**National University of Computer and Emerging Sciences**



**Laboratory Manual**

*for*

# Data Structures Lab

|  |  |
| --- | --- |
| Course Instructor | Mr. Uzair Naqvi |
| Lab Instructor | Mr. Durraiz Waseem |
| Semester | Fall 2024 |

# Department of Computer Science

FAST-NU, Lahore, Pakistan

**Objectives:** Que implementation using linked list

**Problem 1: Implementing a Queue**

Implement a template-based queue using **Linked List**. The required member methods are:

1. **void enqueue()**: Adds an element to queue
2. **void dequeue()**: Removes an element from queue
3. **bool isFull():** return true if queue is full else false.
4. **int size()**: returns the count of total element stored in the stack.
5. **bool isEmpty()**: returns true if the stack is empty else false.
6. **int front()**: returns the element on Front of queue
7. **int rear()**: return the element on Rear of queue

**Problem 2: Implementing a Circular Queue**

Implement a circular queue data structure in C++. The circular queue should support the following operation:

**isPalindrome():** Check if the elements in the circular queue form a palindrome. A palindrome is a sequence that reads the same forwards as backward. You should ignore the wrap-around when checking for palindromes. For example, if the queue contains [1, 2, 3, 2, 1], it is considered a palindrome because [1, 2, 3, 2, 1] reads the same backward.

**Problem 3: Implementing a Min Max Queue**

Design a queue data structure that supports the following operations:

1. **enqueue(item):** Add an item to the back of the queue.
2. **dequeue():** Remove and return the item from the front of the queue.
3. **getMin():** Retrieve the minimum element from the queue in constant time.
4. **getMax**(): Retrieve the maximum element in the queue in constant time.

Ensure that the **getMin** and **getMax** operations provides the elements efficiently without iterating through the entire queue every time.

**Problem 4:** **Reverse a Queue**

Write a function **reverseQue**() to **reverse the elements** of a queue. Hint: You can implement this using a stack.