Design and Analysis of Algorithms

Course Title: Design and Analysis of Algorithms **Credit: 3**
Course No: CSIT.311 **Number of period per week: 3+3**
Nature of the Course: Theory + Lab
Year: Third, Semester: Fifth
Level: B. Sc. CSIT

1. Course Introduction

This course introduces basic elements of the design and analysis of computer algorithms. Topics include asymptotic notations and analysis, divide and conquer strategy, greedy methods, dynamic programming, basic graph algorithms, NP-completeness, and approximation algorithms. For each topic, beside in-depth coverage, one or more representative problems and their algorithms shall be discussed.

2. Objectives

Upon completion of this course, students will be able to do the following:

- Analyze the asymptotic performance of algorithms.
- Demonstrate a familiarity with major algorithms and data structures.
- Apply important algorithmic design paradigms and methods of analysis.
- Solve simple to moderately difficult algorithmic problems arising in applications.
- Be able to demonstrate the hardness of simple NP-complete problems

3. Specific Objectives and Contents

Specific Objectives	Contents
• Introduce time and space complexity. • Exemplify complexity of some simple algorithms • Discuss asymptotic notations used in algorithm analysis • Understand and solve recurrence relations	**Unit I: Foundation of Algorithm Analysis (4)** 1.1. Algorithm Analysis Introduction: Algorithm and its properties, RAM model, Time and Space Complexity, detailed analysis of factorial algorithm. 1.2. Asymptotic Notations: Big-O, Big-Ω and Big-Θ Notations their Geometrical Interpretation and Examples. 1.3. Recurrences: Recursive Algorithms and Recurrence Relations, Solving Recurrences (Recursion Tree Method, Substitution Method, Application of Masters Theorem)
• Analyze complexity of iterative algorithms. • Understand analysis of single loops, multiple loops and	**Unit II: Iterative Algorithms (4)** 2.1. Basic Algorithms: Algorithm for GCD, Fibonacci Number and analysis of their time and space complexity

nested loops	2.2. Searching Algorithms: Sequential Search and its analysis 2.3. Sorting Algorithms: Bubble, Selection, and Insertion Sort and their Analysis
• Understand components of divide and conquer strategy • Able to write recurrence relations for divide and conquer algorithms • Solve recurrence relations and find time complexity of divide and conquer algorithms • Understand notion of order statistics and solve this problem	**Unit III: Divide and Conquer Algorithms (10)** 3.1. Searching Algorithms: Binary Search, Min-Max Finding their Analysis 3.2. Sorting Algorithms: Merge Sort and Analysis, Quick Sort and Analysis (Best Case, Worst Case and Average Case), Heap Sort (Heapify, Build Heap and Heap Sort Algorithms and their Analysis), Randomized Quick sort and its Analysis 3.3. Order Statistics: Selection in Expected Linear Time, Selection in Worst Case Linear Time and their Analysis.
• Understand notions of optimization problems and optimal solutions. • Explain concepts behind greedy algorithms • Develop the capability of designing and analyzing greedy algorithms • Discuss message compression and Huffman coding	**Unit IV: Greedy Algorithms (4)** 4.1. Optimization Problems and Optimal Solution, Introduction of Greedy Algorithms, Elements of Greedy Strategy. 4.2. Greedy Algorithms: Fractional Knapsack, Job sequencing with Deadlines, Task Scheduling Algorithms able to designd their Time Complexity. 4.3. Huffman Coding: Purpose of Huffman Coding, Prefix Codes, Huffman Coding Algorithm and its Analysis
• Compare greedy strategy, DP strategy, and divide and conquer strategy • Identify problem that are solvable by DP strategy • Develop the capability of designing and analyzing DP algorithms • Compare DP and Memoization	**Unit V: Dynamic Programming (6)** 6.1. Greedy Algorithms vs Dynamic Programming, Recursion vs Dynamic Programming, Elements of DP Strategy 6.2. DP Algorithms: Matrix Chain Multiplication, String Editing, Zero-One Knapsack Problem, Travelling Salesman Problem and their Analysis. 6.3. Memoization Strategy, Dynamic Programming vs Memoization
• Able to provide different representations of graphs and compare them. • Understand graph traversal techniques, develop their algorithms and analyze them • Develop algorithms for generating MST and shortest paths and analyze them	**Unit VI: Graph Algorithms (8)** 6.4. Graph Representation: Adjacency List, Incidence Matrix and their Efficiency Comparison 6.5. Graph Traversal: Breadth First Search, Depth First Search and their Analysis. 6.6. Spanning Trees: Definition of MST, Kruskals Algorithm, Prims Algorithm and their Analysis 6.7. Shortest Path Algorithms: Bellman Ford, Dijkstra, Floyd Warshwall Algorithms and their Analysis.
• Understand concepts and applications of number theory.	**Unit VII: Number Theoretic Algorithms (4)** 7.1. Number Theoretic Notations, GCD, Euclid's and

