PROGRAMACIÒN ORIENTADA A OBJETOS

TALLER No. 2 JAVA	PROGRAMACIÓN ORIENTADA A OBJETOS

Profesor: Jaime Alberto Guzmán Luna

Contenido del taller:

1. Clases y objetos
2. Atributos y Métodos

Ejercicio 1 – Clases y Objetos. (Preparación para el quiz)

a) Crear un nuevo proyecto de Eclipse llamado Taller2.
b) Agregar al proyecto de Eclipse, las siguientes clases:

```
1  public class Animal {
2        String nombre;
3        String genero = "M";
4        double peso;
5        Animal pareja;
6        static int totalAnimales;
7
8        void setPareja(Animal pareja) {
9              this.pareja = pareja;
10       }
11
12       Animal tenerHijo (String nombre) {
13             Animal hijo = new Animal();
14             totalAnimales++;
15             hijo.nombre = nombre;
16             hijo.peso = 1;
17             double random = Math.random();
18             if (random < 0.5)
19                   hijo.genero = "F";
20             else
21                   hijo.genero = "M";
22
23             return hijo;
24       }
25
26       Familia procrear(String nombreHijo) {
27
28             if (pareja == null) {
29                   System.out.println(this.nombre + " no tiene pareja");
30                   return null;
31             }
32
33             if (pareja.genero.equals(genero)) {
34                   System.out.println(nombre + " y " + pareja.nombre + "
```

```
son del mismo genero");
35                      return null;
36              }
37
38              System.out.println(nombre + " y " + pareja.nombre + " van a
tener un hijo");
39
40              Familia f = new Familia();
41
42              if (this.genero.equals("M")) {
43                      f.mama = pareja;
44                      f.papa = this;
45              } else {
46                      f.mama = this;
47                      f.papa = this.pareja;
48              }
49
50              Animal hijo = f.mama.tenerHijo(nombreHijo);
51              f.hijo = hijo;
52              System.out.println(hijo.nombre + " ha llegado al mundo");
53              return f;
54      }
55
56      static void morir (Animal animal) {
57              System.out.println(animal.nombre + " ha muerto");
58              totalAnimales--;
59      }
60
61      public String toString() {
62              return "Me llamo " + nombre + " y peso " + peso;
63      }
64  }
```

```
1   public class Familia {
2       Animal papa;
3       Animal mama;
4       Animal hijo;
5
6       void imprimir() {
7               String genero;
8               if (hijo.genero.equals("M"))
9                       genero = "masculino";
10              else
11                      genero = "femenino";
12              System.out.println(papa.nombre + " es el papá, "
13                      + mama.nombre + " es la mamá, y " + hijo.nombre
14                      + " es el hijo de genero " + genero);
15      }
16  }
```

```
1   public class FamiliaAnimales {
2       public static void main(String args[]) {
3               Animal animal1 = new Animal();
```

```
4               Animal.totalAnimales++;
5               Animal animal2 = new Animal();
6               Animal.totalAnimales++;
7
8               animal1.nombre = "Cebra";
9               animal1.genero = "F";
10
11              animal2.nombre = "Caballo";
12              animal2.peso = 98.0;
13
14              System.out.println(animal1.nombre + " se va a casar con " +
                                   animal2.nombre);
15              animal1.setPareja(animal2);
16              animal2.setPareja(animal1);
17
18              Familia familia = animal1.procrear("Cebrallo");
19
20              if (familia != null) {
21                    familia.imprimir();
22                    System.out.println(familia.hijo);
23              } else {
24                    System.out.println("No se pudo formar familia");
25              }
26              System.out.println("Total de animales: " +
                                   Animal.totalAnimales);
27              Animal.morir(animal2);
28              System.out.println("Nuevo total de animales: " +
                                   Animal.totalAnimales);
29        }
30  }
```

