HW7
Out: November 11 -- DUE: November 20, Monday 12:00pm
EC327 Introduction to Software Engineering – Fall 2023

Total: 100 points

- *Comment your code and use a clean, easily understandable coding convention. A few coding style guides for your reference:*

http://geosoft.no/development/cppstyle.html
https://google.github.io/styleguide/cppguide.html

Submission:

1. Failure to follow guidelines (filenames, program I/O, and others) WILL result in a significant loss of points on the assignment as we use autograders.
2. Completed assignments **must** be submitted on GradeScope.
3. Submit the required files for each question:
   Stack.cpp StackException.cpp StackException.h Q2.cpp Q3.cpp
   Please make sure your code compiles and runs as intended on the **engineering grid**. Code that does not compile will NOT be graded and will receive a 0.

Late Homework Submissions:

Recall the late submission policy and the fixed penalty of 20% for submitting up to two days after the deadline. Also recall that we only grade your latest submission.

Please write C++ code for solving the following problems.

Q1. Files to submit: Stack.cpp, StackException.cpp, and StackException.h – *Exceptions* **[35 points]**
Implement a new customized exception class, StackException, that inherits from std::exception. Your class should include one method:

```
virtual const char * what() const throw();
// Returns "There are no items on stack!"
```

Put your new exception class implementation and header in **StackException.cpp** and **StackException.h,** respectively. Modify the Stack class given to you so that the pop function throws a StackException object if the stack is empty. You should modify only the **pop** member function in Stack.cpp.

Make sure to write a test program with try-catch to handle this new exception and have it print out using only your customized what() method. You will not be including your test code with your submission. Your test code should look like the following:

```
//create a Stack object of some type
try
{
  //call the pop function for that Stack (when there are no items)
}
catch(StackException & ex)
{
  cout << ex.what(); // prints the error message
}
```

Q2. Files to submit: Q2.cpp – *Templates and linked lists* **[30 points]**
You are provided with a template LinkedList class. Do not submit LinkedList.cpp and LinkedList.h and do not make changes to these files (except for a possible minor change noted in LinkedList.h).

Inherit the LinkedList class to create a new derived class called UniqueList. UniqueList holds **int** values in its elements and has the following additional functionality (as a member function):

```
void insert(int x);
//inserts x at the end of the list only if
//x does not already exist in this list.
```

Your class implementation should work with the provided test code in Q2.cpp. You can expand the test cases as you see fit. Implement all classes and functionality in Q2.cpp.

Q3. Files to submit: Q3.cpp – *Vectors* **[35 points]**

a. Assume an expression is represented in a string (i.e., 36*4+5*4+5). Write a C++ function that splits the expression into numbers and operators. You should consider only + and * as operators. The function takes an expression as the input argument and returns a vector of strings. Each string in the returned vector represents a *positive number* OR an *operand*.

   The header of the function is given as follows:
   ```
   vector<string> split(const string &expression);
   ```

   For example, split("36*4+5*4+5") returns a vector that contains the strings 36, *, 4, +, 5, *, 4, +, 5.

b. Assume the given string always consists of positive numbers, operators, or spaces, and no other characters. Make sure your function works with or without spaces.
   E.g., split("5*7 + 22* 41+1 ") and split("5*7+22*41+1") should give the same result.

c. Write a test program that prompts the user to enter an expression and displays the split items on the console using the split function you implemented. Example:

Enter an expression:	*Please use this exact prompt.*
4 + 50	*User input may or may not include spaces.*
4	*Program output in this example will have three strings in three lines.*
+	
50	

Please implement all classes and functionality in a single file in Q3.cpp.