Program	ADP Data Science	
Course Code	CC-213	
Course Title	Data Structures	
Credit Hours	**Theory**	**Lab**
	3	1
Lecture Duration	90 minutes (1.5 Hours), 2 lectures per week, 3 hours lab session per week	
Semester	3	
Pre-requisites	**Courses**	**Knowledge**
	Programming Fundamentals	Nil
Follow Up Courses	Operating Systems, Analysis of Algorithms	

Course Learning Outcomes (CLOs):

CLO No	Course Learning Outcome	Bloom Taxonomy
CLO-1	Implement various data structures and their algorithms and apply them in implementing simple applications	C3 (Apply)
CLO-2	Analyze simple algorithms and determine their complexities.	C5 (Analyze)
CLO-3	Apply the knowledge of data structure to other application domains.	C3 (Apply)
CLO-4	Design new data structures and algorithms to solve problems.	C6 (Design)

Aims and Objectives	1. To introduce data structures as basic building blocks of large programs. 2. To learn the commonly used data structures. 3. To introduce the notion of time and space complexity. 4. To develop the skills to analyze time and space requirements for a data structure and associated algorithms. 5. To prepare the students to pick the right data structure for a given problem.
Learning Outcomes	• Implement various data structures and their algorithms and apply them in implementing simple applications • Analyze simple algorithms and determine their complexities. • Apply the knowledge of data structure to other application domains. • Design new data structures and algorithms to solve problems.

Syllabus	Abstract data types, complexity analysis, Big Oh notation, Stacks (linked lists and array implementations), Recursion and analyzing recursive algorithms, divide and conquer algorithms, Sorting algorithms (selection, insertion, merge, quick, bubble, heap, shell, radix, bucket), queue, dequeuer, priority queues (linked and array implementations of queues), linked list & its various types, sorted linked list, searching an unsorted array, binary search for sorted arrays, hashing and indexing, open addressing and chaining, trees and tree traversals, binary search trees, heaps, M-way tress, balanced trees, graphs, breadth-first and depth-first traversal, topological order, shortest path, adjacency matrix and adjacency list implementations, memory management and garbage collection.
Contents	1. Collections, Abstract data types, Complexity analysis, Big Oh notation 2. Recursion and analyzing recursive algorithms, divide and conquer algorithms 3. Sorting algorithms (selection, insertion, merge, quick, bubble, heap, shell, radix, bucket) 4. Array vs. Linked representation of Collections 5. Stacks (linked lists and array implementations)
	6. Queue (linked lists and array implementations), Introduction to priority queues 7. Lists (linked and array implementations), Various types of linked lists, sorted linked list 8. Trees and tree traversals, binary search trees, heaps, M-way tress, balanced trees 9. Heaps and priority queues 10. Graphs, breadth-first and depth-first traversal, topological order, shortest path, adjacency matrix and adjacency list implementations 11. Searching an unsorted array, binary search for sorted arrays, hashing and indexing, open addressing and chaining • Memory management and garbage collection.
Teaching-learning Strategies	• Interactive class session • Hands on practices in class • Brainstorming and Group discussion sessions • Coding in LABS
Assignments	• Paper based written assignments 4 • Coding assignments 6
Textbooks	• Data Structures and Algorithm Analysis in C++ by Mark Allen Weiss

Reference Material/Suggested Readings	• Data Structures and Algorithms in C++ by Adam Drozdek • Data Structures and Algorithm Analysis in Java by Mark A. Weiss Data Structures and Abstractions with Java by Frank M. Carrano & Timothy M. Henry • Java Software Structures: Designing and Using Data Structures by John Lewis and Joseph Chase
Notes	• Academic integrity is expected of all students. Plagiarism or cheating in any assessment will result in at least an F grade in the course, and possibly more severe penalties. You bear all the responsibility for protecting your assignments from plagiarism. If anyone else submits your assignment or uses your code in his/her assignment, you will be considered equally responsible. • The instructor reserves the right to modify the grading scheme/marks division and course outline during the semester. • Introductory knowledge of using the computers is assumed for this course. All code written in quizzes, assignments, homework's, and exams must be in JavaScript. Code must be intelligently documented (commented). Undocumented code may not be given any credit. • The IDE use is not allowed, Notepad++ has to be used for coding. • There is no makeup for a missed sessional grading instruments like quizzes, assignments, and homework's.

Detailed Lecture wise plan

Week	Lecture	Topic	Source Book	Recommendation for Learning Activities
1	1	Introduction to Data Structures; Role of Data Structures in Computer Science Defining Algorithm: Properties of Algorithm		
	2	Introduction to Algorithm's Performance Analysis and Measurement Learning to Calculate Running Time of Different Code Snippets, Examples i.e. Binary search, Selection sort etc;		

2	3	More on Step Counting (Big Oh Notation)		
	4	Case Study: Polynomial as ADT: Take it as sample application to decide its structure and operations and also calculating the step counting of its operations.		
3	5	(Arrays)Matrix, Row major and column major Representation of N-Dimensional Arrays in different Languages.		
	6	Sparse Matrices		
4	7	The Stack ADT, Applications of Stack: Function Call Stack, Usage of Stack in different CS Applications.		
	8	Application of Stack: Expressions Evaluation		
5	9	Queues: Linear/Circular, Applications of Queue.		
	10	Recursive Definition and Processes, Direct Recursion, Learning the Recursive Trace		
6	11	Recursion Continued: Binary Search, Exiting from Maze, Towers of Hanoi and Islamic Fractals as an example		
	12	Recursion Continued:		
7	13	Review of Dynamic Memory Allocation; Object Manipulation of Self Referential objects		
	14	Linear Single Link List Linked Stacks/Queues Linear Double Link List		
8	15	Circular Single Link List, Circular Double Link List Container vs Iterator: Defining Iterator for Link List		
	16	Array-based implementation of Linkbased Structures, Generalized Lists		
Midterm Exam				
9	17	Introduction to Trees, Tree Terminology, Logical construction and Representation of Trees, Introduction to Binary Tree ADT, Mathematical properties Tree Traversals Array-Based Implementation of Binary Trees (Insertion and Traversing)		

Week	Lecture	Topic	Source Book	Recommendation for Learning Activities
	18	Linked Implementation of Binary Trees (Insertion, Traversing, Searching and deletion in Binary Trees)		
10	19	Linked Implementation of Binary Trees Continued:		
	20	Binary Search Tree: Mathematical Properties and its implementation		
11	21	Height Balance Trees: AVL Tree: Insertion in AVL		
	22	Deletion Operation in AVL		
12	23	Heaps (MinHeap and MaxHeap) Heaps as Priority Queues		
	24	Heap continued: (Min-Max Heap, Deaps)		
13	25	Introduction to graph and related terminology Representation of Graphs Elementary Graph Operations, DFS, BFS		
	26	Spanning Trees Connectivity in Graphs		
14	27	Hashing and Overflow Handling		
	28	Hashing continued…		
15	29	Introduction to Sorting types and Techniques, Logical and Algorithmic Implementation of Bubble, Insertion, Selection, Merge, and Quick Sort		
	30	Sorting Continued…		
16	31	Balanced Search Trees: Theoretical Comprehension of Insertion/Deletion Operations in Balanced-Search Trees; 2-3:Tree insertion		
	32	Balanced Search Trees cont…: 2-3 Tree Deletion		
		Final Exam		