Algunos comentarios a los arrays de objetos

Supongamos las siguientes declaraciones:

```
int[] numeros = {12, 4, 2, 6, 7, -13, 23};
String[] palabras = {"casa", "tren", "bicicleta", "coche", "naranja"};
```

Si hacemos:

```
Arrays.sort(numeros);
Arrays.sort(palabras);
```

sabemos que *numeros* después de ordenar queda {-13, 2, 4, 6, 7, 12, 23} y
palabras queda con los valores {"bicicleta", "casa", "coche", "naranja", "tren"}

Si después de ordenar hacemos:

```
int pos = Arrays.binarySearch(numeros, 12);    // pos vale 5
pos = Arrays.binarySearch(numeros, 33);    // pos vale un valor < 0
pos = Arrays.binarySearch(palabras, "bicicleta");    // pos vale 0
pos = Arrays.binarySearch(palabras, "BICICLETA");    // pos vale un valor < 0
```

Ahora supongamos la clase Estudiante:

```
public class Estudiante
{
        private String nombre;
        private int nota;
        ……...

}
```

y un array Estudiante[] estudiantes = { new Estudiante("ana", 8),
new Estudiante("luis", 6),
new Estudiante("Alberto", 4),
new Estudiante("Rosa", 7) };

Funcionarán las siguientes sentencias?:

a) Arrays.sort(estudiantes); **NO**

b) Arrays.binarySearch(estudiantes, new Estudiante("Rosa", 7)); …… **NO**

En ninguno de los dos casos anteriores los métodos sort() y binarySearch() funcionarán.
Tras los métodos sort() y binarySearch() subyace el uso del interface Comparable.

Java debe saber qué criterio utilizar para ordenar nuestros objetos Estudiante. Hemos de decírselo.

Aprenderemos a hacerlo en la UT7.