Assignment: Stacks & Queues

Q1. Balanced Parentheses

Read a single-line string and print “Balanced” or “Not Balanced” considering (), {}, []. Use a stack.

Q2. Circular Queue (Array)

Implement a circular queue with fixed capacity supporting enqueue, dequeue, and display. Handle wrap-around and detect full/empty.

Q3. Double-Ended Queue (Deque)

Implement a deque supporting insertFront, insertRear, deleteFront, deleteRear, getFront, getRear, and display. Show underflow/overflow when applicable.

Q4. Infix → Postfix Conversion

Read an infix expression (single-letter operands; operators + - * / ^; parentheses). Output the equivalent postfix using a stack with correct precedence.

Q5. Postfix Evaluation

Read a postfix expression of single-digit operands and + - * /. Evaluate and print the integer result using a stack (assume valid input).

Q6. Queue with Two Stacks (Dequeue-Costly)

Build a queue using two stacks where dequeue may move elements. Support enqueue, dequeue, front, and display (FIFO preserved).

Q7. Queue with Two Stacks (Enqueue-Costly)

Build a queue using two stacks where enqueue may move elements. Support enqueue, dequeue, front, and display (FIFO preserved).

Q8. Reverse Individual Words

Given a line with words separated by spaces, reverse each word’s characters while preserving the original word order and spacing. Print the result.

Q9. Reverse Entire String Using Stack

Push all characters of a string onto a stack and pop to print the full reversal.

Q10. Sort Using Stacks

Given an unsorted list of integers, sort it in ascending order (smallest at top) using two stack. Print top → bottom.