Gyanmanjari Innovative University

Course Syllabus
Gyanmanjari Institute of Technology
Semester-5 (B.Tech.)

Subject: Analysis and Design of Algorithm-BETCE15313

Type of course: Professional Core

Prerequisite: Data structures like Stack, Queue, Linked list, Tree, Graph, Hashing, File structures, any structured programming language (like Java or python).

Rationale:

Developing efficient algorithms is crucial in modern computer engineering, as there is a growing demand for applications that optimize time, memory, and energy usage. This course equips learners with the knowledge and skills to understand and evaluate algorithms, ensuring their effectiveness across various applications.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks					Total Marks
				Theory Marks		Practical Marks		CA	
CI	T	P	C	ESE	MSE	V	P	ALA	
4	0	2	5	60	30	10	20	30	150

Legends: CI-ClassRoom Instructions; T – Tutorial; P - Practical; C – Credit; ESE - End Semester Examination; MSE- Mid Semester Examination; V – Viva; CA - Continuous Assessment; ALA- Active Learning Activities.

GYANMANJARI INNOVATIVE UNIVERSITY BHAVNAGAR

Course Content:

Sr. No	Course Content	Hrs.	% Weightage
1	**Algorithms: Basics and Analysis** What is an algorithm?, Algorithm Specification, Analysis of Algorithms: Analyzing control statement, Loop invariant, Asymptotic Notations :Big-Oh notation(O), Omega notation(Ω),Theta notation(Θ), Amortized analysis, Recurrence equations and methods : Substitution method , Master Theorem.	10	15%
2	**Searching, Sorting & Advanced Data Structures** Searching: Linear search and Binary search Sorting Algorithms and analysis: Bubble sort, Selection sort, Insertion sort, Heap sort, Shell sort Sorting in linear time: Bucket sort, Radix sort and Counting sort. **Exploring Graphs:** An introduction using graphs and games, Undirected Graph, Directed Graph, Traversing Graphs, Depth First Search, Breadth First Search, Topological sort, Connected components.	10	20%
3	**Divide and Conquer Algorithm:** Introduction, Multiplying large Integers Problem, Problem Solving using divide and conquer algorithm - Binary Search, Max-Min problem, Sorting (Merge Sort, Quick Sort), Strassen's matrix multiplication, Exponential. **Greedy Algorithm**: Characteristics of greedy algorithms, Make a change problem, Activity Selection Problem, Fractional Knapsack problem, Huffman Coding, Minimum Spanning Trees: Kruskal's algorithm, Prim's algorithm.	15	25%
4	**Dynamic Programming:** The Principle of Optimality, Problem Calculating the Binomial Coefficient, Making Change Problem, Knapsack problem, Assembly line Scheduling Problem, Matrix Chain Multiplication, Longest Common Subsequence, All pair shortest path: Floyd-Warshal's algorithm. **Backtracking and Branch and Bound:** The Eight queen's problem, Knapsack problem, Travelling Salesman problem, Minimax principle.	15	25%
5	**String Matching:** The naive string-matching algorithm, The Rabin-Karp algorithm, The Knuth-Morris-Pratt algorithm, String Matching with finite automata. **NP-Completeness:** The class P and NP, Polynomial Reduction, NP- Completeness Problems, NP-Hard Problems.	10	15%

Continuous Assessment:

Sr. No	Active Learning Activities	Marks
1	**Sorting in Action: A Visual Approach** Students will use online tools like **VisuAlgo, Sorting Algorithm Animations, or Toptal Sorting Visualizer** to observe sorting algorithms in action. They will analyze time complexities, efficiency of algorithms like Bubble Sort, QuickSort, and Merge Sort. A well-structured **report (PDF)** with screenshots, comparisons, and diagrams must be submitted in document, demonstrating their understanding of sorting techniques and uploaded individually on GMIU web portal.	10
2	**Core Logic Exploration: Understanding Algorithms in Applications** Students will identify real-world algorithmic problems such as finding the shortest route in Google Maps, data compression in ZIP files, or network traffic optimization. They must research its working principle, explain the algorithm and analyze how it applies to the given real-world scenario. They must then present their findings in a PDF report individually on the GMIU web portal.	10
3	**Reviewing Modern Algorithmic Approaches in Computing: A Literature Survey** Students will work in teams of 3 to review 5–8 recent research papers from ACM, IEEE, or arXiv on modern algorithmic techniques. They will select a specific area (like searching, optimization, etc.), analyze and compare the algorithms, and write a survey paper highlighting key findings, trends, limitations, and future research directions. The paper should include proper references, structured sections. Final PDF must be uploaded on the GMIU web portal.	10
	Total	30

Suggested Specification table with Marks (Theory):60

Distribution of Theory Marks **(Revised Bloom's Taxonomy)**						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage %	15%	35%	25%	15%	5%	5%

Course Outcome:

After learning the course, the students should be able to:	
CO1	Explain notation of algorithmic complexity and logic of fundamentals of algorithms.
CO2	Identify suitable data structures to solve a problem effectively and efficiently
CO3	Formulate appropriate algorithms for real life problems.
CO4	Apply optimal solution approach for complex problems.
CO5	Analyze pattern matching algorithms and categorize problems as polynomial or non-polynomial.

List of Practical:

Sr. No	Description	Unit No	Hrs.
1.	Implementation and Time analysis of factorial program using iterative and recursive method.	1	2
2.	Implementation and Time analysis of sorting algorithms. Bubble sort, Selection sort, Insertion sort, Merge sort and Quicksort.	2	4
3.	Implementation and Time analysis of linear and binary search algorithm.	2	2
4.	Implementation of Graph and Searching using DFS (Depth First Search).	2	2
5.	Implementation of Graph and Searching using BFS (Breadth First Search).	2	2
6.	Implementation of Max-Heap Sort Algorithm.	2	2
7.	Implementation of a Knapsack problem using Greedy Algorithm.	3	2
8.	Implement prim's algorithm to find Minimum Spanning Tree.	3	2
9.	Implement kruskal's algorithm to find Minimum Spanning Tree	3	2
10.	Implementation of a knapsack problem using Dynamic Programming.	4	2
11.	Implementation of Matrix Chain Multiplication using Dynamic Programming.	4	4
12.	Implement LCS Algorithm using Dynamic Programming.	4	2
13.	Implement a program for String Matching using Knuth-Morris-Pratt Algorithm on a text file content.	5	2
		Total	30

Instructional Method:

The course delivery method will depend upon the requirement of content and the needs of students. The teacher, in addition to conventional teaching methods by black board, may also use any tools such as demonstration, role play, Quiz, brainstorming, MOOCs etc.

The internal evaluation will be done on the basis of continuous evaluation of students in the laboratory and class-room.

Practical examination will be conducted at the end of semester for evaluation of performance of students in the laboratory.

Students will use supplementary resources such as online videos, NPTEL videos, e-courses, Virtual Laboratory

Reference Books:

[1] Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, PHI.

[2] Fundamental of Algorithms by Gills Brassard, Paul Bratley, PHI.

[3] Design and Analysis of Algorithms, Dave and Dave, Pearson

[4] Fundamental of Algorithms, Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran.

[5] Introduction to Design and Analysis of Algorithms, Anany Levitin, Pearson.