|  |  |
| --- | --- |
| **TAD** | RedBlackTree |
| **Grafico** | **A=** |
| **Invariantes** | * La raíz del árbol debe ser de color negro * Los nodos de color rojo solo pueden tener hijos nodos de color negro * Los hijos de cada nodo hoja deben ser nil * Cada nodo nuevo que se agrega debe de ser color rojo * Los nodos deben ser de color rojo o negro * La altura negra de un nodo debe ser igual tanto por izquierda como por derecha |
| **Operaciones** | * Buscar Nodo x A 🡺 Nodo |
| * Insertar Nodo x T 🡺 A |
| * Eliminar A x Nodo 🡺 A |

Eliminar(N)

“Elimina el nodo N del árbol roji-negro”

{Pre: True}

Insertar (T)

“Inserta un nodo genérico en el árbol roji-negro”

{Pre: True}

Buscar (N)

“Busca el Nodo N en el árbol roji-negro”

{Pre: True}

|  |  |
| --- | --- |
| **TAD** | AVLTree |
| **Grafico** | **A=** |
| **Invariantes** | * El factor de balance de cada nodo debe estar entre -1 y 1 * El árbol debe de estar balanceado |
| **Operaciones** | * Buscar Nodo x A 🡺 Nodo |
| * Insertar Nodo x T 🡺 A |
| * Eliminar A x Nodo 🡺 A |

Eliminar(N)

“Elimina el nodo N del árbol AVL”

{Pre: True}

Insertar (T)

“Inserta un nodo genérico en el árbol AVL”

{Pre: True}

Buscar (N)

“Busca el Nodo N en el árbol AVL”

{Pre: True}