|  |
| --- |
| **TAD BST** |
| BST<T,S> = {T value = null, S key = null} |
| { inv : x is the root and y a node. If y belongs to the left subtree, key[y] ≤ key[x] and if y belongs to the rigth subtree key[y] ≥ key[x]} |
| Primitive operations   * BST None  BST * insert Object, Object  BST * search Object  BST * delete BST  None * getSuccessor BST  BST * getMinimum BST  BST * getMaximum BST  BST * isLeaf BST 🡪 boolean * isEmpty None 🡪 boolean * bstHeight BST 🡪 int |

Constructor

|  |
| --- |
| BST()  “initialize the constructor of the BST class  {pre: none}  { post : BST = { T value= null, S key = null } } |

Modifier

|  |
| --- |
| insert(T value, S key)  “This method adds a new node with a value and key”  { pre : constructor already initialized, value & key Will be different of a null value}  { post : a node with value and key was added to the BST} |

Analyzer

|  |
| --- |
| search(S key)  “This method search and return the node BST if that belong to BST”  { pre : constructor already initialized }  { post : a null value if that key does not belong to the BST or an BST node if belong } |

|  |
| --- |
| getSuccesor(node BST)  “This method search and return the succesor node”  { pre : constructor already initialized, node different from null }  { post: null node if not exist a successor or the successor BST node } |

|  |
| --- |
| getMinimum(node BST)  “This method search and return the minimum value contained in the BST”  { pre : constructor already initialized, node BST different from null }  { post: Node containing the key with the minimum value} |

|  |
| --- |
| getMaximum(node BST)  “This method search and return the maximum value contained in the BST”  { pre : constructor already initialized, node BST different from null }  { post: Node containing the key with the maximum value } |

|  |
| --- |
| isLeaf(node BST)  “This method search if a BST node has or not children”  { pre : constructor already initialized }  { post: true if the node is a leaf or false if not} |

|  |
| --- |
| bstHeight(node BST)  “This method returns the height of the tree”  { pre : constructor already initialized , node different from null}  { post: integer that contain the tree height} |

|  |
| --- |
| isEmpty()  “This method return a boolean value indicating if the BST is empty or not”  { pre : constructor already initialized }  { post: true if the BST is empty or false if not } |

Destruction

|  |
| --- |
| delete(node BST)  “This method delete the node that is containing in BST”  { pre : constructor already initialized, node different from null }  { post: delete the BST node if that node is contained in the tree} |