|  |
| --- |
| **TAD AVL Tree** |
| Atributos o desctipción |
|  |
| Operaciones básicas   * Operacions Basicas   + AVL TREE   + Left rotate   + Right rotate   + Rebalance   + Insert   + Delete   + getRoot   + getMin   + getMax   + getPredecessor   + getSuccesor   + getWeight   + search   + is in tree   + search Equal To   + search lower or equal too |

Operaciones

|  |
| --- |
| AVL Tree  “Este metodo se encargad e crear un árbol AVL”  Pre:  Post: Se creo un nuevo árbol AVL |
| leftRotate(x)  “Este método se encarga de rotar cierto nodo hacia la izquierda”  Pre: AVLTree!=null, AvlTree desbalanceado  Post:Nodos rotados hacia la izquierda |
| RightRotate(x)  “ Este método se encarga de rotar cierto nodo haia la derecha”  Pre: AvlTree!=null, AVLTree desbalanceado  Post: Nodo rotado a la derecha |
| Rebalance(n)  “Este metodo se encarga de rebalancear el árbol siempre y cuando se agregue o se elimine un nodo del mismo”  Pre: AVLTree ¡= null, AVLTree Weight>2, AVL Tree desbalanceado  Post: El arbol es balanceado |
| Insert(z)  “Este método se encarga de insertar un nodo dentro del árbol AVL y llama al metodo rebalancear si es necesario”  Pre:  Post: Nodo agregado |
| Delete(z)  “Este método se encarga de eliminar el nodo z del árbol y llama al método rebalancear si es necesario”  Pre:AVLTree!= null, z esta el árbol,  Post:Se elimina el nodo z |
| getRoot()  “Este método retorna la raíz del árbol”  Pre:  Post: |
| GetMin() |
| “Este emtodo retorna el valor más pequeño en el árbol  Pre: AVL Tree!= null  Post: |
| getMax()  “Este metodo retorna el valor más grande del árbol” pre: AVL Tree!= null  Post: |
| getPredecessor(x)  “Este metodo retorna el valor que sigue antes de x”  Pre: AVLTree!=null,  Post: |
| getSuccesor(x)  “Este metodo retorna el valor que le sigue después de x”  Pre: AVLTree!=null,  Post: |
| getWeight()  “Este metodo retorna el peso total del árbol”  Pre:  Post: |
| Search(z)  “Este metodo busca al nodo z en el árbol y lo retorna”  Pre: AVLTree!=null  Post: |
| IsInTree(z)  “Este metodo revisa si el nodo z se encuentra en el árbol”  Pre: AVLTree!=null  Post: |

**Diseños de casos de pruebas unitarias**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba : Se va a probar si el método insertar, agrega los valores de la manera adecuada y rebalancea el arbol | | | | |
| Clase | Método | Escenario | Entrada | Resultado |
| AVLTree | Insert() | Hay un árbol AVL vacio | (30,1) | GetRoot().getKey()==30 |
| AVLTree | Insert() | Hay un árbol AVL con las llaves: 30 | (15,2)  (40,3) | GetRoot().getLeft().getKey()==15  GetRoot().getRight().getKey()==40 |
| AVLTree | Insert() | Hay un árbol AVL con las llaves: 30,15,40 | (45,19  (46,19) | Search(45).getLeft().getKey()==40  Search(45).getRight().getKey()==46 |
| AVLTree | Insert() | Hay un árbol AVL con las llaves: 30,15,40,45,46 | (12,1)  (7,1) | Search(12).getLeft().getKey()==7  Search(12).getRight().getKey()==15 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba : | | | | |
| Clase | Método | Escenario | Entrada | Resultado |
| AVLTree | Delete() | Hay un árbol AVL con las llaves: 30,15,40,45,46,12,7 | 30 | GetRoot().getKey()==40 |
| AVLTree | Delete() | Hay un árbol AVL con las llaves: 15,40,45,46,12,7 | 45 | getRoot().getRight().getKey()==46 |
| AVLTree | Delete() | Hay un árbol AVL con las llaves: 15,40,46,12,7 | 46 | getRoot.getKey()==15  getRoot.getRight.getKey()==40 |