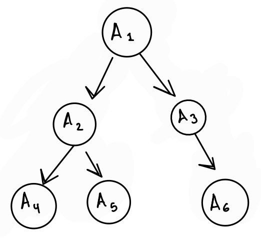
# TAD BST Tree

BST TREE = {a1, a2, a3 ... aN}

-a1 is the main element, a2 and a3 are subtrees of a1, any element less than a1 goes to the left, and any element to the right is greater than a1.



$A_4 < A_2 < A_5 < A_1 < A_3 < A_6$

Inv: {a1>a2, a1<a3} for any BST tree and sub tree, to the left of the element, elements are less than and to right of the element, elements are greater than

Primitive operations:

| | |
|---|---|
| -createBST: | ->BST |
| -Insert: Element x BST | ->BST |
| -delete: Element x BST | ->BST |
| -search: BST | ->Element |
| -searchElement: BST | ->Element |

## Insert(K key,E newItem) : Modifier

"Insert a new key inside the binary tree, if the key already exists, insert a new position"

{ pre: Binary Tree initializated }
{ post: Increments the depth of the branch with +1 in this specific sub-tree }

## Delete( K key): Modifier

"Delete a specific element or key from the binary tree"

{ pre: Binary Tree initializated }
{ post: Decrements the depth of the branch with -1 in this specific sub-tree }

## Search(K key): Analayzer

"Search a specific key value inside the Binary Tree and returns it"

{ pre: Binary Tree initializated }
{ post: Return the ArrayList of elements or return a "False" if the the key don't exists }

## SearchElement(K key): Analyzer

"Search a specific element with a unique key value and returns it"

{ pre: Binary Tree initializated }
{ post: Element : The element with the specific key value, if the element don't exists, it returns False }

## CreateBST( ) : Constructor

"Create (Initializate) a new empty Binary tree to add new elements"

{ pre: TRUE }
{ post: NewTree: The new created binary tree ready to add new elements }