University of Computer and Emerging Sciences

Laboratory Manual

for

Data Structures Lab

Course Instructor	Ma'am Arooj Khalil
Lab Instructors	Mr. Sohaib Ahmad —
Section	BSE-3A
Semester	Fall 2022

Department of Computer Science

FAST-NU, Lahore, Pakistan

NOTE

Zero Tolerance for Plagiarism. Penalty will be given in accordance with the severity of plagiarism. This also includes forwarding the case to the DC Committee.

Objective of this lab:

Linked List

Instructions:

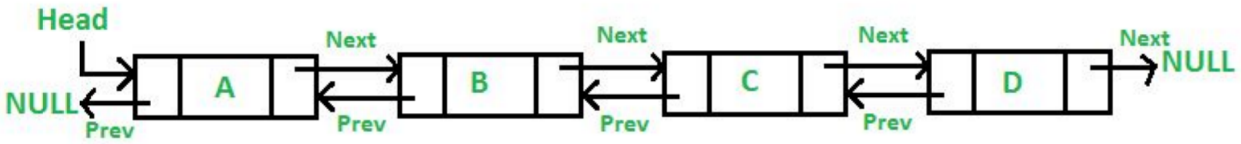
- Make a separate project for each task.
- Indent your code properly.
- Use meaningful variable and function names. Follow the naming conventions.
- Use meaningful prompt lines and labels for all input/output.
- Make sure that there are NO dangling pointers or memory leaks in your program.

Problem:

Implement a class named DoublyLinkList which has the following functionalities

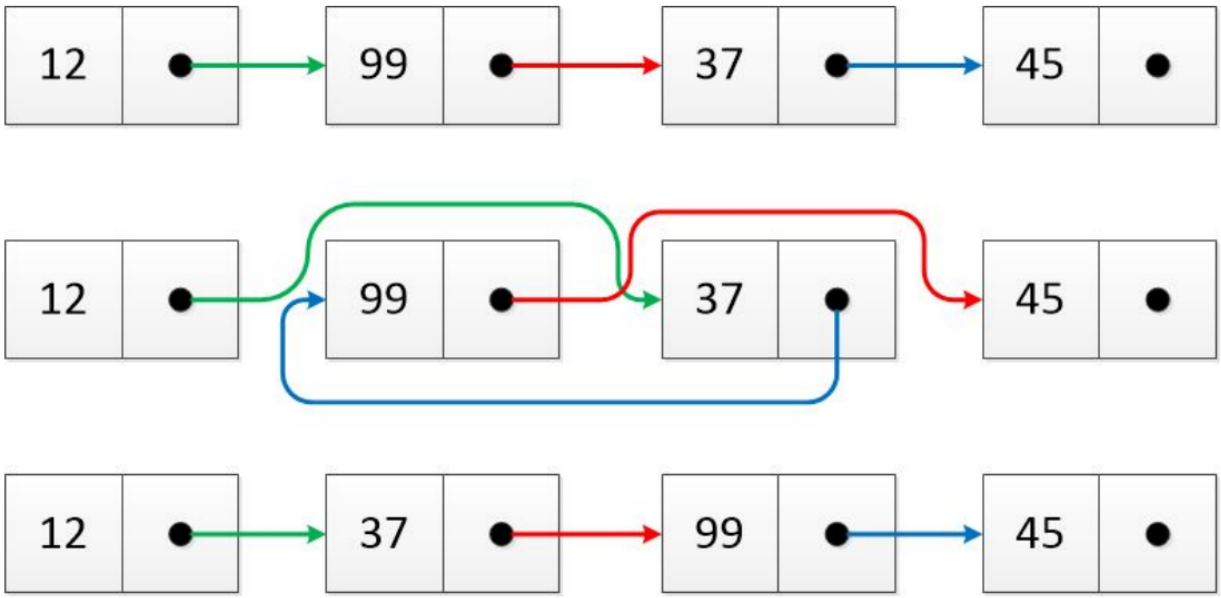
Each node in DoublyLinkList will have two pointers Next and Previous.
Front=> A, Tail => D

NULL<=A<=>B<=>C<=>D=>NULL



1. Implement a template class **'Node'** that contains three data members: A template variable 'data', a Node pointer 'next', and another node pointer 'prev'. You may define any member functions, if required, for this template class.
2. Now implement a **doubly linked list** class having two private data members Node pointer 'head' and Node pointer 'tail'.
3. Now make an '**iterator'** class having one private data member Node pointer current. Please note that iterator class is a nested class of linked list class. (Note that the iterator class is defined inside the List class).
4. Now using the above class, implement a list which supports the following operations:
    a. void insertAtStart(T const element);
    b. void insertAtEnd(T const element);
    c. void printForward() const; This function should print values from head to tail.
    d. void PrintReverse() const; This function should print values from Tail to head
    e. void DeleteFromStart();
    f. void DeleteFromend();
    g. int size() const; This function should return the size of Doubly Linked List.
    h. Node<T>* ReturnMiddle() const; This function should return the middle node of the Doubly Linked List. you are not allowed to use the size of List in any way.
    i. IsEmpty(); Return true if FRONT/TAIL is pointing to NULL otherwise false.
    j. InsertAfter(val, key); It should enter the new Node with the value key, after the first occurance of value val. If not found insert at Tail
    k. InsertBefore(val, key); It should enter the new Node with the value key, before the first occurance of value val. If not found insert at Tail

l. **bool Swap(LeftIndex,RightIndex); Swap the Node on Left index with Node on Right index, 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. Example: Swap between the Nodes on index 1 and 2 is shown in this image. Maintain the previous pointers as well. (BONUS)**



m. Desctructor(); It will delete all the Nodes Independently using a loop, calling each node's destructor.

5. Now create a main function which has the following instructions:
   a. Implement a function bool isPalindrome(char* arr); in your main file with the help of functions created in doubly Linked List.
   b. Insert all char elements one by one in a doubly linked list within the function scope.
   c. Print 'R','E','V','I','V','E','R' is inserted through using both of the list's print functions.
   d. Use deleteFromStart() & deleteFromend() functions to identify if the inserted word is a Palindrome(Or any other function which goes well with your understanding).
   e. Define a char array containing 'R','E','V','I','V','E','R' as a word in main.
   f. Pass your char array in IsPalindrome() function in main and identify if it's palindrome