Responder:

a) ¿Cuántas clases se están definiendo en este ejercicio?
b) ¿En cuál clase se define el programa principal? Corra el programa principal.
c) ¿Cuántos objetos de la clase `Animal` se están creando en la clase que define el programa principal?
d) ¿Cuáles objetos se están creando de la clase `Animal` en la clase que define el programa principal?
e) ¿Cuáles atributos tiene la clase `Animal`?
f) ¿Cuáles atributos de la clase `Animal` están haciendo referencia a tipos primitivos?
g) ¿Cuáles atributos de la clase `Animal` están haciendo referencia a objetos?
h) ¿Con qué valor se inicializa el atributo `pareja` de la clase `Animal`?
i) ¿Cuál es el nombre que tienen los objetos `animal1` y `animal2` **antes de la línea 7** en la clase `FamiliaAnimales`?
j) ¿Cuál es el peso de `animal1` en la clase `FamiliaAnimal`?
k) Dibuje el estado de memoria luego de establecer como pareja del caballo a la cebra y viceversa.
l) ¿Cuál es el género del `animal2` en la clase `FamiliaAnimales`?

m) Qué sucede si…
   - …se comenta la línea 15 de la clase `FamiliaAnimales`
   - …se comenta la línea 16 de la clase `FamiliaAnimales`
   - …se comentan las líneas 15 y 16 de la clase `FamiliaAnimales`

n) ¿En el contexto de cuál objeto se está ejecutando el método procrear cuando es invocado en la línea 18 de la clase `FamiliaAnimales`?

o) ¿Qué sucede si al atributo pareja de la clase `Animal` se le coloca el modificador `final`?

p) ¿Se puede eliminar el modificador `static` del método morir de la clase `Animal` sin que se afecte el funcionamiento del programa?

q) ¿Qué sucede si modifica la línea 3 de la clase `Animal` como se indica a continuación?

Línea original: `String`
```
genero = "M";
```

Nueva línea: `static final String`
```
genero = "M";
```

r) ¿Por qué no es necesario asignarle el valor inicial al atributo `totalAnimales` de la clase `Animal`? Explique.

s) ¿Por qué razón, si se reemplaza la línea 28 de la clase `FamiliaAnimales` por la línea indicada a continuación, el resultado se mantiene igual?

Línea original:
```
System.out.println("Nuevo total de animales: " + Animal.totalAnimales);
```

Nueva línea:
```
System.out.println("Nuevo total de animales: " + animal1.totalAnimales);
```

t) ¿Por qué razón, no se afecta el resultado, si se reemplaza la línea 27 de la clase `FamiliaAnimales` por la indicada a continuación?

Línea original:
```
Animal.morir(animal2);
```

Nueva línea:
```
animal1.morir(animal2);
```

u) ¿Cuántos métodos tiene la clase `Animal`?

v) ¿Cuál es el tipo de retorno de los métodos `procrear()` y `tenerHijo()` de la clase `Animal`?

w) ¿Al método `setPareja()` de la clase `Animal` se le está pasando el parámetro por valor o por referencia?

x) ¿A quién está haciendo referencia la variable `this` de la línea 29 de la clase `Animal` cuando se ejecuta el programa principal? ¿Podría omitirse el uso de la variable `this` en este caso?

y) ¿Por qué no se afecta la ejecución del programa si se reemplaza la línea 38 de la clase `Animal` por la indicada a continuación?

Línea original:

```
System.out.println(nombre + " y " + pareja.nombre + " van a tener un
hijo");
```

Nueva línea:

```
System.out.println(this.nombre + " y " + this.pareja.nombre + " van a
tener un hijo");
```

z) ¿Podría eliminarse el modificador `static` del método `morir()` de la clase `Animal` sin que se afecte el programa?

aa) ¿A quién hace referencia la variable `this` de las líneas 44 y 46 de la clase `Animal` cuando se ejecuta el programa principal? ¿Por qué es necesario?

bb) Modifique el método imprimir de la clase `Familia` para que sea un método de clase.

cc) ¿Por qué razón es útil que el atributo `totalAnimales` sea un atributo de clase y no un atributo de ejemplar?

dd) ¿Se puede colocar el modificador `static` al método `tenerHijo()` de la clase `Animal` sin que se afecte el funcionamiento del programa? Explique.

