

Advanced Stack Theory Questions

1. Explain the stack data structure and its main operations.

- What is LIFO?
- Time complexity of push, pop, peek operations.

2. How can you implement a stack using arrays and linked lists?

- Discuss advantages and disadvantages of each.
- How does dynamic resizing work for an array-based stack?

3. Explain how to implement a stack that supports **getMin()** operation in $O(1)$ time.

- What is the strategy to keep track of minimum elements?
- Explain using an auxiliary stack or modifying the main stack.

4. How does recursion use the system call stack?

- Explain function call stack frames.
- How does recursion depth affect memory?

5. How do you evaluate a postfix (Reverse Polish Notation) expression using a stack?

- Step-by-step algorithm.
- How is it different from infix evaluation?

6. Explain how to check for balanced parentheses using a stack.

- What is the algorithm for matching parentheses, braces, and brackets?
- How do you handle errors or mismatched cases?

7. What is stack overflow and underflow?

- When and why do these occur?
- How can they be prevented or handled?

8. How would you implement multiple stacks in a single array?

- Discuss fixed division vs dynamic division.
- How to handle overflow in one stack without affecting others?

9. Describe how you would reverse a stack using recursion without using any extra data structure.

10. Explain the monotonic stack and its applications.

- How do you use it to solve the “Next Greater Element” problem?
- Other problems solved by monotonic stacks.

11. What is the difference between a stack and a queue? When to use one over the other?

12. Explain how you would implement a stack using two queues.

13. What is the time complexity of operations in a stack implemented with linked lists vs arrays?

14. Explain how a stack is used in Depth-First Search (DFS) on graphs.

15. Describe a real-world scenario where stack data structure is used in software development.

www.tpcglobal.in