National University of Computer and Emerging Sciences Islamabad
Data Structures (Fall 2014)

Assignment: 2
Version: 2
Open Date: 18th Sep, 2014
Due Date: 24th Sep, 2014 (5:00PM)
Max Marks: 100
Sections: All

INSTRUCTIONS

- **Submit your codes (one text file containing all the code) on SLATE. No other file is accepted as you all are already aware. A sample file is attached for your assistance.**
- **Failing to submit the code in the required format, you will get a '0'.**
- **Name your text file as Section_RollNumber_Assignment2 (F_13-1234_A2)**
- **Plagiarised code will be awarded an F grade.**
- **Submission through SLATE messages or Email will not be accepted.**
- **Remember there are always marks to follow guidelines.**

Description

The usual notation for writing arithmetic expressions (the notation we learned in elementary school) is called infix notation, in which the operator is written between the operands. For example, in the expression a + b, the operator + is between the operands 'a' and 'b'. In infix notation, the operators have precedence. That is, we must evaluate expressions from left to right, and multiplication and division have higher precedence than addition and subtraction. If we want to evaluate the expression in a different order, we must include parentheses. For example, in the expression a + b * c, we first evaluate * using the operands 'b' and 'c', and then we evaluate + using the operand a and the result of b*c. [1]

In the early 1920s, the Polish mathematician Jan Lukasiewicz discovered that if operators were written before the operands (prefix or Polish notation; for example, + a b), the parentheses can be omitted. In the late 1950s, the Australian philosopher and early computer scientist Charles L. Hamblin proposed a scheme in which the operators follow the operands (postfix operators), resulting in the Reverse Polish notation. This has the advantage that the operators appear in the order required for computation. [1]
For example, the expression: a + b * c in a postfix expression is: a b c * +

Shortly after Lukasiewicz's discovery, it was realized that postfix notation had important applications in computer science. In fact, many compilers use stacks to first translate infix expressions into some form of postfix notation and then translate this postfix expression into machine code. [1]

Task

Your task is to make an INPREPO Calculator. A console based, menu driven application that takes input infix expression and converts it to prefix and postfix expressions. It also evaluates the expression for given values. (If you make GUI base application it will be highly appreciated)

Operators

Possible operators are:

+ - * / $ ( )

Input

Your program shall display a menu ('main' menu) to enter infix expression. There will be two ways to take input:

1. **Variable Expression**
   e.g. a+b+c or ab+bc+c
2. **Numerical Expression**
   e.g. 7+8+6 or 66+77+9

Variable:

1) Length must be greater than 0

2) Only alphabets (abc…) are allowed in variable names

3) Parse input string to get variable names where operators can be used as delimiters between variables names

Examples:

a+b+c ('a', 'b' and 'c' are variables)

ab+bc+c ('ab', 'bc' and 'a' are variables)

wood+coal+water (wood, coal and water are variables)

4) Ask user to input numerical values for all variables.

Numerical:

1) Length must be greater than 0

Examples:

3+4+5 (3, 4, and 5 are operands)

34+44+54 (34, 44, and 54 are operands)

100+1000+10000 (100, 1000, and 10000 are operands)

Output

In case of Variable Expression:

Input Infix Expression: a+b*c

(e.g user enters a=2, b=3 and c=4)

Postfix Expression: abc*+ = 14

Prefix Expression: +*bca = 14

(Your program shall ask user if he/she wants to continue with the same variable expression for different numeric inputs, wants to go back to main menu, or wants to exit program)

In case of Numerical Expression:

Input Infix Expression: 3+4*5

Postfix Expression: 345*+ = 23

Prefix Expression: +*453 = 23

(Your program shall ask user if he/she wants to go back to main menu or wants to exit program)

In case of invalid infix expression, it is required to handle the error properly.

Reference

[1] D. Malik, Data Structures Using C++, ser. Data Structures Series. Cengage Learning, 2009. [Online]. Available: http://books.google.com.pk/books?id=LAU-BKWYdFYC