**EJERCICIO 1**

Insertar un nuevo nodo en el árbol.

**LENGUAJE NATURAL:**

En el caso de que el árbol no contenga ningún elemento, el nodo a insertar será la raíz del árbol. En todos los casos siguientes de inserción se debe ir verificando si el nodo actual en el que se está parado, la clave del nodo a insertar es mayor o menor, en el caso de que sea

menor, se verificará si el hijo izquierdo existe, si no existe se inserta el nodo allí. Si el nodo a insertar tiene clave mayor a la clave del nodo actual, se verificará si el hijo derecho existe, si no existe se inserta allí el nodo.

PRE CONDICIONES:

-El nodo a insertar debe contener una clave de tipo comparable.

-El árbol debe ser binario de búsqueda.

POST CONDICIONES:

-Al finalizar la inserción el árbol debe contener un nodo más.

-Si buscamos el nodo en el árbol, este debe ser encontrado.

PSEUDOCÓDIGO:

insertarNodo()

COM

metodoArbol(nodo)

SI raiz = nulo entonces

raiz <- nodo

SINO

raiz.insertarNodo(nodo)

FIN SI

metodoNodo(nodo)

SI nodo.clave < raiz.clave entonces

SI hijoIzq = nulo entonces

hijoIzq <- nodo

SINO

hijoIzq.insertarNodo(nodo)

FIN SI

SINO

SI hijoDer = nulo entonces

hijoDer <- nodo

SINO

hijoDer.insertarNodo(nodo)

FIN SI

FIN SI

FIN

**EJERCICIO 2**

Contar todas las hojas

**LENGUAJE NATURAL:**

Se recorre la totalidad de nodos del árbol y se evalúa cada nodo para ver si tienen hijos, en caso de que el nodo tenga uno o dos hijos se continua con el recorrido, en caso de que el nodo no contenga hijos, se sumará una unidad en una variable. Al final del recorrido se devolverá el valor de la variable.

PRE CONDICIONES:

-El árbol es de búsqueda binaria

POST CONDICIONES:

-Al finalizar el recorrido, el árbol y los nodos no deben haber sufrido modificaciones.

-Si el árbol contiene al menos un nodo, el resultado del contador debe ser mayor a 0.

-Si el árbol contiene un sólo elemento, el contador debe ser igual a 1.

PSEUDOCÓDIGO:

contarHojas(): de tipo entero

COM

metodoArbol()

SI raiz = nulo entonces

devolver 0

SINO

raiz.contarHojas()

FIN SI

metodoNodo()

contador <- 0

SI hijoIzq = nulo y hijoDer = nulo entonces

contador++

FIN SI

SI hijoIzq <> nulo entonces

contador = contador + hijoIzq.contarHojas()

FIN SI

SI hijoDer <> nulo entonces

contador = contador + hijoDer.contarHojas()

FIN SI

devolver contador

FIN

**EJERCICIO 3**

Calcular suma de claves

**LENGUAJE NATURAL:**

Se recorre en orden la totalidad de los nodos del árbol, por cada nodo, se lee su clave y se la suma a un acumulador inicializado en 0 antes del

recorrido. Una vez finalizado el recorrido, se devolverá el resultado obtenido en la variable.

PRE CONDICIONES:

-El árbol es de búsqueda binaria

POST CONDICIONES:

-Al finalizar el recorrido, el árbol y los nodos no deben haber sufrido modificaciones.

-Si el árbol contiene al menos un nodo con clave mayor a 0, el resultado del acumulador debe ser mayor a 0.

-El resultado del acumulador no puede ser negativo.

PSEUDOCÓDIGO:

sumaClaves(): de tipo entero

COM

metodoArbol()

SI raiz == nulo entonces

devolver 0

SINO

raiz.sumaClaves()

FIN SI

metodoNodo()

acumulador <- 0

acumulador = acumulador + this.clave

SI hijoIzq <> nulo entonces

acumulador = acumulador + hijoIzq.sumaClaves()

FIN SI

SI hijoDer <> nulo entonces

acumulador = acumulador + hijoDer.sumaClaves()

FIN SI

devolver acumulador

FIN