• Trace different number theoretic algorithms and analyze them. • Understand and solve the problem of primility testing	Extended Euclid's Algorithms and their Analysis. 7.2. Definition of x modulo n, Solving Modular Linear Equations, Chinese Remainder Theorem 7.3. Primility Testing: Miller-Rabin Randomized Primility Test
• Able to classify problems among different classes. • Understand the concept of problem reduction and polynomial & super polynomial time complexity. • Develop capability of providing proof of NP-completeness • Explain concepts behind approximation algorithms and use them to solve NP complete problems.	**Unit VIII: NP Completeness (5)** 8.1. Tractable and Intractable Problems, Concept of Polynomial Time and Super Polynomial Time Complexity 8.2. Complexity Classes: P, NP, NP-Hard and NP-Complete. NP Complete Problems 8.3. NP Completeness and Reducibility, Cooks Theorem, Proofs of NP Completeness (CNF-SAT, Vertex Cover and Subset Sum) 8.4. Approximation Algorithms: Concept, Vertex Cover Problem, Subset Sum Problem

Evaluation System

Undergraduate Programs							
External Evaluation	**Marks**	**Internal Evaluation**	**Weightage**	**Marks**	**Practical**	**Weightage**	**Mark**
End semester examination	60	Assignments	20%	20	Practical Report copy	25%	20
(Details are given in the separate table at the end)		Quizzes	10%		Viva	25%	
		Attendance	20%		Practical Exam	50%	
		Internal Exams	50%				
Total External	60	Total Internal	100%	20		100%	20
Full Marks 60+20+20 = 100							

External evaluation

1. **End semester examination:**
   It is a written examination at the end of the semester. The questions will be asked covering all the units of the course. The question model, full marks, time and others will be as per the following grid.

2. **External Practical Evaluation:**
   After completing the end semester theoretical examination, practical examination will be held. External examiner will conduct the practical examination according to the above

mentioned evaluation. There will be an internal examiner to assist the external examiner. Three hours time will be given for the practical examination. In this examination Students must demonstrate the knowledge of the subject matter.

Full Marks: 100, Pass Marks: 45, Time: 3 Hrs

Nature of question	Total questions to be asked	Total questions to be answered	Total marks	Weightage
Group A: multiple choice	20	20	20×1 = 20	60%
Group B: Short answer type questions	8	6	6×8 = 48	60%
Group C: Long answer type question/long menu driven programs	3	2	2×16 =32	60%
			100	100%

Each student must secure at least 50% marks in internal evaluation in order to appear in the end semester examination. Failed student will not be eligible to appear in the end semester examinations.

Internal evaluation

Assignment: Each student must submit the assignment individually. The stipulated time for submission of the assignment will be seriously taken.

Quizzes: Unannounced and announced quizzes/tests will be taken by the respective subject teachers. Such quizzes/tests will be conducted twice per semester. The students will be evaluated accordingly.

Attendance in class: Students should regularly attend and participate in class discussion. Eighty percent class attendance is mandatory for the students to enable them to appear in the end semester examination. Below 80% attendance in the class will signify NOT QUALIFIED (NQ) to attend the end semester examination.

Presentation: Students will be divided into groups and each group will be provided with a topic for presentation. It will be evaluated individually as well as group-wise. Individual students have to make presentations on the given topics.

Mid-term examination: It is a written examination and the questions will be asked covering all the topics in the session of the course.

Discussion and participation: Students will be evaluated on the basis of their active participation in the classroom discussions.

Instructional Techniques: All topics are discussed with emphasis on real-world application. List of instructional techniques is as follows:

- Lecture and Discussion

- Group work and Individual work
- Assignments
- Presentation by Students
- Quizzes
- Guest Lecture

Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during that period. If a student fails to attend a formal exam/quiz/test, there won't be any provision for re-exam.

Laboratory Work

Student should write programs, prepare lab sheet for each of the topics discussed in classes. Minimum 3 lab hour per week in required. In laboratory students should perform empirical analysis of different searching and sorting algorithms. Besides this students should implement greedy algorithms, DP algorithms and graph algorithms. Lab sheet of around 15 moderately large programming problems is recommended.

Prescribed Text

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, "*Introduction to algorithms*", Third Edition.. The MIT Press, 2009.

References

- **Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekiaran**, "*Computer Algorithms*", Second Edition, Silicon Press, 2007.
- **Kleinberg, Jon, and Eva Tardos, "** *Algorithm Design"* **,** Addison-Wesley**,** First Edition, 2005