**National University of Computer and Emerging Sciences**



**Laboratory Manual**

*for*

# Data Structures Lab

|  |  |
| --- | --- |
| Course Instructor | Mr. Uzair Naqvi |
| Lab Instructor(s) | Seemab Ayub, Lubaina Zubair |
| Section | BCS-3H |
| Date | Wed, Sep 27, 2023 |
| Semester | Fall 2023 |

# Department of Computer Science

FAST-NU, Lahore, Pakistan

**Objectives:**

In this lab, students will practice:

1. Doubly Linked List
2. Circular LinkedList
3. Usage of iterators in Linked list

# Question 1

# Task1

Implement a Doubly linked list using template classes which supports the following operations:

a. Insert at start void insertAtHead(T const element);

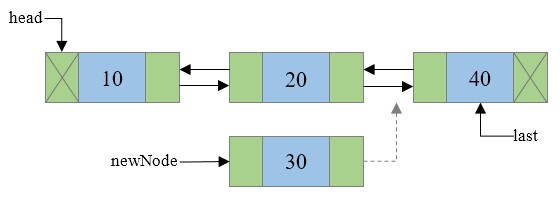
1. Insert at end void insertAtTail (T const element);
2. Print void print() const;
3. Delete node at random position and position should be taken as input from user void deleteRandom ();

# Task2

Make a function insertSorted that takes an element as argument and inserts in doubly linked list in sorted order.

**Example:**

Inserting 30 in the give LL

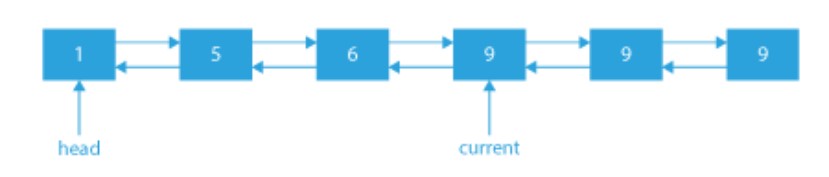


# Task 3

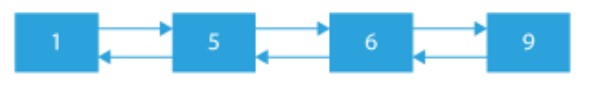
Make a function DeleteDuplicates that deletes the duplicate elements from this sorted list.(Traverse only once)

**Example:**

Input:



Output:



# Task 4

Make a function **swapNodes** in **doubly linked list** that takes a number **‘n’** as argument and swaps the nth node from start with the nth node from end. you are not allowed to swap the data, you have to swap the addresses of these nodes to apply the Swap take care of the edge cases like swapping the first and last value. Maintain the previous pointers as well. (BONUS)

A white rectangular object with black numbers

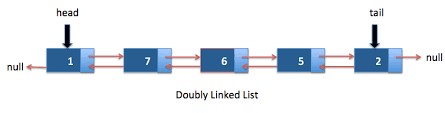
Description automatically generated

# Task 5

Get\_nth\_node\_from\_end(int n): Returns the nth node from end of DLL.

**Example:**

Get\_nth\_node\_from\_end(int 2):



Output: 5

# Task 6

Partition(int n): Rearranges the DLL such that all the values less than the given value come before all values greater than or equal to n.

**Example:**

Input : 3 -> 5 -> 10 -> 2 -> 8 -> 2 -> 1 x = 5

Output : 1-> 2-> 2-> 3-> 5-> 10-> 8

Create a main function to test all the operations

# Question 2

# Task 1

Write a C++ program that splits a circular linked list into two separate circular linked lists, one containing even-positioned nodes and the other containing odd-positioned nodes. Implement an iterator to traverse and display both lists.

**Example:**

Input: Circular linked list with nodes:

1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> NULL

Output:

Even-positioned nodes: 2 -> 4 -> 6 -> 8 -> 2

Odd-positioned nodes: 1 -> 3 -> 5 -> 7 -> 9 -> 1

# Task 2

**Search:** Search a integer taken from user as input

# Task 3

**Insert:** Inserts any given integers at the end of the circular linked list

# Task 4

**Update:** Updates a given integer with another integer on user defined location such as modifying specific elements and displaying the modified list using the iterator.