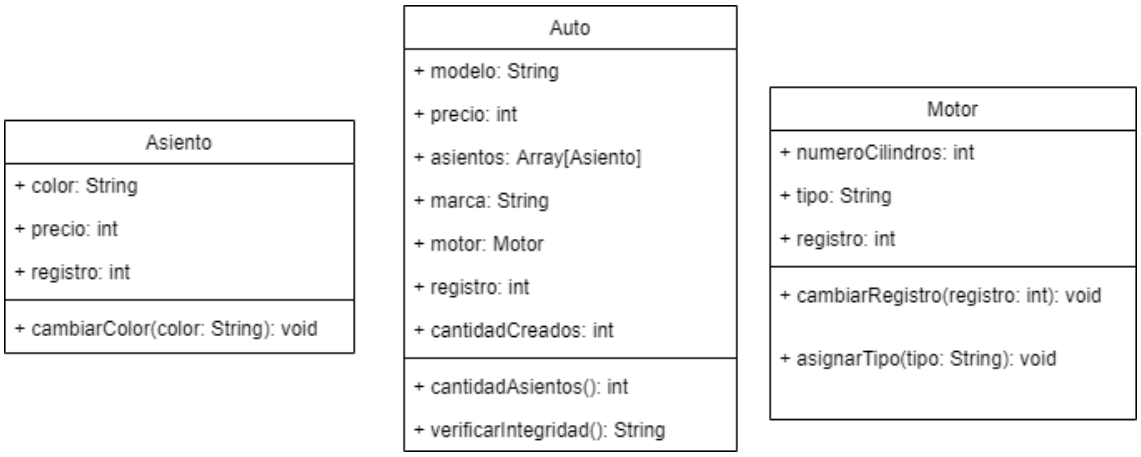
ee) ¿Qué hace el método `toString()` de la línea 59 de la clase Animal?

Ejercicio 2 – GitHub

Enlace entrega: https://classroom.github.com/a/SfEP68Z0

PROBLEMA – Componentes de un Auto

Se le ha contratado para el diseño de la estructura de componentes de un Auto, y se le entrego un diagrama que mostraba un poco las Clases, los atributos y métodos, además se le proporciono unas ciertas instrucciones.



Cree las clases Auto, Asiento y Motor, con sus respectivos atributos y métodos…

Tener en cuenta para la Clase Asiento que:

- El método de instancia cambiarColor(), recibirá un argumento String que será el valor a asignar al atributo color del objeto, tenga en cuenta que los únicos valores

permitidos para cambiar el color serán rojo, verde, amarillo, negro y blanco, cualquier otro color no cambiara el color del Asiento.

Tener en cuenta para la Clase Auto que:

- cantidadCreados es un atributo de clase
- El método de instancia cantidadAsientos() retornara la cantidad de asientos que efectivamente sean objetos Asiento en la lista del objeto Auto.
- El método verificarIntegridad(), se encargara de revisar que el atributo registro de Motor, Auto y Cada Asiento sean el mismo, esto para ir en contra de la piratería de piezas. En caso de encontrar que un Asiento, el Auto o el Motor tiene un registro diferente al de los demás retornara el mensaje “Las piezas no son originales” en caso contrario, retornara “Auto original”

Tener en cuenta que para la Clase Motor que:

- El método cambiarRegistro(), recibirá como argumento un int que cambiara el numero del registro del objeto
- El método asignarTipo(), recibirá como argumento un String que cambiara el tipo de motor, este valor solo podrá ser cambiado por el valor electrico o gasolina, en caso contrario no modificara el valor

Nota: Tenga en cuenta que cuando una lista de objetos es creada cada posición se le asigna el valor null.

IMPORTANTE: Agregue a la primera línea de cada archivo java lo siguiente:
package test;

Además, pegue las clases creadas a la carpeta src/test/java/test. Esto con el fin de que se pueda autocalificar y entregar de manera efectiva la actividad.