CF212 - Data Structures

Homework #2

Due: 10pm, Thursday, January 04, 2018

thangdn - 29/12/2017 Name: _____

Problem 1 (100%) In previous lecture, we have discussed how to evaluate postfix expressions and how to convert an infix expression to a postfix expression. In this problem, you are required to write a program that utilizes a stack.

1. to convert an infix expression to a postfix expression and
2. to evaluate a postfix expression.

Postfix expressions are another type of expression representations without parentheses. Some examples are given in Table 1.

Infix	**Postfix**
2 + 3 * 4	2 3 4 * +
(1 + 2) * 7	1 2 + 7 *
2 + 3 * (4 - 6) + 7	2 3 + 4 6 - 7 + *

Table 1: Some examples of the conversion from an infix expression to a postfix expression

Precedence	**Operators**
1	+ -
2	* / %

Table 2: The precedence of operators used in this problem

Your program should take the input from the standard input device (stdin) - the expression contains only the binary operators +, -, *, /,and % (all integer operations, including divisions) and the operands in the expression are single digit integers. See Table 2 for the precedence of all operators used in this problem. In addition, you will need to consider the case that an expression contain parentheses. Note that you have to directly evaluate the expression from the postfix expression (we will look at your source code). The maximum length of the input string (the infix expression) is 1024 characters and the result of expression evaluation will not be larger or smaller than what can be stored in a signed 32-bit integer.

Hint: this problem can be solved by using a stack.

Please use the following input/output format:

Input format: One single line which contains an infix expression.

Example:

2+3*(4-6)+7

Output format: Two lines. The first line shows the prefix expression converted from the infix expression and the second line shows the result of the expression evaluation.

Example:

2 3 4 6 - * 7 + +
3

You must upload your homework in the format of a compressed zip file to the elearning.thanglong.edu.vn, and the zip file should include the following two files:

1. The source code (.cpp file),
2. A document in PDF format to describe how your program/algorithm works.

Your score of 100% is divided into two parts: correctness (with 4 test cases)(80%) and explanations in the document(20%).

Solution:

Infix Expression: $A + (B * C - (D/E^F) * G) * H$.

Symbol	Scanned	STACK	Postfix Expression	Description
1.		(		Start
2.	A	(	A	
3.	+	(+	A	
4.	(	(+(	A	
5.	B	(+(	AB	
6.	*	(+(*	AB	
7.	C	(+(*	ABC	
8.	-	(+(-	ABC*	'*' is at higher precedence than '-'
9.	(	(+(-(	ABC*	
10.	D	(+(-(	ABC*D	
11.	/	(+(-(/	ABC*D	
12.	E	(+(-(/	ABC*DE	
13.	^	(+(-(/^	ABC*DE	
14.	F	(+(-(/^	ABC*DEF	
15.	)	(+(-	ABC*DEF^/	Pop from top on Stack, that's why '^' Come first
16.	*	(+(-*	ABC*DEF^/	
17.	G	(+(-*	ABC*DEF^/G	
18.	)	(+	ABC*DEF^/G*-	Pop from top on Stack, that's why '^' Come first
19.	*	(+*	ABC*DEF^/G*-	
20.	H	(+*	ABC*DEF^/G*-H	
21.	)	Empty	ABC*DEF^/G*-H*+	END