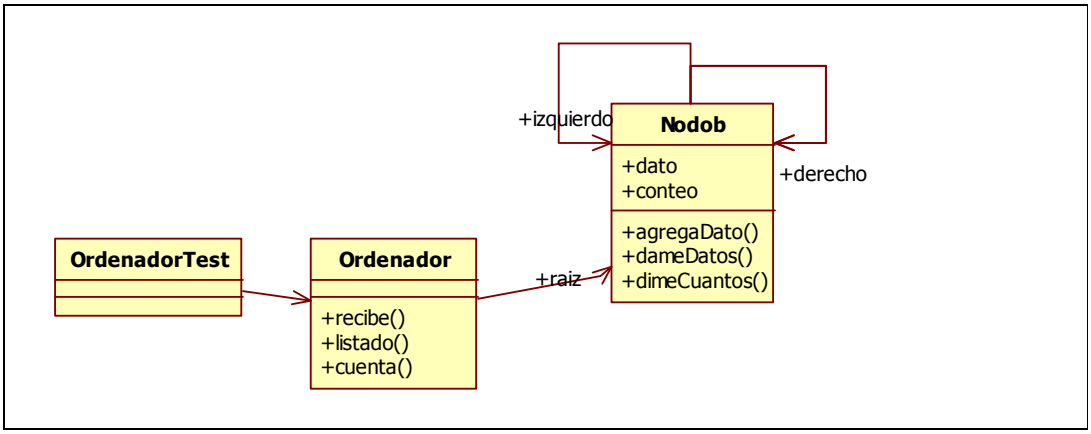
Programación Avanzada
Dr. Juan Manuel Fernández Peña
Curso 2013

A continuación se muestran las clases Nodob y Ordenador, las cuales permiten ordenar una lista de cadenas alfanuméricas utilizando el método de árbol binario. El ejemplo muestra cómo una clase puede estar relacionada consigo misma (Nodob), es decir, los objetos de ésta clase están interrelacionados con otros objetos de la misma clase. La clase de prueba permite apreciar algunos elementos de su funcionamiento.

Se muestran las clases en UML y el código de las clases.

Clases en UML



Código

```
public class Nodob {
	/*
	 * Ejemplo de construcción de árbol binario para ordenamiento
	 * Juan Manuel Fernández. Marzo 2009
	 */

	//atributos
	private String dato;
	private int conteo;
	private Nodob izquierdo, derecho;

	//constructor
	public Nodob(String dd){
		dato = dd;
		izquierdo = null;
		derecho = null;
		conteo = 0;
	}

	//métodos
	public void agregaDato(String dd){
		int res = dato.compareTo(dd);
		if (res == 0) conteo++;
		else
			if (res>0)
				if (izquierdo == null)
					izquierdo = new Nodob(dd);
```

```
                        else izquierdo.agregaDato(dd);
                  else
                        if (derecho == null)
                              derecho = new Nodob(dd);
                        else derecho.agregaDato(dd);
      }

      public String dameDatos(){
            String resp="";
            if (izquierdo != null) resp+=izquierdo.dameDatos()+", ";
            resp+=dato;
            if (derecho != null) resp+=", "+derecho.dameDatos();

            return resp;
      }

      public int dimeCuantos(){
            int res =1;
            if (izquierdo != null) res += izquierdo.dimeCuantos();
            if (derecho != null) res += derecho.dimeCuantos();
            return res;
      }
}
```

```
public class Ordenador {
      /*
       * Esta clase prepara un árbol binario con objetos de la
       * clase Nodob para ordenar una lista de cadenas de caracteres
       * Juan Manuel Fernández Peña. 2009, 2013
       */

      private Nodob raiz;

      public Ordenador(){
      }

      public void recibe(String algo){
            if (raiz == null)
                  raiz = new Nodob(algo);
            else raiz.agregaDato(algo);
      }

      public String listado(){
            if (raiz == null)
                  return " <lista vacía>";
            else
                  return raiz.dameDatos();
      }

      public int cuenta(){
            if (raiz == null)
                  return 0;
            else return raiz.dimeCuantos();
      }

}
```

```
import static org.junit.Assert.*;

import org.junit.Before;
import org.junit.Test;
```

```
public class OrdenadorTest {
	Ordenador q;
	@Before
	public void setUp() throws Exception {
		q = new Ordenador();
	}

	@Test
	public void testListado() {
		q.recibe("Salitre");
		assertEquals("Salitre",q.listado());
	}

	@Test
	public void testCuentaVacia() {
		assertEquals(0,q.cuenta());
	}

	@Test
	public void testAmplio() {
		q.recibe("Salitre");
		q.recibe("Begonia");
		q.recibe("Diamante");
		q.recibe("Tabla");
		q.recibe("Zapato");
		q.recibe("Ave");
		System.out.println(q.listado());
		assertEquals(6,q.cuenta());
	}
}
```