QUESTION BANK

DATA STRUCTURES AND ALGORITHMS

Electronics and Communication Department

QUESTION BANK

Subject: Data Structures and Algorithms | Subject Code:BTCS-301-18

Examiner: Dr.Raju Sharma | Semester: 4th

MODULE 1: INTRODUCTION

2 Marks Questions

1. Define **data structure** and its types.
2. What are **asymptotic notations**? List its types.
3. Define **time-space tradeoff**.
4. What are the basic operations on **data structures**?
5. Differentiate between **linear search and binary search**.
6. Define **pointers** and their use in C.
7. What is **dynamic memory allocation**?
8. Define **self-referential structures**.
9. What is **traversal in data structures**?
10. What is the **best case and worst case complexity** of binary search?
11. What is the significance of **Big-O notation**?
12. What is the **difference between static and dynamic memory allocation**?
13. Write the formula for **time complexity of linear search**.
14. Define **garbage collection** in dynamic memory allocation.
15. List any two **advantages of pointers**.

4 Marks Questions

1. Compare **linear search and binary search** in terms of performance.
2. Explain **time-space complexity** with examples.
3. Write an algorithm for **binary search** and analyze its complexity.
4. Explain **asymptotic notations (O, Ω, and Θ)** with examples.
5. Discuss the **advantages and disadvantages of pointers**.
6. Explain **dynamic memory allocation functions** in C with examples.

7. Differentiate between **stack and heap memory**.
8. Write a C program to **demonstrate malloc() and free() functions**.

8 Marks Questions

1. Discuss **asymptotic notations** in detail with graphical representation.
2. Implement **binary search** in C and analyze its complexity.
3. Explain **self-referential structures** and demonstrate with an example program.
4. Explain **pointers and their applications in data structures** with a program.

MODULE 2: STACKS AND QUEUES

2 Marks Questions

1. Define **stack** and its operations.
2. What is a **queue**? How is it different from a stack?
3. Define **LIFO and FIFO** principles.
4. What is a **circular queue**?
5. What is the **use of a stack in recursion**?
6. Define **priority queue**.
7. What is **postfix notation**?
8. What is a **deque (double-ended queue)**?
9. What is the time complexity of **push and pop operations** in a stack?
10. What are the **applications of queues**?
11. Define **expression evaluation** using a stack.
12. What is the maximum number of elements a circular queue can hold?
13. Define **overflow and underflow conditions** in a stack.
14. What is a **linked list-based queue**?
15. What are the advantages of **double-ended queues**?

4 Marks Questions

1. Explain **stack ADT** with its operations.
2. Write an algorithm for **infix to postfix conversion**.

3. Discuss the **applications of stack in expression evaluation**.
4. Explain **circular queue** with an example.
5. Differentiate between **simple queue, circular queue, and priority queue**.
6. Explain **expression evaluation using stacks** with an example.
7. Write a program to **implement stack using an array**.
8. Explain **applications of stacks in recursion**.

8 Marks Questions

1. Implement **stack using an array** and write algorithms for **push, pop, and peek operations**.
2. Explain **expression conversion techniques** (infix, prefix, postfix) with **detailed examples and algorithms**.
3. Explain **priority queues** and write an algorithm for insertion and deletion.
4. Write an algorithm to implement a **queue using linked lists** and analyze its complexity.

MODULE 3: LINKED LISTS AND TREES

2 Marks Questions

1. Define **linked list**.
2. What is a **header node** in a linked list?
3. List any **two advantages of linked lists over arrays**.
4. Differentiate between **singly and doubly linked lists**.
5. Define **AVL tree**.
6. What is a **binary search tree (BST)**?
7. Define **traversal in a tree**.
8. What is a **circular linked list**?
9. Define **balanced tree**.
10. What is the height of an AVL tree with 7 nodes?
11. What is the use of **header nodes in linked lists**?
12. What are **leaf nodes** in a binary tree?

