Data Structures HW2

Logistics

Due date: **4/23 (Sun) 23:59**

Submission

- Via LMS (no email submission)
- **Two files**

1. output.txt (answer file)
2. Zip file of your program (**Compress it as a single file.**)

Note: No restrictions on programming languages and platforms.

Problem Description: Infix to Postfix Conversion

Write a program that uses a stack to convert a given infix expression into a postfix expression. The infix expressions are provided in the "input_hw2.txt" file in the following format:

2
3
(A+(B*C))
2
(A+B)

The number in the first line (i.e., 2) shows the number of infix expressions to be processed. From the second line, each prefix expression is given over two lines; the number 3 in the second line shows the number of the operands in the first expression, which is given in the next line (i.e., in (A+(B*C)) there are three operands: A, B, and C.) The operators used in all the expressions are restricted to the following four types: +, -, *, and /. Plus, all the given expressions are enclosed in parentheses, and you may want to refer to the algorithm in P. 41 for handling it.

For all the given infix expressions, write their postfix expressions line by line in "output.txt" (**only one answer per line**), as you did in HW1. For the above example, your "output.txt" must be as follows:

ABC*+
AB+

*****Important Notes*****

Note 1.

DO NOT make any spaces between characters in each postfix expression. If the format is wrong, you will get some penalty. Below are some wrong examples:

A B C * +

A B +

Note 2.

The purpose of this homework is to implement your own stack and apply it to making an application. Without implementing your own stack, you cannot get full scores.

Note 3.

Make sure your answers have no parenthesis. For example, writing (AB+) as the answer for the second case will deduct your score.

Evaluation Policy (10 pts in total)

Score (10pts) = Accuracy (5pts) + Implementation (5pts)

Accuracy: 5 x # right answers / # total cases

Implementation

4pts, if you fully implement a stack from scratch.

2pts, if you use an existing stack library.

1pt, if you don't use a stack.

Penalties

1. Unable to build code → Implementation <= 2
2. **Plagiarism → Score = 0 (will affect your overall grade)**
3. Late Submission → Score -= 2
4. Wrong output format and missing files (in case you forget to submit output.txt, or...) → Accuracy <= 2