Sheet 5 (ADT Binary Tree)

A. Implement Linked-based ADT Binary Tree as shown in lecture.
   1. Binary tree, implement the following methods:

ADT Binary Tree Operations

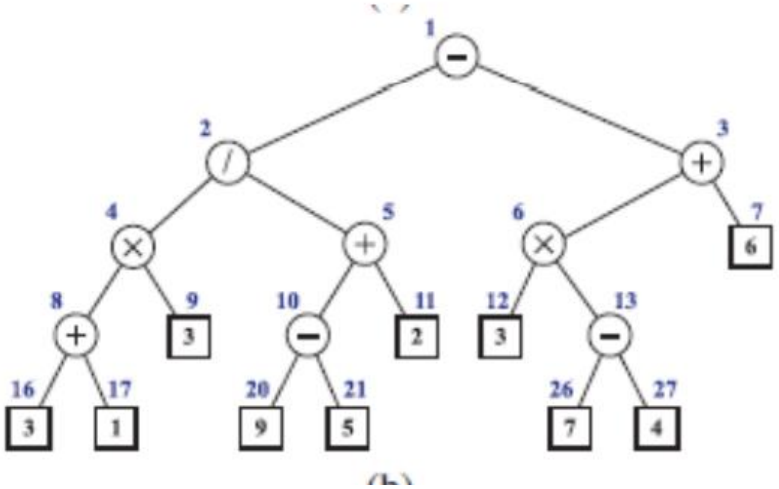
- Each node of the tree is associated with a **position** object p, and the following functions

p.parent():	Return the parent of p; an error occurs if p is the root.
p.children():	Return a position list containing the children of node p.
p.isRoot():	Return true if p is the root and false otherwise.
p.isExternal():	Return true if p is external and false otherwise.

The tree itself provides the following functions

size():	Return the number of nodes in the tree.
empty():	Return true if the tree is empty and false otherwise.
root():	Return a position for the tree's root; an error occurs if the tree is empty.
addRoot(e):	Creates a root for an empty tree, storing e as the element, and returns the position of that root; an error occurs if the tree is not empty.
addLeft(p, e):	Creates a left child of position p, storing element e, and returns the position of the new node; an error occurs if p already has a left child.
addRight(p, e):	Creates a right child of position p, storing element e, and returns the position of the new node; an error occurs if p already has a right child.
set(p, e):	Replaces the element stored at position p with element e, and returns the previously stored element.
attach(p, T_1, T_2):	Attaches the internal structure of trees T_1 and T_2 as the respective left and right subtrees of leaf position p and resets T_1 and T_2 to empty trees; an error condition occurs if p is not a leaf.
remove(p):	Removes the node at position p, replacing it with its child (if any), and returns the element that had been stored at p; an error occurs if p has two children.

   2. Using ADT Binary Tree, implement a function that builds the following tree



3. Implement a function that evaluates the expression in the above tree.
4. Implement a function that writes the expression in the above tree using parentheses.