Fe de erratas y comentarios de la Práctica 3

Errata 1

El código inicial –que os teníamos que haber proporcionado– de `removeItem(Item item)` de la clase `Tank` debería haber sido (en amarillo la modificación):

```
public void removeItem(Item item) throws Exception{
        if(!items.remove(item)) {
         throw new Exception("[ERROR] The item does not exist in the tank");
        }
        item.setTank(null);
}
```

Esto también ocurre con la solución del Ejercicio 2 de la Práctica 2.

Errata 2

Hay una errata con los constructores de `Animal` y `Fish`. Éstos no reciben el parámetro `status` puesto que su valor se calcula en función del valor de `energy` (valor que sí se pasa como argumento a los dos constructores).

Errata 3

En el método `addItem` de `Tank` que os hemos proporcionado aparece:

```
item.setTank(this);
items.add(item);
```

Y se debería cambiar el orden de estas dos instrucciones. Es decir:

```
items.add(item);
item.setTank(this);
```

Esto también ocurre con la solución del Ejercicio 2 de la Práctica 2.

Errata 4

El enunciado dice que `Food` tiene dos atributos propios, en verdad, son tres: `speed`, `eaten` y `energy`.

Errata 5

Cuando se instancia un objeto de tipo `Snail`, el constructor de esta clase asigna los valores de `speed`, `requiredFoodQuantity` y `thresholdReverse` a `0.01`, `0.1` y `0.00000003`, respectivamente.

Errata 6

Todas las apariciones de `thresholdReverse`, ya sea como parámetro o atributo, debe declararse como `double`, no como `float`. Si está como float no es un gran problema, si se hace un *cast* correctamente.

Comentario 1

En el Ejercicio 2 se dice que tanto `Animal` como `SubmarineToy`, `Kelp` y `Food` son subclases de `Item`. De hecho, esto se dice en el primer párrafo del Ejercicio 2. Además el ejercicio se llama "Especializando los ítems".

Comentario 2

En los ejercicios 2 y 3 no damos fichero `Check.java`. Ya estamos en la Práctica 3 y queremos que hagáis este trabajo (no hace falta que entreguéis los ficheros de prueba que hagáis). Como fichero base, podéis coger el fichero `Check.java` del Ejercicio 1 y modificarlo para comprobar los otros dos ejercicios. En la Práctica 4 explicaremos, muy superficialmente, el uso de tests unitarios usando JUnit.

Comentario 3

Los métodos `moveUp`, `moveDown`, `moveRight` y `moveLeft` deben, cada vez que se les llama, desplazar el elemento en cuestión hacia la dirección que corresponde "`speed`" unidades. Es decir, si un pez tiene una velocidad de 5 y está en la posición x = 10, entonces si se llama a `moveRight`, ahora el pez estaría en x = 15.