University of Central Punjab

**Faculty of Information Technology**

# Data Structures and Algorithms

# Fall 2020

|  |  |  |
| --- | --- | --- |
| **Lab 07** | |  |
| **Topic** | * LinkedList |
| **Objective** | * The basic purpose of this lab is to implement ADT of Linked List   and test its applications. |
|  | | |

**Instructions:**

* Indent your code.
* Comment 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 system**("**pause**")
* **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

Modify the code done in class and implement the linked list using **Head** and **Tail** pointers.

**Functions of Linked List:**

**front()** – Returns the value of the first element in the linked list.

**back()** – Returns the value of the last element in the linked list.

**insetAtFront(int)** – Adds a new element at the beginning of the linked list.

**insetAtEnd(int)** – Adds a new element at the end of the linked list**.**

**removeFromFront()** – Removes the first element of the linked list, and reduces size of the linked list by 1.

**RemoveFromEnd()** – Removes the last element of the linked list, and reduces size of the linked list by 1.

**empty()** – Returns whether the list is empty(1) or not(0).

**size()** – Returns the number of elements in the list.

## Task 2

Implement **Stack** (LIFO) **and Queue** (FIFO) using **Linked Lists** only.

It should be a menu driven program which should first ask whether you want to use stack or queue. Then, operations related to stack or queue may be called.

Functionality of Stack:

1. Push // adds on top
2. Pop//removes from top
3. Peek//returns top most element
4. Display()//displays contents of stack

Functionality of Queue:

1. Enqueue
2. Dequeue
3. Display
4. get\_front //displays element present at front end of queue
5. get\_rear // displays element present at rear end of queue

## Task 3

Using the linked list implemented above, create a Search() function and a removeDuplicates() function. Your program should ask the user what action they wish to perform, i.e. search or remove. First insert integer data and create 10 nodes.

The search function will search for an integer number and tells if the number occurs, and in case of occurrence also tells the number of time it occurs.

The remove function will remove all the duplicates in the list. Your remove function should also display the elements left in the list after removing duplicates.

* Your input nodes should have at least one pair of duplicates