EECS 560 Lab 2: Doubly Linked list

Due date:
09/11/2020, 11:59 pm -Friday

Objective
Get familiar with ADT (Doubly linked list) implementation with C++.Work on basic operations of Doubly linked list.

Specification of the ADT
Implement the code available in Figure 3.11 and Figure 3.12 of the textbook (Data Structures and Algorithm Analysis in C++ by Mark Weiss, 4th Edition). Please make sure that you are write the missing parts of the code from reference figures (Figure 3.13 to Figure 3.20) as mentioned in the parts of code.

Additional requirement:

1. Rename the object name into **"myDlList",** instead of "List" as indicated in the textbook.

2. Implement **Operator()--** and **Operator(int)--** for Iterator classes which are used in myDlList class for pop_back() and back() methods.

3. Implement an **"void reverselist()"** method, Which reverses the current list and stores the reversed list

4. Implement an "appendList" method, which accepts as parameter another "myDlList" object and appends all items in the parameter to the end of the current object. The constructor should have an interface of **"myDlList<Object> appendList (myDlList<Object>&data)"**

Testing and Grading
We will test your implementation using a tester main function, on a number of instances that are randomly generated. We will release the tester main function, several instances (will be different from the grading instances but will have the same format), and expected output together with the lab instruction via Blackboard. You code will be compiled under Ubuntu 20.04 LTS using g++ version 9.3.0 (default) with C++11 standard.

The command line we are going to use for compiling your code is:
"g++ -std=c++11 main.cpp" (note that main.cpp will try to include the .hpp file you submit, and your .hpp file needs to be property implemented to compile successfully).

Your final score will be the percentage your program passes the grading instances. Note that if your code does not compile (together with our tester main function), you will receive 0. Therefore, it is very important that you ensure your implementation can be successfully compiled before submission.

Submission and Deadline
Please submit your implementation as a single .hpp file, with a file name "myDlList_[YourKUID].hpp". For example, if my KU ID is c123z456, my submission will be a

single file named “myOwnVector_c124z456.hpp”. Submissions that do not comply with the naming specification will not be graded. Please submit through Blackboard.