8.B. Documentación del código.

1. Documentación del código.

Llegados a este punto, vamos a considerar una cuestión de gran importancia, la documentación del código fuente. Piensa en las siguientes cuestiones:

- ¿Quién crees que accederá a la documentación del código fuente? Pues serán los autores del propio código u otros desarrolladores.

- ¿Por qué hemos de documentar nuestro código? Porque facilitaremos su mantenimiento y reutilización.

- ¿Qué debemos documentar? Obligatoriamente: clases, paquetes, constructores, métodos y atributos. Opcionalmente: bucles, partes de algoritmos que estimemos oportuno comentar, ...

A lo largo de nuestra vida como programadores es probable que nos veamos en la necesidad de reutilizar, modificar y mantener nuestro propio código o incluso, código de otros desarrolladores. ¿No crees que sería muy útil que dicho código estuviera convenientemente documentado? ¿Cuántas veces no hemos leído código de otros programadores y quizá no hayamos comprendido qué estaban haciendo en tal o cual método? Como podrás comprender, la generación de una documentación adecuada de nuestros programas puede suponer una inestimable ayuda para realizar ciertos procesos en el software.

Si analizamos la documentación de las clases proporcionada en los paquetes que distribuye Sun, nos daremos cuenta de que dicha documentación ha sido generada con una herramienta llamada `Javadoc`. Pues bien, nosotros también podremos generar la documentación de nuestro código a través de dicha herramienta.

Si desde el principio nos acostumbramos a documentar el funcionamiento de nuestras clases desde el propio código fuente, estaremos facilitando la generación de la futura documentación de nuestras aplicaciones. ¿Cómo lo logramos? A través de una serie de comentarios especiales, llamados **comentarios de documentación** que serán tomados por `Javadoc` para generar una serie de archivos HTML que permitirán posteriormente, navegar por nuestra documentación con cualquier navegador web.

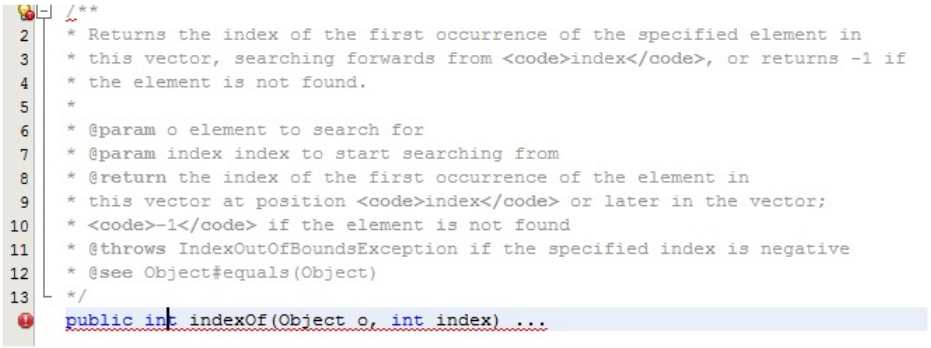
Los comentarios de documentación tienen una marca de comienzo (`/**`) y una marca de fin (`*/`). En su interior podremos encontrar dos partes diferenciadas: una para realizar una descripción y otra en la que encontraremos más etiquetas de documentación. Veamos un ejemplo:

```
/**
* Descripción principal (texto/HTML)
*
* Etiquetas (texto/HTML)
*/
```

Este es el formato general de un comentario de documentación. Comenzamos con la marca de comienzo en una línea. Cada línea de comentario comenzará con un asterisco. El final del comentario de documentación deberá incorporar la marca de fin. Las dos partes diferenciadas de este comentario son:

- **Zona de descripción:** es aquella en la que el programador escribe un comentario sobre la clase, atributo, constructor o método que se vaya a codificar bajo el comentario. Se puede incluir la cantidad de texto que se necesite, pudiendo añadir etiquetas HTML que formateen el texto escrito y así ofrecer una visualización mejorada al generar la documentación mediante `Javadoc`.
- **Zona de etiquetas:** en esta parte se colocará un conjunto de etiquetas de documentación a las que se asocian textos. Cada etiqueta tendrá un significado especial y aparecerán en lugares determinados de la documentación, una vez haya sido generada.

En la siguiente imagen puedes observar un ejemplo de un comentario de documentación.



```
/**
 * Returns the index of the first occurrence of the specified element in
 * this vector, searching forwards from <code>index</code>, or returns -1 if
 * the element is not found.
 *
 * @param o element to search for
 * @param index index to start searching from
 * @return the index of the first occurrence of the element in
 * this vector at position <code>index</code> or later in the vector;
 * <code>-1</code> if the element is not found
 * @throws IndexOutOfBoundsException if the specified index is negative
 * @see Object#equals(Object)
 */
public int indexOf(Object o, int index) ...
```

Reflexiona

Documentar el código de un programa es añadir suficiente información como para explicar lo que hace, punto por punto, de forma que no sólo los ordenadores sepan qué hacer, sino que además los humanos entiendan qué están haciendo y por qué. Documentar un programa no es sólo un acto de buen hacer del programador por aquello de dejar la obra rematada. Es además una necesidad que sólo se aprecia en su debida magnitud cuando hay errores que reparar o hay que extender el programa con nuevas capacidades o adaptarlo a un nuevo escenario.