Stack and Queue Problem Set

1. Reverse Individual Words Using Stack

Given a string, reverse each word in the string using a stack. For example, if the input is "Hello World", the output should be "olleH dlroW".

Input	Output
"Hello World"	"olleH dlroW"

2. Balanced Parentheses

Given a string containing just the characters `(`, `)`, `{`, `}`, `[` and `]`, determine if the input string is valid.

Input	Output
"()[]{}"	True
"(]"	False

3. Reverse a Stack

Write a function to reverse the elements of a stack using recursion. Do not use any other data structures.

Input	Output
push(1) -> push(2) -> push(3) -> reverse()	3 -> 2 -> 1

4. Queue Implementation Using Two Stacks

Implement a queue using two stacks, supporting the following operations: enqueue(x), dequeue(), isEmpty().

Input	Output

enqueue(1) -> enqueue(2) -> dequeue() -> isEmpty()	1 -> False

5. Queue Reversal

Write a function to reverse a queue using recursion. Do not use any additional data structures.

Input	Output
queue(1) -> enqueue(2) -> enqueue(3) -> reverseQueue()	3 -> 2 -> 1

6. Implement a Circular Queue

Implement a circular queue where elements can be added or removed in a circular manner.

Input	Output
eue(1) -> enqueue(2) -> dequeue() -> enqueue(3) -> isFull()	1 -> 3 -> True

7. Find the Middle Element of a Stack

Design a function to find the middle element of a stack. If there are two middle elements, return the second one.

Input	Output
push(1) -> push(2) -> push(3) -> push(4) -> findMiddle()	3

8. Check for Palindrome Using Stack

Check if a string is a palindrome using a stack. Ignore spaces and case sensitivity.

Input	Output
"A man a plan a canal Panama"	True
"hello"	False

9. Infix to Postfix Conversion

Write a function to convert an infix expression to postfix.

Input	Output
"3 + 5 * 2"	"3 5 2 * +"

10. Postfix Evaluation

Given a postfix expression, evaluate it and return the result.

Input	Output
"3 5 2 * +"	13

11. Implement Stack Using Two Queues

Design a stack using two queues, supporting the following operations: push(x), pop(), peek(), isEmpty().

Input	Output
push(1) -> push(2) -> pop() -> isEmpty()	2 -> False

12. Sort a Stack

Sort a stack in ascending order using only a temporary stack.

Input	Output
push(3) -> push(1) -> push(2) -> sortStack()	1 -> 2 -> 3