Data Structures and Algorithms

Course Title: Data Structures and Algorithms
Course No: CSC206
Nature of the Course: Theory + Lab
Semester: III

Full Marks: 60 + 20 + 20
Pass Marks: 24 + 8 + 8
Credit Hrs: 3

Course Description:
This course includes the basic foundations in of data structures and algorithms. This course covers concepts of various data structures like stack, queue, list, tree and graph. Additionally, the course includes idea of sorting and searching.

Course Objectives:
- To introduce data abstraction and data representation in memory
- To describe, design and use of elementary data structures such as stack, queue, linked list, tree and graph
- To discuss decomposition of complex programming problems into manageable sub-problems
- To introduce algorithms and their complexity

Detail Syllabus:

Unit 1	**Introduction to Data Structures & Algorithms**	**Teaching Hours (4)**
Data types, Data structure and Abstract date type	Concept of data type. Basic and user defined data types. Concept of data structure and its uses. Definition and use of ADT. Benefits of using ADTs.	1 hr
Dynamic memory allocation in C	Concept of dynamic memory allocation.	1 hr
Introduction to Algorithms	Definition of algorithm. What is a good algorithm? Different structures used in algorithms.	1 hr
Asymptotic notations and common functions	Time and space complexity. Big Oh (O) notation.	1 hr
Unit 2	**Stack**	**Teaching Hours (4)**
Basic Concept of Stack, Stack as an ADT, Stack Operations, Stack Applications	Concept and example of stack. Stack ADT. Stack operations. Different applications: Bracket matching.	2 hr
Conversion from infix to postfix/prefix expression, Evaluation of postfix/prefix expressions	Basic definitions and examples of prefix, infix, and postfix expressions. Conversion from one expression to another. Using stack to convert an infix expression to postfix/prefix. Using stack to evaluate postfix/prefix expression	2 hr
Unit 3	**Queue**	**Teaching Hours (4)**
Basic Concept of Queue, Queue as an	Concept of queue. Sequential representation. Queue as and ADT. Different operations.	2 hrs

ADT, Primitive Operations in Queue		
Linear Queue, Circular Queue, Priority Queue, Queue Applications	Circular queue. Linear vs. Circular queue. Priority queue. Queue applications: print server, operating system scheduler etc.	2 hrs
Unit 4	**Recursion**	**Teaching Hours (3)**
Principle of Recursion, Comparison between Recursion and Iteration, Tail Recursion	Recursion definition. Recursive vs. Iterative algorithm. Definition and example of tail recursion	1 hr
Factorial, Fibonacci Sequence, GCD, Tower of Hanoi(TOH)	Example of recursive algorithms (factorial, Fibonacci Sequence, GCD, and Tower of Hanoi algorithms).	2 hrs
Applications and Efficiency of Recursion	Applications of recursion. Finding efficiency of recursive algorithms.	
Unit 5	**Lists**	**Teaching Hours (8)**
Basic Concept, List and ADT, Array Implementation of Lists, Linked List	List concepts. List as an ADT. List operations. Array implementation of lists. Linked list concepts	2 hrs
Types of Linked List: Singly Linked List, Doubly Linked List, Circular Linked List.	Singly and doubly linked lists. Circular linked list.	2 hrs
Basic operations in Linked List: Node Creation, Node Insertion and Deletion from Beginning, End and Specified Position	Inserting and deleting nodes at different positions in a singly and doubly linked list.	3 hrs
Stack and Queue as Linked List	Linked implementation of stack and queue.	1 hr
Unit 6	**Sorting**	**Teaching Hours (8)**
Introduction and Types of sorting: Internal and External sort	Concept of sorting. Internal vs. External sort.	4 hrs
Comparison Sorting Algorithms: Bubble, Selection and Insertion Sort, Shell Sort	Concept of bubble, selection, insertion, and shell sorts	
Divide and Conquer Sorting: Merge, Quick and Heap Sort	Concept of divide and conquer algorithms. Concept of merge, quick, and heap sort	4 hrs
Efficiency of Sorting Algorithms	Efficiency of all above sorting algorithms.	