13. Define **preorder traversal**.
14. What is a **degenerate tree**?
15. What is the maximum number of children a binary tree node can have?

4 Marks Questions

1. Explain **linked representation of a stack and queue**.
2. Write an algorithm for **insertion and deletion in a singly linked list**.
3. Explain **binary search tree (BST) insertion** with an example.
4. Compare **binary tree, binary search tree, and AVL tree**.
5. Explain **preorder, inorder, and postorder traversal techniques**.
6. What are the advantages and disadvantages of a **circular linked list**?
7. Write an algorithm to **insert a node in a doubly linked list**.
8. Explain **AVL tree rotations** with an example.

8 Marks Questions

1. Implement a **singly linked list** in C and write functions for insertion, deletion, and traversal.
2. Explain **different types of trees** and discuss their operations.
3. Implement **tree traversal algorithms (preorder, inorder, postorder)** and analyze their complexities.
4. Explain **applications of binary search trees (BSTs) in real-world scenarios**.

MODULE 4: SORTING AND HASHING, GRAPHS

2 Marks Questions

1. Define sorting. Why is sorting important?
2. What are the properties of Bubble Sort?
3. Define time complexity and space complexity in sorting algorithms.
4. What is the worst-case complexity of Quick Sort?
5. What is the key difference between Merge Sort and Quick Sort?
6. What is a stable sorting algorithm? Give an example.

7. What is hashing?
8. Define collision in hashing.
9. What are two methods of resolving collisions in hashing?
10. Define graph in data structures.
11. What is the difference between DFS and BFS traversal techniques?
12. What is an adjacency matrix?
13. What is an adjacency list?
14. What is the time complexity of Heap Sort?
15. Give one real-life application of graph traversal algorithms.

---

4 Marks Questions

1. Explain the working of Bubble Sort with an example.
2. Compare Selection Sort and Insertion Sort in terms of efficiency.
3. Explain Merge Sort with its time complexity.
4. Write an algorithm for Quick Sort and explain its working.
5. Compare Merge Sort and Quick Sort in terms of complexity and performance.
6. What is open addressing and chaining in hashing? Explain with examples.
7. Explain linear probing and quadratic probing in hashing.
8. Describe DFS and BFS algorithms with an example graph.
9. Write an algorithm for BFS traversal of a graph.
10. Write an algorithm for DFS traversal of a graph.
11. Explain insertion and deletion in a heap with examples.
12. Compare Heap Sort and Quick Sort.
13. Discuss graph representations (adjacency matrix and adjacency list).
14. What are directed and undirected graphs? Explain with examples.
15. Explain the Dijkstra algorithm for shortest path with an example.

---

8 Marks Questions

1. Explain all sorting techniques (Bubble, Selection, Insertion, Quick, Merge, Heap Sort) with their complexities.
2. Implement Quick Sort in C and analyze its complexity.
3. Write a C program to implement Merge Sort.
4. What is hashing? Explain different collision resolution techniques with examples.
5. Explain the Heap Sort algorithm and implement it in C.
6. Discuss graph traversal techniques (DFS and BFS) with example graphs and code.
7. Implement graph representation (adjacency matrix and adjacency list) in C.
8. Write a program to perform hashing with linear probing.

Numerical Problems

Problem 1: Quick Sort

Sort the array **{34, 7, 23, 32, 5, 62}** using **Quick Sort**. Show the **step-by-step process** and determine its time complexity.

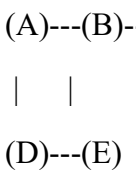
Problem 2: Hashing (Division Method)

Given keys **{27, 43, 34, 15, 51, 19}**, use a **hash table of size 7** and apply the **division method (h(k) = k mod 7)**.

1. Insert all keys into the table.
2. Show collision resolution using **linear probing**.

Problem 3: Graph Traversal

Consider the following graph:



Perform **BFS traversal** starting from **A**.
Perform **DFS traversal** starting from **A**.