University of Windsor
School of Computer Science
Fall 2017

CS 60 254 Data Structures and Algorithms

Assignment 2 Due dates (Oct 16 - 18)

For this assignment, you will write a computer program to evaluate arithmetic expressions represented as text (a space-delimited infix expression). For example, the string **"1 + 2 + (3 * 4)"** would evaluate to **15**. Your program should:

1. Read arithmetic expressions in infix format from input text file **(10 points)**
2. Validate the infix expression **(10 points)**
3. Using Stack as ADT convert the infix expression into a postfix expression. **(30 points)**
4. Using Stack as ADT evaluate the postfix expression from step 3 **(30 points)**
5. Display the data in the stack before and after each pop and push operation **(20 points)**

You must handle the unary negation operator. You may use built-in functions in C or Java to handle the power operator (e.g., 2^3 =8) The usual order of operations is in effect:

- Parentheses have higher precedence evaluated left-to-right
- The ^ operator has higher precedence than unary negation operator, and other binary operators (*, /, +, and -)
- The unary negation operator - has higher precedence than the binary operators, and is evaluated right-to-left (right-associative)
- The * and / have higher precedence than + and
- All binary operators are evaluated left-to-right (left-associative)

Your program must read n number of a space-delimited infix expression from an input text file and evaluate them, where each line represents an infix expression.

Examples:

Input:

(25 + 30) * 2 ^ (2 + 1)
30 / 2 + 5 * 2
-3 ^ 2 + 10
(6 + (- 3) - 1)

Output:

440
25
1
2