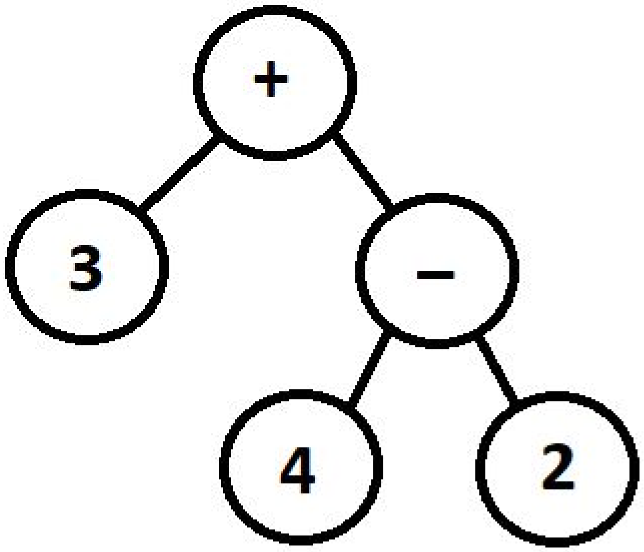
Do this with fellow students **on a piece of paper**, or each do your own solution and then compare. Discuss each step and make sure you agree on your understanding of these trees.

Prefix statement trees

Imagine a prefix statement as a binary tree.
Each operator is a node that has two children, it's left and right operands.
Each number is a leaf.

Example: this is the tree for the prefix statement **"+ 3 - 4 2"**:

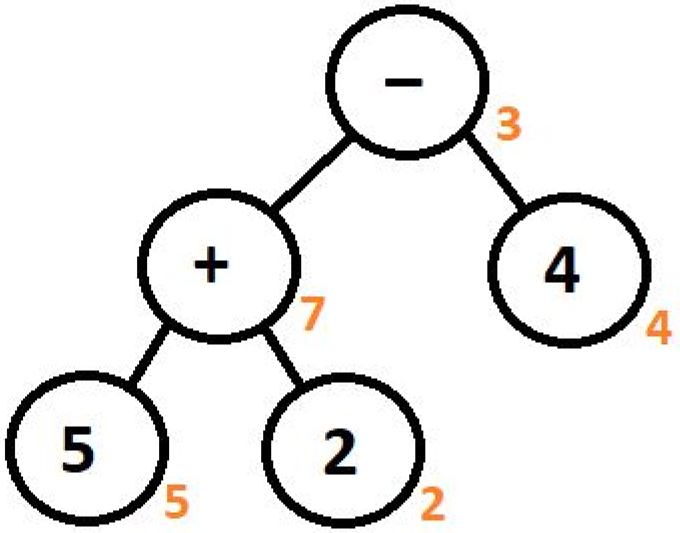


Draw a tree for each of the following statements:
* - 7 1 2
- * - 5 3 - 2 8 + - 8 4 8
+ + 0 7 - 2 7
* - - 7 3 + 3 3 * 1 5
+ * * 1 6 0 - - 2 5 1

More on next page...

Evaluating trees:
Each node, both leaves and other nodes have a numeric value.
The numeric value of a leaf is simply its number.
The numeric value of an operator node is the result of using that operator on the numerical values of its two children.

Example: this is the tree for the prefix statement **"- + 5 2 4"** with numerical values:



For each tree that you have drawn, go step by step through the process of calculating the numerical value of the root.

If this is done recursively, where each call to a recursive operation takes a single node, would these calculations happen in the same order as in the previous class assignment on the recursive PrefixParser?

Try to imagine these operations happening recursively, where each call evaluates one node, and then evaluates its children by calling itself with each child as a parameter, waiting for the result and then continuing, either to the next child, or by calculating and returning a result.