

At this lab section, we will experiment evaluation trees.

Expression Trees

Asst. Prof. Dr. Feriştah DALKILIÇ Res. Asst. Fatih DİCLE

## PART 1 – Expression Tree

You are given some necessary interfaces and classes to build an expression tree, and some parts are left blank for you to fill them according to the instructions. After completing the blank parts, you should generate an expression tree for the below equation, display the prefix and postfix form of the given expression and evaluate the result.

(5 \* 2 - 5) \* 2 - 4 / 2 + 9 / 3

You are expected to fill the given missing parts:

In **class** DriverET

|  |
| --- |
| **public** **static** **void** main(String[] args)  {  // Generate an expression tree and name it as "myTree"  // for the expression "(5 \* 2 - 5) \* 2 - 4 / 2 + 9 / 3"    // Display the expression in prefix form by using displayPrefixExpression() method    // Display the expression in postfix form by using displayPostfixExpression() method    // Display the result of the equation by using evaluate() method  } |

In **class** ExpressionTree

|  |
| --- |
| **private** **double** evaluate(BinaryNode<String> rootNode)  {  **double** result;  **if** (rootNode == **null**)  result = 0;  **else** **if** (rootNode.isLeaf())  {  String variable = rootNode.getData();  result = Integer.*parseInt*(variable);  }  **else**  {  // calculate the result for nonleaf nodes (Hint: use recursive function)  // and use "compute" function in the final step  }  **return** result;  } |

|  |
| --- |
| **private** **void** postorder(BinaryNode<String> rootNode)  {  // This function should print the tree in the postorder  } |

|  |
| --- |
| **private** **void** preorder(BinaryNode<String> rootNode)  {  // This function should print the tree in the preorder  } |

**Example Scenario:**

To illustrate, you are given an equation as: 5 \* (3 + 2 \* 6) / 3

It can be represented as an expression tree below:

A diagram of a diagram

Description automatically generated

**Example Output:**

prefix form: / \* 5 + 3 \* 2 6 3

postfix form: 5 3 2 6 \* + \* 3 /

result: 25