DSA Question Bank with Stack and Queue

for Fundamental Problems

Here are **40 questions** combining stack and queue with fundamental problems like prime factorization, anagram checking, reverse number, etc.

Prime Factorization and Number Theory
1. **Prime Factors Using Stack**
   Write a program to find the prime factors of a number using a stack.
   Testcase: Input: `84` → Output: `[2, 2, 3, 7]`

2. **GCD Using Two Stacks**
   Use two stacks to compute the GCD of two numbers. Push the factors of each number onto separate stacks and pop them to find the common highest factor.
   Testcase: Input: `20, 30` → Output: `10`

3. **LCM Using Queue**
   Use a queue to generate multiples of two numbers and find their LCM by identifying the first common multiple.
   Testcase: Input: `4, 6` → Output: `12`

4. **Sum of Digits Using Stack**
   Use a stack to calculate the sum of the digits of a number.
   Testcase: Input: `4321` → Output: `10`

5. **Reverse Digits Using Queue**
   Use a queue to reverse the digits of a number.
   Testcase: Input: `12345` → Output: `54321`

String Manipulations
6. **String Reversal Using Stack**
   Reverse a string using a stack.
   Testcase: Input: `"hello"` → Output: `"olleh"`

7. **String Concatenation Using Queue**
   Use a queue to concatenate two strings character by character.
   Testcase: Input: `"abc"`, `"def"` → Output: `"abcdef"`

8. **String Palindrome Check Using Stack and Queue**
   Check if a string is a palindrome by comparing characters popped from a stack and dequeued from a queue.

Testcase: Input: `"radar"` → Output: `True`

9. **String Method Application Using Queue**
Create a program where each string in a queue undergoes a method (like `toUpperCase()`).
Testcase: Input: `["java", "stack"]` → Output: `["JAVA", "STACK"]`

10. **Find Largest Character Using Stack**
Push all characters of a string onto a stack and find the lexicographically largest character by popping each.
Testcase: Input: `"bacde"` → Output: `"e"`

Number Manipulations
11. **Check Prime Using Stack**
Push all divisors of a number onto a stack and use it to verify if the number is prime.
Testcase: Input: `7` → Output: `True`

12. **Find Factorial Using Stack**
Calculate the factorial of a number using a stack to store intermediate results.
Testcase: Input: `5` → Output: `120`

13. **Queue for Fibonacci Series**
Use a queue to generate the first `n` numbers of the Fibonacci series.
Testcase: Input: `5` → Output: `[0, 1, 1, 2, 3]`

14. **Sum of Digits Using Queue**
Add the digits of a number using a queue.
Testcase: Input: `432` → Output: `9`

15. **Find the Largest Number Using Stack**
Push digits of a number onto a stack and find the largest number by popping elements.
Testcase: Input: `9573` → Output: `9`

Array and List Operations
16. **Array Reversal Using Stack**
Reverse an array using a stack.
Testcase: Input: `[1, 2, 3, 4]` → Output: `[4, 3, 2, 1]`

17. **Access Elements Using Deque**
Access the first and last elements of an ArrayList using a deque.
Testcase: Input: `[10, 20, 30, 40]` → Output: `First: 10, Last: 40`

18. **Sum of Array Elements Using Queue**
Enqueue elements of an array and calculate their sum.

Testcase: Input: `[1, 2, 3]` → Output: `6`

19. **Find Maximum in Array Using Stack**
Use a stack to find the maximum value in an array.
Testcase: Input: `[1, 3, 2]` → Output: `3`

20. **Sort Array Using Stack**
Sort an array using two stacks.
Testcase: Input: `[4, 2, 1, 3]` → Output: `[1, 2, 3, 4]`

Anagram Checking
21. **Anagram Checker Using Queue**
Enqueue characters of two strings into separate queues and check if they are anagrams.
Testcase: Input: `"listen"`, `"silent"` → Output: `True`

22. **Sort Characters for Anagram Using Stack**
Push characters of two strings onto stacks, sort them, and compare to check for anagrams.
Testcase: Input: `"earth"`, `"heart"` → Output: `True`

Stack-Queue Interaction
23. **Stack and Queue for Even-Odd Check**
Push numbers into a stack and enqueue them into a queue. Check if the sum is even or odd.
Testcase: Input: `[2, 3]` → Output: `Odd`

24. **Alternate Push and Enqueue**
Push even numbers into a stack and enqueue odd numbers into a queue.
Testcase: Input: `[1, 2, 3, 4]` → Stack: `[2, 4]`, Queue: `[1, 3]`

Sorting and Searching
25. **Queue for Binary Numbers**
Generate binary numbers up to `n` using a queue.
Testcase: Input: `3` → Output: `["1", "10", "11"]`

26. **Sort Stack Using Recursion**
Sort a stack of integers using recursion.
Testcase: Input: `[3, 1, 4, 2]` → Output: `[1, 2, 3, 4]`

Practical Problems

27. **Infix to Postfix Conversion**
   Convert an infix expression to postfix using a stack.
   Testcase: Input: `"A + B * C"` → Output: `"ABC*+"`

28. **Validate Parentheses**
   Use a stack to check if parentheses in an expression are balanced.
   Testcase: Input: `"(()())"` → Output: `True`

29. **Queue for Recent Calls**
   Implement a queue to track timestamps of recent API calls (last 3000 ms).

30. **Reverse Queue Using Stack**
   Reverse a queue by transferring elements into a stack.
   Testcase: Input: `[1, 2, 3]` → Output: `[3, 2, 1]`

Game Logic
31. **Next Greater Element Using Stack**
   Use a stack to find the next greater element for each element in an array.
   Testcase: Input: `[4, 5, 2, 10]` → Output: `[5, 10, 10, -1]`

32. **Sliding Window Maximum Using Deque**
   Use a deque to find the maximum in each sliding window of size `k`.
   Testcase: Input: `[1, 3, -1, -3, 5, 3, 6, 7]`, `k=3` → Output: `[3, 3, 5, 5, 6, 7]`

33. **Evaluate Postfix Expression**
   Use a stack to evaluate a postfix expression.
   Testcase: Input: `"23*54*+"` → Output: `26`

Advanced String Manipulations
34. **Substring Removal Using Stack**
   Remove all occurrences of a given substring from a string using a stack.
   Testcase: Input: `"abbaca"`, Substring: `"ab"` → Output: `"ca"`

35. **Remove Duplicates Using Stack**
   Remove adjacent duplicates in a string using a stack.
   Testcase: Input: `"abbaca"` → Output: `"ca"`

Miscellaneous
36. **Merge Two Queues**
   Merge two sorted queues into a single sorted queue.
   Testcase: Input: `[1, 3]`, `[2, 4]` → Output: `[1, 2, 3, 4]`

37. **Sort Queue Using Stack**
    Sort a queue using a stack.
    Testcase: Input: `[4, 3, 2, 1]` → Output: `[1, 2, 3, 4]`