# AMRITA VISHWA VIDYAPEETHAM AMRITA SCHOOL OF COMPUTING, BENGALURU

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING IV SEMESTER AIE (January 2024 – May 2024)**

**22AIE212 – DESIGN AND ANALYSIS OF ALGORITHMS COURSE PLAN** (L-T-P-C: 2-0-3-3)

# Course Objectives:

* This course helps students to impart various design techniques for formulation of algorithm.
* This course helps students to understand basic categories of algorithms.
* This course helps students to understand and apply analysis of space and time complexity of algorithms and understand concept of growth rate.
* This course helps students to deliver standard notations and representations of algorithmic complexity and known complexities.
* This course helps students to comprehend basic complexity classes.
* This course helps students to acquaint with will know tractable and intractable problems and map solutions to it.

# Course Outcomes:

After completing this course student will be able to,

|  |  |
| --- | --- |
| CO1 | Develop skills for analyzing algorithmic strategies. |
| CO2 | Apply appropriate algorithmic technique for a given problem. |
| CO3 | Implement standard algorithms on arrays, strings, trees and graph. |
| CO4 | Analyse the nature of known classes of tractable or intractable problem. |

# CO-PO Mapping

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 | PSO3 |
| CO |
| CO1 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | - | 3 | 3 | 2 | 3 | 3 | 1 | - |
| CO2 | 3 | 3 | 3 | 2 | 3 | 2 | - | - | 3 | 3 | 2 | 3 | 3 | 2 | - |
| CO3 | 3 | 3 | 3 | 3 | 2 | 1 | - | - | 3 | 3 | 3 | 3 | 3 | 3 | - |
| CO4 | 3 | 3 | 3 | 3 | 2 | 1 | - | - | 3 | 3 | 3 | 3 | 2 | 3 | - |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Lec Hrs** | **TOPICS** | **KEYWORDS** | **OBJECTIVE** | **Remarks** | **CO** |
| 1 | Algorithm Analysis | Notion of an Algorithm – Fundamentals of Algorithmic Problem Solving | To introduce the methods of analyzing an algorithm, flow charts and pseudo code. The assumption is part of it is covered  in Data Structures |  | CO1 |
| 2 to 8 | Analyzing Recursion | Fundamentals of the Analysis of Algorithmic Efficiency – Asymptotic Notations and Growth Rate – Empirical Analysis – Recursive and | Deriving and solving recurrence equation & relations. Method to solve them: Recursion tree, |  | CO1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Non-Recursive Templates | Substitution, Master Method |  |  |
| 9 to 12 | Sorting algorithms | Important Problem Types: Bubble sort, Insertion sort, Selection sort, Heap sort, Quick sort, Merge sort, Counting Sort, Bucket Sort, Radix sort | Understanding of sorting algorithms, their analysis and application | Analyze with worst, average, best case, In-place sorting, External memory sorting, stable sorting | CO1, CO2 |
| 13 to  15 | Brute Force | Exhaustive Search and String Matching – Travelling Salesman Problem – knapsack Problem – Assignment Problem | Understand the Brute force approach and its analysis | Must understand the Naive approach | CO1, CO2 |
| 16 to  19 | Divide and conquer Methodolog y | Binary Search – Merge Sort – Quick Sort – Heap Sort – Multiplication of Large Integers [Maximum sub array sum, Min-max] | Understand and apply the divide and conquer strategy from basic recursion, and analyzing it using recurrence analysis | Student must be able to identify if divide and conquer can be applied and design it appropriately | CO2 CO3 |
| 20 to  23 | Greedy Algorithms | Container loading problem – Huffman Trees. Iterative Methods: The Simplex Method – Maximum Flow Problem, Stable Marriage Problem.  [knapsack Problem & Fractional knapsack Problem] | Overview of greedy approach and analysis, demonstrating correctness of solution, proof by contradiction | Must be able to identify if a problem has greedy choice property and apply appropriately | CO2 CO3 |
| 24 to  27 | Dynamic Programmin g | Principle of optimality – Computing a Binomial Coefficient – Floyd‘s algorithm – Multi stage graph – Coin Changing Problem– Optimal Binary Search Trees – Knapsack Problem and Memory functions. | Study of Dynamic Programming technique, designing a dynamic programming solution and derive the basic equations, and analyze running time. | Solve lot of problems to make students develop the ability to design dynamic programming solutions to a new problem | CO2 CO3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 28 to  31 | Backtrackin g | n-Quees problem – Hamiltonian Circuit Problem – Subset Sum Problem | To understand the techniques of backtracking | To make students develop the ability to design  backtracking solutions | CO2 CO3 |
| 32 to  35 | Branch and Bound | LIFO Search and FIFO Search – Assignment Problem – Knapsack Problem – Travelling Salesmen Problem | To understand the techniques of branch and bound for solving problems | Solve lot of problems to make students develop the ability to design branch and bound solutions to a new  problem | CO2 CO3 |
| 36 to  38 | Graph Algorithms | Maximum Matching in Bipartite Graphs, Computing a Binomial Coefficient – Floyd’s Algorithm – Multi Stage  Graph | Graph algorithms and analysis, ability to map a given problem to a graph and apply the right  graph algorithm |  | CO3 |
| 39 to  40 | Measuring Limitations | Lower Bound Arguments  – P, NP: NP-Complete and NP Hard Problems | Introduction to NP Completeness |  | CO4 |
| 41 to  42 | Approximat ion Algorithms | Algorithms for NP-Hard Problems – Travelling Salesman problem – Knapsack Problem  Revisited | To understand the algorithms in terms of approximation |  | CO4 |
| **End Semester Exam** | | | | |  |

