# University of Central Punjab Faculty of Information Technology

**Data Structures and Algorithms Spring 2024**

|  |  |  |
| --- | --- | --- |
| **Lab 09** | |  |
| **Topic** | * Abstract Classes * Templates * Types of Linked List * Application of Circular Linked List * Application of Doubly Linked List * Application of Doubly Circular Linked List |
| **Objective** | The basic purpose of this lab is to implement types of Linked List and test its Applications |

**Instructions:**

* Indent your code.
* Use meaningful variable names.
* Plan your code carefully on a piece of paper before you implement it.
* Name of the program should be same as the task name. i.e. the first program should be Task\_1.cpp

# void main() is not allowed. Use int main()

* **You have to work in multiple files. i.e separate .h and .cpp files**
* **You are not allowed to use any built-in functions**

# You are required to follow the naming conventions as follow:

* + **Variables:** firstName; (no underscores allowed)
  + **Function:** getName(); (no underscores allowed)
  + **ClassName:** BankAccount (no underscores allowed)

# Students are required to complete the following tasks in lab timings.

**Task 1 Circular Linked List**

* Create Struct **Node** which is having

***Attributes:***

* **Type data;**
* **Node<Type>\* next;**
* Create abstract class named as **LinkedList**

***Attributes of LinkedList:***

* **Node<Type>\* head;**

***Functions:***

* **virtual void convertList(Node<Type>\* head)=0; convert singly linked list into circular linked list**
* **virtual Type printNthFromLast(Node<Type>\* head ,int n)=0; return value at the nth node from the end of the linkedlist**
* **virtual void sortList(Node<Type>\* head)=0; sort a linkedlist of 0s, 1s and 2s**

Create **constructor** and **Destructor** for this class

Now you have to make a derived class named as **myLinkedList** andimplement the above functions

After Implementation of the functions in **myLinkedList** create menu based program to perform the following operations .:

1. **Insert data at end**
2. **Delete data from end**
3. **Insert data at head**
4. **Delete data from head**
5. **Convert singly Linked List into Circular Linked List**
6. **Print nth node From Last**
7. **Sort a linked list of 0s, 1s and 2s**
8. **Exit**

**Task 2 Doubly Linked List**

* Create Struct **Node** which is having

***Attributes:***

* **Type data;**
* **Node<Type>\* next;**
* **Node<Type>\* prev;**
* Create abstract class named as **LinkedList**

***Attributes of LinkedList:***

* **Node<Type>\* head;**

***Functions:***

* **virtual void deleteDuplicateNodes(Node<Type>\* head)=0;**
* **virtual void swapNodes(Node<Type>\* head)=0; swap head node with tail node**

Create **constructor** and **Destructor** for this class

Now you have to make a derived class named as **myLinkedList** andimplement the above functions

After Implementation of the functions in **myLinkedList** create menu based program to perform the following operations .:

1. **Insert data at end**
2. **Delete data from end**
3. **Insert data at head**
4. **Delete data from head**
5. **Delete duplicate nodes**
6. **Swap Head Node with Tail Node**
7. **Exit**

**Task 3 Doubly Circular Linked List**

* Create Struct **Node** which is having

***Attributes:***

* **Type data;**
* **Node<Type>\* next;**
* **Node<Type>\* prev;**
* Create abstract class named as **LinkedList**

***Attributes of LinkedList:***

* **Node<Type>\* head;**

***Functions:***

* **virtual bool checkPalindrome(Node<Type>\* head)=0; check if the linkedlist is palindrome or not**
* **virtual void rotateByNnodes(Node<Type>\* head, int pos)=0; rotate doubly linked list by N nodes**

Create **constructor** and **Destructor** for this class

Now you have to make a derived class named as **myLinkedList** andimplement the above functions

After Implementation of the functions in **myLinkedList** create menu based program to perform the following operations .:

1. **Insert data at end**
2. **Delete data from end**
3. **Insert data at head**
4. **Delete data from head**
5. **Check If the linked list is palindrome or not**
6. **Rotate doubly linked list by N nodes**
7. **Exit**