Unit 7	**Searching and Hashing**	**Teaching Hours (6)**
Introduction to Searching, Search Algorithms: Sequential Search, Binary Search	Concept of searching. Concept of sequential and binary search	3 hrs
Efficiency of Search Algorithms	Efficiency of sequential and binary search algorithms.	
Hashing : Hash Function and Hash Tables, Collision Resolution Techniques	Concept of hashing, hash function, and hash table. Concept of hash collision. Choosing a hash function. Concept of rehashing and chaining. Inserting and deleting items in a hash table.	3 hrs
Unit 8	**Trees and Graphs**	**Teaching Hours (8)**
Concept and Definitions, Basic Operations in Binary Tree, Tree Height, Level and Depth	Concept of binary tree. Strictly binary tree. Level and depth. Complete and almost complete binary tree. Operations on binary trees. Binary tree traversal.	1 hr.
Binary Search Tree, Insertion, Deletion, Traversals, Search in BST	Concept of binary search tree. Insertion, deletion, and search in binary search tree.	2 hr.
AVL tree and Balancing algorithm, Applications of Trees	Concept and example of balanced binary tree. Balancing algorithm. Tree applications: expression tree and game tree.	2 hr.
Definition and Representation of Graphs, Graph Traversal, Minimum Spanning Trees: Kruskal and Prims Algorithm	Concept of graph. Types of graph. Graph representations: adjacency matrix representation and linked representation. Depth-first and breadth-first traversal. Concept of spanning and minimum spanning trees. Kruskal's and Prim's algorithms for finding minimum spanning tree.	3 hr.
Shortest Path Algorithms: Dijksrtra Algorithm	Concept of shortest path. Using Dijkstra's algorithm for finding shortest path.	1 hr

Laboratory Works:

After completing this course, students should have practical knowledge of data structures, algorithms, and ADTs. The laboratory work includes.

- Writing programs with dynamic memory allocation and de-allocation.
- Writing programs to implement stack operations.
- Writing programs using stack to convert infix expression to postfix/prefix expression and to evaluate postfix/prefix expression.
- Writing programs to implement queue operations for linear, circular, and priority queue.
- Writing recursive programs to implement factorial, Fibonacci sequence, GCD, and Tower of Hanoi algorithms.
- Writing programs to implement list using array and linked list.

- Writing programs for linked list implementation of stack and queue.
- Writing programs to implement sorting, searching and hashing algorithms.
- Writing programs to implement Binary Search Trees and AVL Tress.
- Writing programs to implement searching, spanning tree and shortest path.

Text Books:

1. Y Langsam , MJ Augenstein and A.M , Tanenbaum Data Structures using C and C++ , Prentice Hall India, Second Edition 2015

Reference Books:

1. Leen Ammeral, Programmes and Data Structures in C, Wiley Professional Computting
2. G.W Rowe, Introduction to Data Structure and Algroithms with C and C++ , prentice Hall India
3. R.L Kruse, B.P. Leung, C.L. Tondo, Data Structure and Program Design in C Prentice-Hall India

Model Questions

Course Title: Data Structure and Algorithms **Full Marks:** 60
Course No: CSC206 **Pass Marks:** 24
Semester: III **Credit Hrs:** 3

Section A

Attempt any two questions. (2 × 10 = 20)

1. Define stack. How is it different from queue? Write a program to implement circular queue.
2. What is linked list? Explain different types of linked lists. Discuss algorithms for inserting and deleting a node at front position of the linked list.
3. Define graph. Discuss Dijkastra's algorithm for finding shortest path in a graph.

Section B

Attempt any eight questions. (8 × 5 = 40)

4. What is ADT? How is it different from data type? What are benefits of using ADT?
5. How can you use stack to evaluate a postfix expression? Discuss.
6. Define recursive algorithm. How do you implement recursive GCD algorithm?
7. Hand-test selection-sort algorithm with the data given below:
   56, 23, 14, 20, 65, 7, 8, 14, 15, 25
8. Discuss binary search algorithm. What is time complexity of this algorithm?
9. What are benefits of using hashing? How do you choose a hash function?
10. How do you balance a binary tree? Discus.
11. Discuss depth-first search in a graph with example.
12. Write short notes on: (2 × 2.5 = 5)
    a. Big O notation
    b. Spanning tree