Algorithms by Aho, Hopcroft and Ullman ,Pearson
4. The Algorithm Design Manual By Steve s. Skiena

List of experiments:

Sr. No	Name of Experiment
1	**Basics:** Find out Big - Oh and Big – Omega of the function. Take necessary data like degree of the function, coefficients, etc… .
2	**Revision of Data Structures**: Write a program to implement: a. A Queue b. A Stack c. A Queue using two Stacks d. A Stack using two Queues
3	**Some Basic Algorithms:** Write an algorithm and find the efficiency of the same for following problems: a. Finding Factorial – Iterative Approach b. Finding Factorial – Recursive Approach c. Printing Fibonacci Series – Iterative Approach

	d. Printing Fibonacci Series – Recursive Approach
4	**Basic Sorting and Searching Techniques:** Design an algorithm and implement a program for: a. Insertion Sort b. Selection Sort c. Bubble Sort d. Count Sort e. Linear Search
5	**Divide and Conquer Approach:** Design an algorithm and implement a program for: a. Merge Sort b. Quick Sort c. Binary Search
6	**Greedy Approach:** Design an algorithm and implement a program to solve: a. Making Change Problem b. Knapsack Problem
7	**Dynamic Programming:** Design an algorithm and implement a program to solve: a. Making Change Problem b. Knapsack Problem c. Finding Optimal Matrix Chain Order Problem
8	**Dynamic Programming:** Design an algorithm and implement a program to solve: a. Longest Common Subsequence Problem b. Finding Optimal Matrix Chain Order Problem using Memoization
9	**Graph Algorithms:** Design an algorithm and write a program to implement: a. Depth First Search of a graph b. Breadth First Search of a graph
10	**Graph Algorithms:**

	Design an algorithm and implement a program for: a. Kruskal’s method of finding Minimum Spanning Tree b. Prim’s method of finding Minimum Spanning Tree c. Dijkstra’s method of finding Single Source Shortest Paths