|  |
| --- |
| **Proyecto 2 · Manual técnico** |
| Karla Ernestina González Polanco 202006688 |
| Imagen que contiene dibujo, plato  Descripción generada automáticamente  ○ **Moviecats** ○ |
|  |

Tabla de contenidos

[**Descripción del programa** 2](#_Toc102049688)

[REQUERIMIENTOS DEL SISTEMA 2](#_Toc102049689)

# REQUERIMIENTOS DEL SISTEMA

# ESTRUCTURAS DE DATOS

## Lista simplemente enlazada

## Árbol Binario de Búsqueda

### Inserción

* **AgregarR:** Método que inicialmente recibe el valor que se insertará y el valor de su raíz el cual irá cambiando (aumentando el número de nodos) después de insertar este nuevo valor.
* **Agregar:**
  + Si no existe la raíz, retorna el valor en forma de nodo y ahora la raíz será este nodo.
  + Si ya hay una raíz se compara su valor (dni) con el del nuevo nodo que se quiere agregar y se hace una función recursiva.
    - Si es menor, se verifica si el nodo izquierdo es nulo (utilizando la función agregar pero como parámetro el nodo izquierdo al nodo actual), si lo es se asigna el nuevo nodo a la izquierda del nodo padre, si no es nulo se sigue haciendo la comparación hasta que el nodo izquierdo al nodo actual sea nulo (significaría que ya llegó al último nivel).
    - Si es mayor, se verifica si el nodo derecho es nulo (utilizando la función agregar pero como parámetro el nodo izquierdo al nodo actual), si lo es se asigna el nuevo nodo a la derecha del nodo padre, si no es nulo se sigue haciendo la comparación hasta que el nodo derecho al nodo actual sea nulo (significaría que ya llegó al último nivel).

Al final se retorna el nodo inicial (la cabeza) pero con el nuevo nodo ya insertado en la posición correspondiente y todos los demás nodos que tenía hasta antes de la inserción.

### Graficación

Utilizando graphviz y d3 para renderizar el archivo dot – primero se verifica si el nodo que se pasa como parámetro (primeramente, la raíz) es nulo, en caso de ser así no se agrega nada a dot. Si existe se va recorriendo el árbol viendo si el nodo actual tiene un valor a su derecha e izquierda y dependiendo de si tiene uno de los dos hijos o los dos se le agregan las etiquetas <C#> al nodo actual para así apuntarlos gráficamente. El mismo proceso se realiza recursivamente para cada nodo del árbol.

### Ordenamientos

Únicamente se hace un recorrido del árbol dependiendo del tipo de ordenamiento. Al pasar por el nodo actual se hace uso de la función newDiv que retorna el código HTML que tendrá el div del actor en el HTML. Su visualización inicial en el HTML es en el archivo Pages, función showActors donde se muestran en postorden.

## Árbol AVL

Gráfico, Gráfico de dispersión, Gráfico de burbujas

Descripción generada automáticamente

### Inserción

* **agregarNodo:** Método que inicialmente recibe el valor que se insertará y el valor de su raíz el cual irá cambiando (aumentando el número de nodos) después de insertar este nuevo valor.
* **Agregar:**
  + Si no existe la raíz, retorna el valor en forma de nodo y ahora la raíz será este nodo.
  + Si ya hay una raíz se compara su valor (id) con el del nuevo nodo que se quiere agregar y se hace una función recursiva.
    - Si es menor, se verifica si el nodo izquierdo es nulo (utilizando la función agregar pero como parámetro el nodo izquierdo al nodo actual), si lo es se verifica la altura que tendrá el nodo restando la altura del nodo derecho – la altura del nodo izquierdo. En caso de que el resultado sea -2, significará que el árbol no está balanceado y tendrá que hacerse otra verificación.

Si el valor del id actual es menor al id del nodo izquierdo al actual, significará que el desbalanceo viene de la izquierda por lo que tendrá que hacerse una rotación simple a la izquierda (figura 1. derecha arriba). Si el valor es mayor, el desbalanceo viene de la derecha por lo que será necesaria una rotación doble a la izquierda (figura 1. Izquierda arriba).

* + - Si es mayor, se verifica si el nodo derecho es nulo (utilizando la función agregar pero como parámetro el nodo derecho al nodo actual), si lo es se verifica la altura que tendrá el nodo restando la altura del nodo derecho – la altura del nodo izquierdo. En caso de que el resultado sea 2, significará que el árbol no está balanceado y tendrá que hacerse otra verificación.

Si el valor del id actual es mayor al id del nodo derecho al actual, significará que el desbalanceo viene de la derecha por lo que tendrá que hacerse una rotación simple a la derecha (figura 1. derecha abajo). Si el valor es menor, el desbalanceo viene de la izquierda por lo que será necesaria una rotación doble a la derecha (figura 1. Izquierda abajo).

* + - Si no es mayor ni menor, significará que está repetido por lo que solo se asigna nuevamente el valor al nodo.

Se cambia el atributo de altura del nodo con el método de maxheight sumándole 1 ya que se agregó un nuevo nodo. Se retorna el nodo con todos los nuevos nodos agregados al árbol.

### Rotaciones

* **Rotación simple a la izquierda:** Figura 1 – izquierda arriba. Recibe como parámetro un nodo, que al ver la figura será el nodo Z y hace uso de un auxiliar que será el nodo Y.

Primero se hace el cambio de hijos, el hijo derecho del nodo auxiliar Y pasa a ser el hijo derecho del nodo Z y el nodo Z pasa a ser el hijo derecho del nodo auxiliar. Se asignan las nuevas alturas de ambos nodos que ahora son parte de un árbol balanceado sumando 1. Por último, ahora el nodo en el nivel superior de la rotación será el nodo auxiliar por lo que se retorna este.

* **Rotación simple a la derecha:** Figura 1 – derecha abajo. Recibe como parámetro un nodo, que al ver la figura será el nodo Z y hace uso de un auxiliar que será el nodo Y.

Primero se hace el cambio de hijos, el hijo izquierdo del nodo auxiliar Y pasa a ser el hijo izquierdo del nodo Z y el nodo Z pasa a ser el hijo izquierdo del nodo auxiliar. Se asignan las nuevas alturas de ambos nodos que ahora son parte de un árbol balanceado sumando 1. Por último, ahora el nodo en el nivel superior de la rotación será el nodo auxiliar por lo que se retorna este.

* **Rotación doble por la izquierda:** Recibe como parámetro un nodo al cual se asignará a su derecha una rotación simple a la izquierda de su nodo derecho. Por último, se retorna una rotación

## Tabla HASH

\*si el dato se inserta a la lista no se suma

El tamaño inicial de la tabla es de 20 posiciones, al sobrepasar el 75% se hace un rehashing – aumentar 5 posiciones por cada posición ocupada.

20 \* 75% = 15

15 \* 5 = **75 ->** nuevo tamaño de la tabla

### Inserción

* Ecuación para inserción:

índice = id\_categoría % tamaño