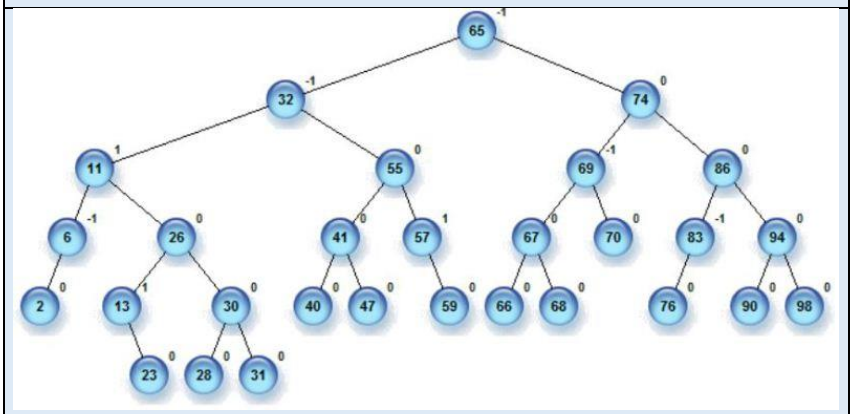
TAD ARN



{*inv.: Se deben cumplir las siguientes propiedades: (1)El factor de balanceo en un árbol AVL no debe diferir de 1. (2) El factor de balanceo debe estar entre -1 y 1, teniendo en cuenta el valor 0*}

Operaciones Primitivas:

- CrearAVL: ->AVL
- InsertarEnAVL: *AVLxLlavexValor* ->AVL
- BalancearInsertar: *AVLxNodo* ->AVL
- EliminarEnAVL: *AVLxLlave* ->AVL
- BalancearEliminar: *AVLxLlave* ->AVL
- RotarIzquierda: *AVLxNodo* ->AVL
- RotarDerecha: *AVLxNodo* ->AVL

CrearAVL()

"Crea un árbol AVL con raíz nula"

{*pre:* TRUE}

{*post: Se crea un árbol AVL vacío*}

InsertarEnAVL(Llave, Valor**)**

"Se insertar un nuevo nodo al árbol AVL. Este nodo va a tendrá Llave y Valor"

{*pre:* AVL!=nulo}

{*post: Se agrega un nuevo nuevo nodo al árbol AVL*}

BalancearInsertar(Nodo**)**

"Se corrige el factor de balanceo de cada nodo del sub-árbol AVL de acuerdo con el nodo dado, llevando a cabo la rotaciones necesarias siempre y cuando sea necesario. Esto se hace cada vez que se agrega un nuevo nodo"

{*pre:* AVL!=nulo}

{*post: Si algún factor del sub-árbol se excede de 1 -> Se aplica el balanceo apropiado al AVL*
De lo contrario -> No se hace ninguna modificación}

EliminarEnAVL(Llave**)**

"Elimina un nodo del árbol AVL de acuerdo con la llave dada"

{*pre:* AVL!=nulo}

{*post: Si la Llave ∈ a algún nodo en el AVL -> Se elimina este nodo y se llevan a cabo las rotaciones necesarias,*
De lo contario -> No se hace ninguna modificación}

BalancearEliminar(Nodo**)**

"Se corrige el factor de balanceo de cada nodo del sub-árbol AVL de acuerdo con el nodo dado, llevando a cabo la rotaciones necesarias siempre y cuando sea necesario. Esto se hace cada vez que se elimina un nodo"

{*pre:* AVL!=nulo}

{*post: Si algún factor del sub-árbol se excede de 1 -> Se aplica el balanceo apropiado al AVL*
De lo contrario -> No se hace ninguna modificación}

RotarIzquierda(Nodo**)**

"Se rota a la izquierda el sub-árbol a partir del nodo dado, teniendo en cuenta las condiciones necesarias del AVL"

{*pre:* AVL!=nulo}

{*post: Se rota el sub-árbol a la izquierda teniendo como punto de partida el nodo dado*}

RotarDerecha(Nodo**)**

“Se rota a la derecha el sub-árbol a partir del nodo dado, teniendo en cuenta las condiciones necesarias del AVL”

{*pre:* AVL!=nulo}

{*post: Se rota el sub-árbol a la derecha a teniendo como punto de partida el nodo dado*}