Javadoc

Para escribir un comentario (ej : “hola mundo”) en javadoc, se escribe antes del método, atributo o clase que se desea documentar y se escribe de la siguiente forma

```
/**
* Hola mundo
*/
```

Estructura

Una estructura de los comentarios puede ser la siguiente :

- colocar primero una explicación de lo que se esta documentando (ej : “esta clase representa ...”)

- colocar las pre y post condiciones (en una clase se puede especificar restricciones de la misma)

- colocar los parámetros (si se aplica)

- colocar el retorno del método (si se aplica)

- colocar las excepciones que arroja (si se aplica)

- colocar links a otros javadoc (si se aplica)

- colocar autor (generalmente solo se pone en la clase)

- colocar versión (si se aplica, generalmente solo se pone en la clase)

etiquetas de valores

@param (1 por parámetro)

permite especificar un parámetro del método que se está documentando

forma de uso : `@param <nombre_parámetro> : <explicación del parámetro> : <tipo_parámetro>`

ej : `@param pos : posición del elemento a recuperar : int`

@return

permite especificar el valor retornado del método que se está documentando

forma de uso : `@return <explicación del valor retornado> : <tipo_retorno>`

ej : `@return el elemento en la posición pos : Object`

@throws (1 por excepción)

permite especificar una excepción que puede ser arrojada por el método que se está documentando

forma de uso : `@throws <Nombre Excepción> : <condición para arrojar a la misma>`

ej : `@throws IndexOutOfRangeException : si pos < 0 ó pos > longitud - 1`

@see (1 por link)

permite linkear (crear un acceso directo) al javadoc de una clase o método

forma de uso :

`@see <clase>`	linkea a una clase
`@see <clase#método>`	linkea a un método
`@see<clase#método(<parámetros>)>`	linkea a un método especificando sus parámetros

ej :

```
@see List
@see List#add
@see List#add(Object)
```

El tercer ejemplo es útil cuando hay un mismo método definido varias veces con distintos argumentos, el segundo ejemplo puede causar confusión y no puede linkear al método deseado

por ejemplo en la clase List hay dos métodos add :

add(Object)
add(int, Object)

@author

permite especificar el autor de un método o clase

forma de uso : `@author <nombre autor>`

ej : `@author Simón Gutiérrez`

@version

permite especificar la versión de un método o clase

forma de uso : `@version <nro>`

ej : `@version 2.1`

@code

Especifica algo que hace referencia a código como variables, clases, etc. Para variables solo se usa luego de que una variable fue definida, es decir :

INCORRECTO!
```
@param {@code pos} : la posición del elemento a insertar : {@code int}
```

CORRECTO!
```
@param pos : la posición del elemento a insertar : {@code int}
…
@throws IndexOutOfRangeException : sii {@code pos} <0 | {@code pos} > {@code length()}
```

forma de uso : `{@code <código>}`

ej : `{@code Integer}`

ej : `@param pos : la posición del elemento a insertar : {@code int}`

etiquetas de estilos

<p>

Delimita un párrafo

forma de uso : `<p> <texto> </p>`

ej : `<p> Esta clase representa una lista ordenada </p>`

Permite listar elementos

forma de uso : `<li> <item_texto> </li>`

ej :

```
<li> El elemento no está </li>
<li> El elemento está al principio </li>
<li> El elemento está al final </li>
<li> El elemento esta entre el principio y el final </li>
```


Indenta un bloque de texto

forma de uso : `<ul> <texto></ul>`

ej :

```
<p> Las condiciones que se deben cumplir son : </p>
<ul>
<li> Estar ordenada </li>
<li> No tener elementos repetidos </li>
</ul>
```

<hr>

Dibuja una linea horizontal

forma de uso : `<hr>`

<pre>

Permite escribir código

forma de uso : `<pre> <código> </pre>`

ej :

```
<p>
Si se desea crear un álgebra con mas restricciones, simplemente se debe extender
esta clase, crear un método que verifique la condición deseada y sobreescribir
la función verificar para que quede de la forma
</p>

<pre>

      protected boolean verificar() {
            return super.verificar() && verificarMiAlgebra();
      }

</pre>
```

Ejemplos de javadoc

Comentario de clase Algebra

```
/**
 * <p>
 * Clase que define un álgebra de la siguiente forma :
 * <ul>
 * <li>-Un conjunto no vacío (finito)</li>
 * <li>-Un conjunto de operaciones (cerradas sobre el conjunto soporte)</li>
 * </ul>
 * <p>
 *
 * Esta clase representa menos que un semigrupo, ya que no se verifica asociatividad
 *
 * <p>
 *
 * Si se desea crear un álgebra con mas restricciones, simplemente se debe extender esta
 * clase, crear un método que verifique la condición deseada y sobreescribir la función
 * {@code verificar} para que quede de la forma
 *
 * <p>
 *
 *
```

```
 * <pre>
 *
 *     protected boolean verificar() {
 *            return super.verificar() && verificarMiAlgebra();
 *     }
 *
 * </pre>
 *
 * <p>
 *
 * Si se desea crear un SemiGrupo por ejemplo, el método verificarMiAlgebra() debería
 * comprobar que solo hay una función binaria asociativa
 *
 * @author Simon Emmanuel Gutierrez Brida
 * @version 0.5
 *
 */
```

Comentario de la clase SemiGrupo

```
/**
 * Clase que extiende a {@code Algebra} y representa un semigrupo
 *
 * <p>
 *
 * Las condiciones para ser semigrupo son :
 *
 * <p>
 * <ul>
 * <li> Ser álgebra </li>
 * <li> Tener una única función binaria</li>
 * <li> La función debe ser asociativa</li>
 * </ul>
 * </p>
 *
 * @see Algebra
 * @author Simon Emmanuel Gutierrez Brida
 * @version 0.1
 *
 */
```

Comentario constructor clase Algebra

```
/**
* Constructor de la clase
*
*@param conjunto el conjunto soporte : {@code TipoConjuntoSoporte}
*@param funs : las funciones asociadas al álgebra :{@code Funciones<TipoConjuntoSoporte>}
*@throws ExcepcionAlgebraInvalida : sii el conjunto es vacío o si alguna función no es
*cerrada
*/
```

Comentario del método obtener de BagDiccionario

```
/**
* <li>pre : Bag-Diccionario no vacío & clave existe</li>
* <li>pos : devuelve todos los elementos que comparten la clave : {@code clave}</li>
* <hr>
* @param clave : la clave de los elementos a obtener : {@code Comparable}
* @return todos los elementos con la clave ({@code clave}) : {@codeTadListaDinList}
* @throws ExcepcionBagDiccionario : cuando no se cumple la pre condición
* @see TadListaDinList
*/
```