CMPSC461 Project 2

Deadline: April 22, 2019
NO extensions will be given

Introduction

In this project, you will work as a group (max 2 students). For a given expression, you will build an expression tree and evaluate the result of the expression. The detailed steps are given below.

1) Build an expression tree for a given postfix expression.
2) Compute the value of the expression by doing an in-order traversal of the expression tree

Mandatory Guidelines

1) You will only use C++ functions.

2) You will incorporate the following C++ concepts in your program. a) pure abstract class b) inheritance with virtual functions.

3) You will implement two virtual functions as shown below.

   a. build_expression_tree(args) **(5pts)**
   b. evaluate_expression_tree (args) **(5pts)**

4) You should use ***strtok*** function to parse the given postfix expression into tokens. https://linux.die.net/man/3/strtok

5) You can use built-in functions like string manipulation functions. *You should not use built-in types (C++ STL) for vector, stack, lists and trees.*

Notes:

A postfix expression could of the following form “1 6 * 12 +”

The in-order traversal of the expression tree would print 1 * 6 + 12

The output of this evaluated expression would be “18”

- The postfix expression will contain only integer operands with the following operations (+ - * /). You will follow regular arithmetic precedence for the operations.

- The number of tokens in the input expression is limited to 100. ***Your input expression can have floating point numbers as well***.

- To use pure abstract classes, you can have a parent class which has the two virtual functions. A second class would inherit the parent class and implement the two virtual functions along with other auxiliary functions if needed.

Submission Guidelines:

We have included a tarball which contains the starter files. You will not use any additional files for your project. ***You cannot change the existing code in the files provided as well. You will add additional code for the implementation***. We included a makefile in your project files. You can use this makefile to compile, run, test, and compress before submitting your implementation.

Testing: You have a test file you can test your implementation with, you can test your executable by

```
make test
```

Keep in mind that, we won't be using these files in our tests and you should always test your implementation with additional tests. The out.txt file is for you to check your results against.

Submitting:

- Your output should print the following:

  - `The input is: "Expression"`
  - `The inorder traversal is: "inorder traversal"`
  - `The output of the expression is: "output"`

  Please print the output in the same format. (example shown below).

```
The input is: 1 6 * 8 +
The inorder traversal is: 1 * 6 + 8
The output of the expression is: 14
```

- You need to submit only p2.cpp file for your submission.

Academic Integrity Policy

- Please be aware of the EECS Department's Academic Integrity Policy at http://www.eecs.psu.edu/students/resources/EECS-CSE-Academic-Integrity.aspx.

- Students are required to follow the university guidelines on academic conduct at all times. Students failing to meet these standards will automatically receive a 'F' grade for the course. The instructor carefully monitors for instances of offenses such as plagiarism and illegal collaboration, so it is very important that students use their best possible judgement in meeting this policy. The instructor will not entertain any discussion on the discovery of an offense, and will assign the 'F' grade and refer the student to the appropriate University bodies for possible further action.

- Note that students are explicitly forbidden to copy anything off the Internet (e.g., source code, text) for the purposes of completing an assignment or the final project. Also, students are forbidden from discussing or collaborating on any assignment except were explicitly allowed in writing by the instructor.