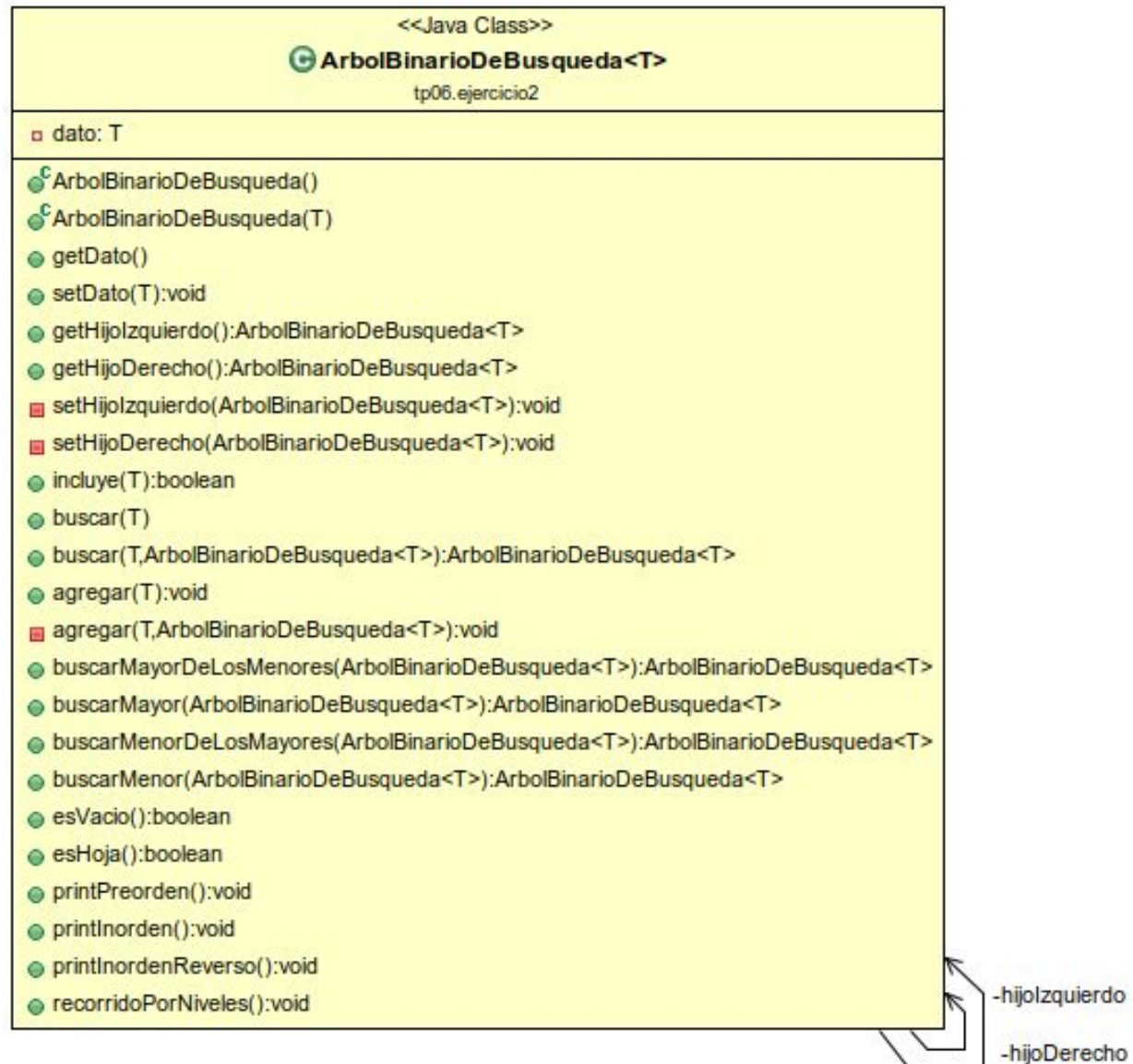


La interface Comparable<T> en Árboles Binarios de Búsqueda

El código de esta presentación es a nivel informativo.
No se implementarán estas estructuras en la práctica.

Árboles Binarios de Búsqueda

Estructura



Árboles Binarios de Búsqueda- Estructura

```
public class ArbolBinarioDeBusqueda<T extends Comparable<T>> {
    private T dato;
    private ArbolBinarioDeBusqueda<T> hijoIzquierdo;
    private ArbolBinarioDeBusqueda<T> hijoDerecho;


    public ArbolBinarioDeBusqueda() {
    }
    public ArbolBinarioDeBusqueda(T dato) {
        this.dato = dato;
    }
    public T getDato() {
        return dato;
    }
    public void setDato(T dato) {
        this.dato = dato;
    }
    public ArbolBinarioDeBusqueda<T> getHijoIzquierdo() {
        return this.getHijoIzquierdo();
    }
    public ArbolBinarioDeBusqueda<T> getHijoDerecho() {
        return this.getHijoDerecho();
    }
    . . .
    public boolean esVacio() {
        return (this.dato == null && this.esHoja());
    }
}
```

Árboles Binarios de Búsqueda

Ejemplos de instanciación y uso de árboles binarios de búsqueda con diferentes tipos:

```
ArbolBinarioDeBusqueda<Integer> abb = new ArbolBinarioDeBusqueda<Integer>();  
abb.agregar(new Integer(2));  
abb.agregar(6);  
abb.agregar(1);  
abb.agregar(13);  
abb.agregar(5);
```

Clases que
implementan la
interface Comparable



```
ArbolBinarioDeBusqueda<Persona> abb = new ArbolBinarioDeBusqueda<Persona>();  
Persona p = new Persona("Paula", "Gomez", 16);  
abb.agregar(p);  
p = new Persona("Ana", "Rios", 6);  
abb.agregar(p);  
p = new Persona("Maria", "Ferrer", 55);  
abb.agregar(p);  
Alumno a = new Alumno("Luis", "Rios", 18); // Alumno subclase de Persona  
Abb.agregar(a);
```

Árboles Binarios de Búsqueda

Inserción de un dato (Weiss)

```
package ayed2021;

public class ArbolBinarioDeBusqueda<T extends Comparable<T>> {
    private T dato;
    private ArbolBinarioDeBusqueda<T> hijoIzquierdo;
    private ArbolBinarioDeBusqueda<T> hijoDerecho;
    . . .

    public void agregar(T x) {
        if (dato == null)
            dato = x;
        else
            this.agregar(x, this);
    }

    private void agregar(T x, ArbolBinarioDeBusqueda<T> t) {
        if (x.compareTo(t.getDato()) < 0)
            if (!t.tieneHijoIzquierdo())
                t.setHijoIzquierdo(new ArbolBinarioDeBusqueda<T>(x));
            else
                this.agregar(x, t.getHijoIzquierdo());
        else if (x.compareTo(t.getDato()) > 0)
            if (!t.tieneHijoDerecho())
                t.setHijoDerecho(new ArbolBinarioDeBusqueda<T>(x));
            else
                this.agregar(x, t.getHijoDerecho());
    }
}
```

```
ArbolBinarioDeBusqueda<Integer> abb =
    new ArbolBinarioDeBusqueda<Integer>();
abb.insertar(2);
abb.insertar(6);
abb.insertar(1);
abb.insertar(13);
abb.insertar(5);
abb.insertar(7);
abb.recorridoporNiveles();
```

