**Expression Tree**

* **Definition:** An expression tree is a **representation of expressions arranged in a tree-like data structure**. In other words, it is a tree with leaves as operands of the expression and nodes contain the operators.
* **Constructing an Expression tree**

**Algorithm(scan postfix expression from left to right)**

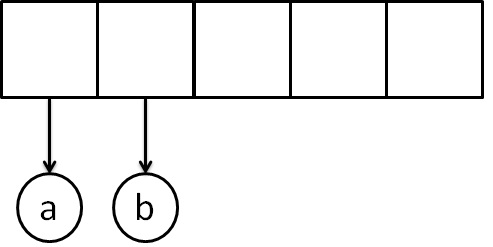
1. Get one symbol at a time
2. If symbol is operand then create a node and push pointer onto stack
3. If symbol is operator then pop pointer to two trees T1 and T2 from stack and form a new tree whose root is operator and whose left and right children point to T1 and T2 respectively.
4. Repeat Steps 1,2 and 3 till end of expression
5. Pop the pointer from stack which is pointer to the root of expression tree.

Example

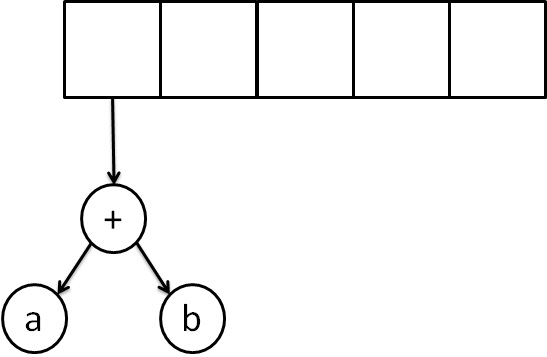
The input is:

a b + c \*

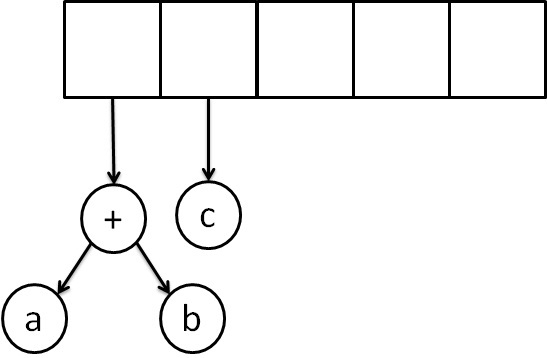
The first two symbols are operands, we create one-node tree and push a pointer to them onto the stack.

[](https://www.krivalar.com/picture/tree/exp/exp1.jpg)

Next, read a**'+'** symbol, so two pointers to tree are popped, a new tree is formed and push a pointer to it onto the stack.



Next, **'c'** is read, we create one node tree and push a pointer to it onto the stack.



Finally, the last symbol is read **' \* '**, we pop two tree pointers and form a new tree with a, **' \* '** as root, and a pointer to the final tree remains on the stack.

