Course: Data Structures and Algorithms (DSA)

Course Code: CS110 Credit:4

Course Description:

The course delves into the fundamental components of efficient and effective programming: data structures and algorithms. It covers essential data structures such as arrays, linked lists, trees, and graphs, highlighting their unique strengths and limitations. The course discusses essential algorithms for tasks such as searching, sorting, and traversing. It also examines techniques to analyze the time and space complexity of algorithms. Throughout the course, there will be extensive implementation of these algorithms in Python, not only enhancing students' programming skills for solving real-world problems but also sharpening their analytical and critical thinking abilities.

Prerequisite(s): Basic Programming Skills in C/C++/ Python

Note: Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Course Objectives:

- Analyze the performance of algorithms
- Introduce fundamental data structures
- Explain the characteristics and trade-offs of different data structures
- Develop proficiency in designing and implementing various types of algorithms
- Apply data structures and algorithms to solve real-world problems
- Promote problem-solving skills: Analytical thinking, logical reasoning, algorithmic design.

Course Outcomes:

- Analyze the time and space complexity of algorithms using Big O, Omega, Theta notations.
- Demonstrate understanding of different data structures and their operations.
- Choose the appropriate data structure for a given problem based on its characteristics and constraints.
- Implement common data structures and algorithms efficiently in chosen programming language (Python).
- Design and develop own algorithms for specific tasks.
- Improve problem-solving skills through algorithmic thinking and reasoning.
- Communicate algorithms and solutions effectively in written and verbal forms.

Approximate weightage of different components in evaluation:

Assignments/Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Policies:

General

1. Computing devices are not to be used during any exams unless instructed to do so.
2. Quizzes and exams are closed books and closed notes.
3. Quizzes are unannounced but they are frequently held after a topic has been covered.
4. ***No makeup quizzes or exams will be given.***

Attendance and Absences

1. Attendance is expected and will be taken each class. Students are not supposed to miss class without prior notice/permission. Any absences may result in point and/or grade deductions.
2. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to get all missing notes or materials.

Textbooks(s):

1. Data Structures and Algorithms in Python (An Indian Adaptation)
   Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser
2. Introduction to Algorithms, 4th edition
   Thomas H. Cormen, Charles E. Leiserson, Ronald Rivest, Clifford Stein
3. Algorithms, 4th Edition
   Robert Sedgewick and Kevin Wayne

Course Outline (tentative) and Syllabus:
The weekly coverage might change as it depends on the progress of the class. Each week assumes 4 hour lectures. Quizzes will be unannounced, so students should maintain close to 100% attendance.

Week	**Content**
Week 1	• Introduction • Algorithm Analysis, Asymptotic Analysis - O, Ω, θ, o, ω
Week 2	• Problem solving session • Recursion, Analyzing Recursive Algorithms, Tower of Hanoi, Recursion types, tail recursion, • Introduction of the Divide and Conquer paradigm of algorithm design: merge sort, binary search, Master theorem
Week 3	• Array-Based Sequences, Python's Sequence Types • Dynamic Arrays and Amortization, Efficiency of Python's Sequence Types • Multidimensional data
Week 4	• Stacks • Queues, and Deques
Week 5	• Quiz 1 • Linked Lists
Week 6	• Trees • Tree Traversal Algorithms
Week 7	• Priority Queues • Heaps
Week 8	• Maps, Hash Tables • Review for Midterm exam
Week 9	• Search Trees - Binary Search, AVL, Splay, and Red-Black Trees
Week 10	• Sorting: Quick sort, Selection, Insertion, Randomized Quick-Select, • Comparision of sorting algorithms
Week 11	• Graph Algorithms - Traversals, Shortest path, Minimum Spanning Trees
Week 12	• Text Processing - Pattern-Matching Algorithms • Class test
Week 13	• Introducing the concept of Dynamic Programming and use of memoization • Greedy Methods
Week 14	• Tries • Union-Find data structure • Quiz 2
Week 15	• Problem Session • Review for Final Exam