CS251 - Project 2 - Stacks and Queues

Out: 1/22/2018 9:00 AM
Due: 2/5/2018 11:59 PM

1. Overview

You are required to implement your own ‘stack’ and ‘queue’ classes in C++ without using STL implementations. Implement an expression evaluator using your ‘stack’ class. We are also providing some test cases for each task, which are only representative, your code will be tested on larger files.

2. Detailed Description

Stack

Implement stack operations in mystack.h. In addition, you need to implement stack_main.cpp which performs the following tasks.

Input to stack_main.cpp:
First line of the input is a string belonging to {*int, double, char or string*}, indicates the type of stack. Following lines have one instruction per line. Instruction can be one of {*push x, pop, top, size, empty*}. You need to perform the corresponding operation on the stack and print output accordingly. See the test cases for further understanding.

Queue

Implement queue in myqueue.h. Implement all the methods given in the seed code. In addition, you need to implement queue_main.cpp which performs the following tasks.

Input to queue_main.cpp:
First line of the input is a string belonging to {*int, double, char or string*}, indicates the type of queue to maintain. Following lines have one instruction per line. Instruction can be one of {*enqueue x, dequeue, front, size, empty*}. You need to perform the corresponding operation on the queue and print the output accordingly. See the test cases for further understanding.

Expression Evaluator

Implement class expression_evaluator in expr_eval.h as described below:
The class has one public method ‘double evaluate (string exp, double x)’

Input to the method is a mathematical expression *exp* in the form of a *string* and value of variable *x*. Expression consists of one of the 3 types of characters: numerical constants, variable *x* or mathematical operators. Not necessarily are all 3 types present in all expressions.
The supported mathematical operators should be:

+	-	addition
-	-	subtraction
*	-	multiplication
/	-	division
^	-	power
sin, cos, tan	-	standard trigonometric functions
log	-	logarithm to the base 10
( )	-	parenthesis

Output is a *double* rounded upto two decimal places.

Expression evaluator must be implemented using the algorithm described in the slides. The algorithm uses 2 stacks, one for operators and one for operands.

In addition to expr_eval.h you need to implement expr_eval_main.cpp which takes strings as input (each line has a new expression and value of x if x is present in the expression) and outputs the value of the expression. See the test files for further details.

3. Submission Instructions

The programming assignment must be turned in by the due date and time using the turnin command:

Execute the following turnin command:

turnin -c cs251 -p project2 <*your_folder_name***>**

(Eg: turnin -c cs251 -p project2 john_smith)

(Important: previous submissions are overwritten with the new ones. Your last submission will be the official one and therefore graded).

Verify what you have turned in by typing

turnin -v -c cs251 -p project2

(Important: Do not forget the -v flag, otherwise your submission would be replaced with an empty one).