ÇANKAYA UNIVERSITY

SENG 201 Data and Game Structures

Programming Assignment 3

This assignment you will work on a binary tree (not binary search tree) to represent mathematical formulas and printing the tree.

ALGEBRAIC EXPRESSIONS

An algebraic expression can be represented using three different notations:

1. **Infix:** The most conventional way to write an expression is called infix notation. The arithmetic operators appear in between two operands. e.g. `4-3*2+1`
2. **Prefix:** In prefix notation, as the name suggests, operators come before the operands. e.g. `*-43+21`
3. **Postfix:** In postfix notation, different from infix and prefix notations, operators come after the operands. e.g. `43-21+*`

One use of the binary trees in computer science is to represent algebraic expressions. In an algebraic expression tree, each leaf node contains an operand and a non-leaf node, including the root node, contains an operator that is applied to the results of its left and right sub-trees. For example, the expression below:

`(4-3)*(2+1)`

can be represented as

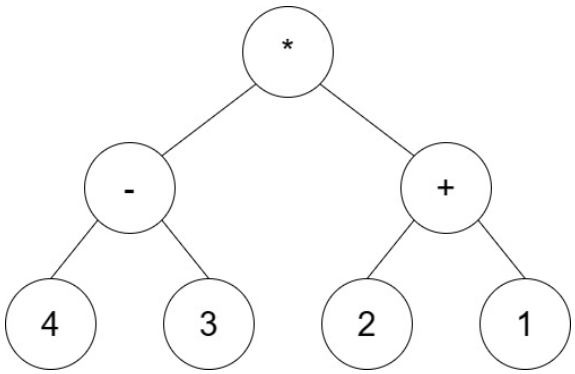

Figure 1 - Binary Tree representation of the mathematical formula: (4-3)(2+1)*

ÇANKAYA UNIVERSITY

PART I – BUILD TREE

In this part you will build an algebraic expression tree using the binary tree data structure. The data structure will represent a single math formula given in **postfix notation** (e.g. the formula in Figure 1 in postfix form is: `43-12+*`)

For simplicity you may assume the following:

- The postfix expression contains single digit integers (0..9) and four operations (`+, -, *, /`)
- In an expression, no whitespace is allowed between characters.

You can use the Node class below to start writing your program:

```
class Node {
	char c;
	Node left, right;

	public Node(char c) {
		this.c = c;
		this.left = this.right = null;
	}
}
```

Write a static method "`buildTree`" that takes a **string formula written in postfix notation**, then builds a binary tree (Figure 1) for that formula and **returns the root node** of the tree.

```
public static Node buildTree( String formulaInPostfix ) { … }
```

ÇANKAYA UNIVERSITY

PART II – PRINT TREE

Write a static method “printTree” which takes the root node of a tree and prints the three as shown in the examples below:

```
public static void printTree( Node root ) { … }
```

Example Tree	Print Result

`Node root = buildTree( "12+34-*" );`
`printTree( root );`
	`*`
	`+ -`
	`1 2 3 4`

`Node root = buildTree( "12+356/-*" );`
`printTree( root );`
	`*`
	`+ -`
	`1 2 3 /`
	`5 6`

`Node root = buildTree( "12+3+578/-*" );`
`printTree( root );`
	`*`
	`+ -`
	`+ 3 5 /`
	`1 2 7 8`

HINTS

- Think about using a stack while reading the formula character by character.
  e.g. `Stack<Node> stack = new Stack<Node>();`
- It will be helpful to keep **level** information and **total nodes count** of the subtree for each node while printing the corresponding node to the correct location. So, you can modify the Node class and add these information
  e.g. add "**`int level, count`**" to the Node class
- To print the tree, create a 2D character array (`char[][]`) that is large enough to store the table output, think of it as a grid, and then put the characters based on the level of the nodes in the tree. Then print the char array row by row.
- You are free to add extra private static helper methods and variables to your code.

ÇANKAYA UNIVERSITY

IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:59 on Sunday, December 31st.

2. You should submit your homework as **a single zip file** including your java files. You should submit to the course webonline page before the deadline.

3. The rules about late homework submissions apply. Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. The submissions that do not obey these rules will not be graded.

5. To increase the efficiency of the grading process as well as the readability of your code, you must follow the following instructions about the format and general layout of your program.

    - Indentation, indentation, indentation... The format of your code makes it readable!

    - This homework will be graded by your TA, Naz Dündar. Thus, you may ask them your homework related questions. You are also welcome to ask your course instructor Bora Çelikkale for help.