Homework#4

CS331 Introduction to C++ & Object-Oriented Programming, Summer 2018

Due: Aug 3 (Friday)

Instruction:

- Make one zip file (YourLastName_YourFirstName_CS331_HW4.zip) for your submission, and submit it to Blackboard.
- Your zip file will have Q1, Q2, Q3, and Q4 directories for your complete source code of each question.
- **Your source codes should be properly indented and documented to have professional appearance. You should also provide proper comments for the program and variables.**
- **The evaluation is based on correct implementation and execution.**

Q1. A class `BCheckString` that is derived from the C++ Standard Library `string` class performs bounds checking and throws an exception. The `BCheckString` class will have two member functions:

1) A `BCheckString(string s)` constructor that receives a `string` object passed by value and passes it on to the base class constructor. It just calls the base class constructor using its constructor initialization list.

2) A overloaded operator function, `char operator[ ](int k),` throws an exception, a `BoundsException` object (i.e., `StrExcept()`) if its parameter, `k` is out bounds (i.e., negative or greater than or equal to the length of the string), otherwise it will return the character at position `k` in the string with calling `operator []` in the string base class.

Given: HW4Q1.cpp

To do: Complete HW4Q1.cpp with writing the definition of the `BCheckString` class and `main()`

Test your class with a main function that attempts to access characters that are within and outside the bounds of a suitably initialized `BCheckString` objects

Q2. The two sets of output below show the results of successive circular rotations of a vector. One set of data is for a vector of integers, and the second is for a vector of strings.

```
1 3 5 7
3 5 7 1
5 7 1 3
7 1 3 5

a b c d e
b c d e a
c d e a b
d e a b c
e a b c d
```

Write two template functions that can be used to rotate and output a vector of a generic type:

1) `void rotateLeft (vector <T>& v)`
2) `void output(vector<T> v)`

The `rotateLeft` function performs a single circular left rotation on a vector.
The `output` function print out the vector passed to it as parameter.

Given: HW4Q2.cpp
To do: Complete HW4Q2.cpp with writing `rotateLeft` and `output` functions, and `main()`.
Show your programming is correctly running.

Q3. Write a function **postFixEval** for stack-based evaluation of postfix expressions.
The program reads postfix expressions and prints their values. Each input expression should be entered on its own line, and the program should terminate when the user enters a blank line. Assume that there are only binary operators and that the expressions contain no variables. Your program should use a tack.

Here are sample input-output pairs.

```
78                              78
78  6   +                       84
78  6   +   9  2   -   /         12
```

Given: HW4Q3.cpp.
To do: Complete HW4Q3.cpp.
Show your programming is correctly running.

Q4. The developed application will have a class, **LinkedList** described below. Write LinkedList class. The LinkedList class should be developed as a **template class** that is implemented based on a **generic types "Type"**. The "Type" represents the list elements (ListElement).

Design

class LinkedList	
Data Members	Description
`protected ListElement *head;`	A pointer points the beginning of a singly linked list that stores ListElement type nodes. ListElement class is an inner class of the LinkedList class and the class implementation is already provided in the LinkedList header file.
`protected ListElement *cursor;`	A pointer points a list node where the next list operations will be applied.
Member Functions	Description
`Constructor`	

`Copy Constructor`
`LinkedList(const LinkedList &src);`	

`Destructor`	Deallocates the memory used to store a list.
`void insert(const Type &item, int i);`	When a list is not full, inserts the item into a list. a) If the list is empty, inserts the item as the first element in the list. The head and cursor pointer should be initialized properly after the insert operation. b) If the list is not empty and the second integer argument has the value of 0, inserts the item immediately after the cursor element. c) If the list is not empty and the second integer argument has the value of -1, inserts the item immediately before the cursor element. In all case, properly move the cursor to designate inserted item as the current element.
`void remove();`	a) When a list is not empty, removes the current element from the list. After deleting the cursor pointing element, set the cursor to the following element. b) If the cursor is pointing to an element that is the only element in the list, you should properly set the head and cursor pointer after deleting the node. c) If the cursor is pointing to the last element of the list, the link field of the previous node of the last node should be set to

	null after deleting the node. Set the cursor to the head pointing element.
`Type retrieve() const;`	When a list is not empty, return a copy of the cursor pointing element.
`int  gotoPrior();`	If a list is not empty and the cursor is not pointing the first element of the list, then designates the element immediately before the current element as the current element and returns 1. Otherwise, returns 0.
`int  gotoNext();`	If the current element is not at the end of the list, then designates the element immediately after the current element as the current element and returns 1. Otherwise, returns 0.
`int  gotoBeginning();`	If a list is not empty, then designates the element at the beginning of the list as the current element and returns 1. Otherwise, returns 0.
`void clear();`	Removes all the elements in a list and deallocates associated dynamic memory.
`int  empty() const;`	Returns 1 if a list is empty. Otherwise, returns 0.

Given:

- LinkedList.h . Students should NOT change the given header file.
- HW4Q4.cpp for testing developed program. This driver program allows you testing your code interactively. Read carefully to be familiar with the testing drivers.

To do: Implement LinkedList.cpp.
Show your programming is correctly running with the testing driver program.