Cristian Cardona

Jesus Garces

TAD Seguimiento 11

TAD Tree		
Tree<T> { Node = <root> }		
{ inv: Node.getKey ∈ ℤ}		
add triggerInOrder inorder triggerSearch search getMin getMax triggerDelete delete	Entero x T Node Entero Node x Entero Node Node Entero Node x Entero	→Node →Texto →Texto →Node →Node →Node →Node →Node →Node

add(int, T)

“Agrega un nodo al arbol binario”

{pre: Node={…, T: <T>} ∧ int ∈ ℤ }

{post: Node(int , T) }

triggerInOrder()

“Activa el método recursivo (inorder)”

{pre: }

{post: <inorder> }

Inorder(Node)

“Imprime las llaves de los nodos de menor a mayor”

{pre: Node={…,Node: <node> …}}

{post: <Texto>}

triggerSearch(int)

“Activa el metodo recursive (search)”

{pre: {…,Int<int>,…}}

{post: <Node>}

search(Node, int)

“Busca y retorna un nodo usando su llave”

{pre: Node={…,Node<node>,…} ∧ $int \in \mathbb{Z}$}

{post: <Node>}

getMin(Node)

“Obtiene el nodo con la llave más baja del árbol”

{pre: Node={…,Node:<Node>,…}}

{post: <Node>}

getMax(Node)

“Obtiene el nodo con la llave de valor más alta del árbol”

{pre: Node={…,Node:<Node>,…} }

{post: <Node>}

triggerDelete(int)

“Activa el metodo recursive (delete)”

{pre: {…,Int<int>,…}} }

{post: <Node>}

delete(Node, int)

“Elimina un nodo del arbol usando la llave”

{pre: Node={…,Node:<node>,…} $int \ \wedge \in \mathbb{Z}$}

{post: <Node>}

TAD Node		
Node<T> { key = <int>, T = <value>, Node = <right>, Node = <left> }		
{inv: key ∈ $\mathbb{Z}$}		
insert getKey setKey getValue setValue getRight setRight getLeft setLeft	Entero x T Node Node x Entero Node Node x T Node Node Node Node	→Node →Entero →Entero →T →T →Node →Node →Node →Node

insert(int, T)

"Inserta un nodo al rabol binario"

{pre: Node={…,T:<T>,…}int ∧ ∈ $\mathbb{Z}$}

{post: <Node(int,T)>}

getKey(Node)

"Obtiene la llave de un nodo"

{pre:Node={…,Key : <int> ,…} Key }

{post: <key>}

setKey(Node, k)

“Cambia el valor de llave de un nodo”

{pre:Node={…,Key:<int> ,…} k ∧ ∈ ℤ }

{post: <node.getKey = k>}

getValue(Node)

“Obtiene el valor que almacena un nodo”

{pre:Node={…,Value: <T> ,…}}

{post: <Value>}

setValue(Node, t)

“Cambia el valor que almacena un nodo”

{pre: Node={…,Value:<T> ,…}}

{post: <Node.getValue = t>}

getRight(Node)

“Obtiene el valor de la derecha en la lista enlazada”

{pre: Node={…,Node:<right> ,…}}

{post: <Node>}

setRight(Node, R)

“Cambia el valor de la derecha en la lista enlazada”

{pre: Node={…,Node:<right> ,…} $R \wedge \in Node$ }

{post: <Node.getRight = R>}

getLeft(Node)

“Obtiene el valor de la izquierda en la lista enlazada”

{pre:Node={…,Node:<left>,…}}

{post: <Node>}

setLeft(Node, L)

“Cambia el valor de la izquierda en la lista enlazada”

{pre: Node={…,Node<left> ,…}$L \wedge \in Node$}

{post: <Node.getLeft = L>}