DATA STRUCTURE AND ALGORITHMS

ENCT 252

Lecture : 3 **Year : II**
Tutorial : 1 **Part : II**
Practical : 3

Course Objectives:
The objective of this course is to impart fundamental knowledge on the design and implementation of data structures for storing information. It also covers various algorithms used in computer science. Upon completion of this course, students will be able to design and choose the appropriate data structure and efficient algorithm to achieve optimal performance

1 Introduction (4 hours)

- 1.1 Introduction to data structures
  - 1.1.1 Need of data structures
  - 1.1.2 Types of data structures and its characteristics
- 1.2 Abstract data type (ADT)
- 1.3 Basics of algorithm design techniques (Brute Force, divide and conquer, Greedy algorithms, branch and bound, backtracking, randomized, recursive, dynamic programming)
- 1.4 Algorithm analysis
  - 1.4.1 Time and space complexity
  - 1.4.2 Best, worst and average case analysis
  - 1.4.3 Rate of growth
  - 1.4.4 Asymptotic notations: Big Oh, Big Omega and Big Theta

2 Stack and Recursion (7 hours)

- 2.1 Definition of stack and its operations
- 2.2 Array implementation of stack ADT
- 2.3 Stack applications
  - 2.3.1 Expression conversion: Infix to postfix and prefix expression
  - 2.3.2 Expression evaluation: Infix and postfix expression evaluation
- 2.4 Recursion
  - 2.4.1 Concept of recursion
  - 2.4.2 Recursion and stack
  - 2.4.3 Recursion vs iteration
  - 2.4.4 Execution of recursive calls
  - 2.4.5 Types of recursions
  - 2.4.6 Applications of recursion: Tower of Hanoi

3 Queues (5 hours)

3.1 Definition of queue and its operations
3.2 Array implementation of queue ADT
3.3 Types of queue ADT: Linear, circular, double ended and priority queues

4 Linked List (6 hours)

4.1 Definition of list and its operations
4.2 Array implementation of list ADT
4.3 Static list and its limitations
4.4 Linked list: Definition and its operations
4.5 Types of linked list: Singly, doubly, circular
4.6 Application of linked list
  4.6.1 Linked list implementation of stack and queue ADT
  4.6.2 Solving polynomial equations using linked list

5 Tree (7 hours)

5.1 Definition and tree terminologies
5.2 Binary trees
  5.2.1 Definition and types
  5.2.2 Array and linked list representation
  5.2.3 Traversal algorithms: Pre-order, in-order and post-order traversal
  5.2.4 Application of full binary tree: Huffman algorithm
5.3 Binary search tree
  5.3.1 Definition and operations on binary search tree: Insertion, deletion, searching and traversing
  5.3.2 Construction of binary search tree
5.4 Balanced binary tree
  5.4.1 Problem with unbalanced binary trees
  5.4.2 Balanced binary search tree
  5.4.3 AVL tree, definition and need of AVL tree, construction of AVL tree: Insertion, deletion on AVL tree and rotation operations
5.5 Introduction to red-black tree
5.6 B-Tree: Need, definition and construction of B-tree

6 Graphs (6 hours)

6.1 Definition, terminologies and types of graphs
6.2 Representation of graphs: Adjacency matrix, incidence matrix and adjacency list
6.3 Transitive closure and Warshall's algorithm
6.4 Graph traversals: Breadth-first search, depth-first search and topological sort

6.5 Minimum spanning tree: Kruskal's algorithm and prim's algorithm
6.6 Shortest-paths problems: Dijkstra's algorithm, Floyd- Warshall algorithm

7 Sorting Algorithms (5 hours)

7.1 Definition of sorting and its applications
7.2 Types of sorting: Internal/external sort, stable/unstable sort, in-place/ not in-place sort, adaptive/ non-adaptive sort
7.3 Sorting algorithms and its efficiency: Bubble, insertion, selection, shell, quick, merge, radix and heap sorting

8 Searching Algorithms (5 hours)

8.1 Definition of searching techniques and its applications
8.2 Different searching algorithms and its efficiency
  8.2.1 Sequential search
  8.2.2 Binary search
8.3 Hashing
  8.3.1 Definition and its applications
  8.3.2 Hash function
  8.3.3 Hash table
  8.3.4 Collision in hash table
  8.3.5 Collision resolution techniques: Chaining method and open addressing method (Linear probing, quadratic probing and double hashing)

Tutorial (15 hours)

1. Analyzing time and space complexity of basic algorithms and comparing best, worst, and average cases with examples
2. Solving problems using stacks: Converting infix expressions to postfix/prefix, evaluating postfix expressions, balancing parentheses, reversing a string
3. Solving Tower of Hanoi recursively and analyzing tail vs non-tail recursion
4. Solving polynomial addition using linked lists
5. Implementing tree traversals: Preorder, inorder, postorder
6. Constructing a binary search tree (BST) and performing insertion, deletion, and searching, balancing BSTs with AVL tree operations (Rotations, insertion, deletion), constructing B-Trees (Insertion, deletion)
7. Implementing Huffman coding for text compression
8. Solving shortest path problems using Dijkstra's and Floyd-Warshall algorithms
9. Implementing Kruskal's and Prim's algorithms for minimum spanning tree
10. Analyzing and comparing sorting algorithm efficiencies
11. Implementing linear search and binary search
12. Designing a hash table with different collision resolution techniques (Chaining, linear probing, quadratic probing, double hashing)

Practical **(45 hours)**

1. Implementation of stack using array, applications of stack- conversion of infix to prefix and postfix expression, evaluation of prefix and postfix expression, matching parenthesis, reversal of string
2. Implementation of recursive algorithms (Tail and non-tail method)- Factorial, sum of natural numbers, Fibonacci series, implementation of Tower of Hanoi
3. Implementations of linear queue and circular queue using arrays
4. Implementation of static list, implementation of linked list: Singly and doubly linked lists
5. Implementation of stack and queue using linked list, application of linked list- polynomial addition
6. Implementation of in-order, pre-order and post-order tree traversals
7. Implementation of breadth-first and depth-first search to traverse a graph
8. Implementation of different sorting algorithms
9. Implementation of different searching algorithms

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	4	5
2	7	9
3	5	7
4	6	8
5	7	9
6	6	8
7	5	7
8	5	7
Total	**45**	**60**

* There may be minor deviation in marks distribution.

Reference

1. Langsam, Y. Augenstein .M. J. and Tenenbaum A. M. (1996). Data Structures using C and C++. Prentice Hall Press.
2. Rowe, G. W. (1997). Introduction to data structures and algorithms with C++. Prentice-Hall, India.
3. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C, (2022). Introduction to algorithms. MIT press.
4. Kruse, R. L., and Ryba, A. J., (1998). Data structures and program design in C++. Prentice Hall, India.
5. Thareja, R. (2014). Data Structures Using C. Oxford University Press.