| Program: B.Sc. - Computer Science | | | | | Semester : IV |
|---|---|---|---|---|---|
| **Course: Fundamentals of Algorithms** | | | | | **Course Code: USMACS401** |
| **Teaching Scheme** | | | | **Evaluation Scheme** | |
| **Lecture (Hours per week)** | **Practical (Hours per week)** | **Tutorial (Hours per week)** | **Credit** | **Continuous Assessment and Evaluation (CAE) (Marks - 25)** | **Term End Examinations (TEE) (Marks-75 in Question Paper)** |
| 02 | 02 | - | 2+1 = 3 | 25 | 75 |

**Learning Objectives:**

- To understand basic principles of algorithm design and why algorithm analysis is important
- To understand how to transform new problems into algorithmic problems with efficient solutions
- To understand algorithm design techniques for solving different problems

**Course Outcomes:**

After completion of the course, learners would be able to:

**CO1:** Examine and evaluate performance of different algorithms

**CO2:** Derive and solve recurrences describing the performance of divide-and-conquer algorithms

**CO3:** Analyze and apply string matching algorithm.

**CO4:** Design optimal solution by applying various algorithm techniques like Dynamic Programming and Greedy Method.

**Outline of Syllabus: (per session plan)**

| Module | Description | No of hours |
|---|---|---|
| 1 | Algorithm Analysis | 10 |
| 2 | Tree, String Matching and Selection Algorithms | 10 |
| 3 | Algorithms Design Techniques | 10 |
| | **Total** | **30** |
| **PRACTICALS** | | **30** |

| Module | Fundamental of Algorithms | No. of Hours/Credits 30/2 |
|---|---|---|
| 1 | **Algorithm Analysis** | 10 |
|  | Introduction to algorithm, Why to analysis algorithm, Running time analysis, How to Compare Algorithms, Rate of Growth, Commonly Used Rates of Growth, Types of Analysis, Asymptotic Notation, Big-O Notation, Omega-Ω Notation, Theta-Θ Notation, Asymptotic Analysis, Properties of Notations, Commonly used Logarithms and Summations, Performance characteristics of algorithms, Master Theorem for Divide and Conquer, Divide and Conquer Master Theorem: Problems & Solutions, Master Theorem for Subtract and Conquer Recurrences |  |
| 2 | **Tree, String Matching and Selection Algorithms** | 10 |
|  | Generic Trees (N-ary Trees), Threaded Binary Tree, Binary Search Trees (BSTs), Balanced Binary Search Trees, AVL (Adelson-Velskii and Landis) Trees, Heapsort<br>String Matching: Introduction, The naive string-matching algorithm, The Rabin-Karp algorithm, String Matching with finite automata, The Knuth-Morris-Pratt algorithm<br>Selection Algorithms: What are Selection Algorithms? Selection by Sorting, Partition-based Selection Algorithm, Linear Selection Algorithm - Median of Medians Algorithm, Finding the K Smallest Elements in Sorted Order |  |
| 3 | **Algorithms Design Techniques** | 10 |
|  | Algorithms Design Techniques: Introduction, Classification, Classification by Implementation Method, Classification by Design Method<br>Greedy Algorithms: Introduction, Greedy Strategy, Elements of Greedy Algorithms, Advantages and Disadvantages of Greedy Method, Greedy Applications, Understanding Greedy Technique<br>Divide and Conquer Algorithms: Introduction, What is Divide and Conquer Strategy? Divide and Conquer Visualization, Understanding Divide and Conquer, Advantages of Divide and Conquer, Disadvantages of Divide and Conquer, Divide and Conquer Applications<br>Dynamic Programming: Introduction, what is Dynamic Programming Strategy? Properties of Dynamic Programming Strategy, Problems which can be solved using Dynamic Programming - Longest Common Subsequence, Dynamic Programming Approaches |  |

**PRACTICALS**

| Sr. No. | Topic. |
|---|---|
| 1 | Write Python program to sort n names using Quick sort algorithm. Discuss the complexity of algorithm used. |
| 2 | Write Python program to sort n numbers using Merge sort algorithm. Discuss the complexity of algorithm used. |
| 3 | Write a Python program to implement Heapsort. |
| 4 | Write a python program to implement Rabin-Karp algorithm. |
| 5 | Write a python program to implement KMP algorithm. |
| 6 | Write Python program for finding the second largest element in an array A of size n using Tournament Method. |
| 7 | Write python program to find kth smallest element using partition-based algorithm. |
| 8 | Write Python program for implementing Huffman Coding Algorithm. Discuss the complexity of algorithm. |

**RECOMMENDED READING:**

**Text Books:**

1. Data Structure and Algorithmic Thinking with Python, Narasimha Karumanchi , CareerMonk Publications, 2016
2. Introduction to Algorithm, Thomas H Cormen, PHI
3. Design and analysis of algorithms, Himanshu Dave, Pearson, 2nd Edition

**Reference Books**

1. Data Structures and Algorithms in Python, Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, 2016, Wiley
2. Fundamentals of Computer Algorithms, Sartaj Sahni and Sanguthevar Rajasekaran Ellis Horowitz, Universities Press
3. Introduction to Design and Analysis of Algorithms, Anany Levitin, Pearson