CSE4014 – Data Structures and Algorithms I
Lab Assignment I
Due: 21/03/2019

⟨ 2018 - 2019, Spring ⟩

March 7, 2019

(Infix to Postfix) See HW01_template folder. An infixToPostfix class is given in infixToPostfix.h, however the member functions are not defined. Define all the functions by following the instructions given below. You can also define additional functions that you may need to. Along with the header file, a client program (demo.cpp), which converts an infix expression read from infixData.txt file into an equivalent postfix expression is provided. Once you complete the infixToPostfix class, you can run this client program.

Suppose infx represents the infix expression and pfx represents the postfix expression. The rules to convert infx into pfx are as follows:

1. Initialize pfx to an empty expression and also initialize the stack.
2. Get the next symbol, sym, from infx.
   (a) If sym is an operand, append sym to pfx.
   (b) If sym is (, push sym into the stack.
   (c) If sym is), pop and append all of the symbols from the stack until the most recent left parentheses. Pop and discard the left parentheses.
   (d) If sym is an operator:
      i. Pop and append all of the operators from the stack to pfx that are above the most recent left parentheses and have precedence greater than or equal to sym.
      ii. Push sym onto the stack.
3. After processing infx, some operators might be left in the stack. Pop and append to pfx everything from the stack.

In this program, you will consider the following (binary) arithmetic operators: +, -, *, and /. You may assume that the expressions you will process are error free. Design a class that stores the infix and postfix strings. The class must include the following operations:

1. setInfix: Stores the infix expression.
2. showInfix: Outputs the infix expression.
3. showPostfix: Outputs the postfix expression.
4. convertToPostfix: Converts the infix expression into a postfix expression. The resulting postfix expression is stored in pfx.

Test your program on the expressions provided with infixData.txt file. For each expression, your answer must be in the following form:

Infix Expression: A + B - C;

Postfix Expression: A B + C -

Infix Expression: A + B * C - D * E;
Postfix Expression: A B C * + D E * -

Infix Expression: ((A + B) * C - D) * E;
Postfix Expression: A B + C * D - E *

Important Remarks:

1. Working alone is suggested. While grading, if it is realized that you've copied the code from *n-1* fellow students, the grade will be divided to *n*.
2. Your submission has to be a compressed folder named as NameSurname (e.g. BjarneStroustrup.zip) that contains infixToPostfix.h, demo.cpp, and infixData.txt files. If you don't obey these submission rules, your homework will worth 50 points, rather than 100.
3. Add comments to your code. The homeworks without comment lines will also worth 50, rather than 100.