**Textbooks / References:**

* + Analysis of Algorithms’, Jeffrey J McConnel Jones and Barlett Publishers Inc.;2ns Revised edition, 2 November 2007.
  + ‘Introduction to the Design and Analysis of Algorithms’, Anany Levin, Third Edition, Person Education, 2012.
  + Algorithms Design and Analysis’, Harsh Bhasin, Oxford University Press, 2016.
  + Michael T Goodrich and Roberto Tamassia. Algorithm Design Foundations -Analysis and Internet Examples. John Wiley and Sons, 2007.

# Evaluation Plan: (70:30)

# PATTERN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Exam Name** | **Max Marks** | **Weightage** | **Tentative Dates & Remarks** |
| Quiz (20 M) | Quiz-1 | 20 | 6 |  |
| Quiz-2 | 20 | 7 |  |
| Quiz-2 | 20 | 7 |  |
| Midterm Exam (20 M) | Mid Term Exam | 50 | 20 |  |
| Assignment (30 M) | Lab Test 1 | 10 | 10 |  |
| Lab Test 2 | 10 | 10 |  |
|  | Lab Assignment\* | 10 | 10 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| End Sem Exam (30 M) | End Sem Part 1 | 10 | 10 | Decide of Topics / abstract submission, with at least 5 literature survey and use of at-least 2 algorithms |
| End Sem Part 2 | 20 | 20 | Methodology including algorithm design and implementation. Analysis for the new algorithm or comparison between two mentioned algorithms |

# \*Lab Assignment Component:

Problem Statement will be provided

Group size – 4 Students (same mini project team) Algorithmic technique – **3 Marks**

Algorithm / Pseudo Code / Program – **5 Marks**

Complexity of the algorithm – **2 Marks**

# [NB: In case of Plagiarism, will be evaluated for 5 Marks]

# \* End Sem Part 1

Problem Finalization and abstract submission and introduction **[NB: No Sorting] – 2 marks**

Related works (Atleast 5 papers) – **4 Marks**

Proposed plan (use of atleast 2 algorithms) (combining or comparing) – **4 Marks**

Group size – **4 Students** (Note: Overlapping topics are not allowed)

# \* End Sem Part 2

Problem Statement, algorithm design and implementation – **3 + 4 + 7 = 14 Marks**

Time complexity of the proposed algorithm or comparison between them along with their presentation and Q&A – **2 + 4 = 6 marks**