PROGRAMACIÒN ORIENTADA A OBJETOS

TALLER No. 6	PROGRAMACIÓN ORIENTADA A OBJETOS

Profesor: Jaime Alberto Guzmán Luna

Contenido del taller:

1) Herencia
2) Constructores
3) Super en Python
4) Enum en Python

---

1. EJERCICIO

¿Qué imprime el siguiente código cuando se crea una instancia de Cuarto?

```
1  class Primero{
2
3        Primero(){
4              m();
5        }
6
7        void m(){
8         System.out.println("m desde Primero");
9     }
10 }
11
12 class  Segundo extends Primero{
13       void m() {
14             System.out.println("m desde Segundo");
15       }
16 }
17 class  Tercero extends Segundo{
18       Tercero(){
19             super.m();
20       }
21
22       void m() {
23                System.out.println("m desde Tercero");
24       }
25 }
26 class  Cuarto extends Tercero{
27       void m() {
28                System.out.println("m desde Cuarto");
29       }
30 }
```

2. EJERCICIO

PROGRAMACIÒN ORIENTADA A OBJETOS

Identifique los errores en el siguiente listado de código y explique brevemente como los resolvería.

```
1  public class Objtaller6 {
2
3        public static void main (String[] args) {
4
5          Persona p1 = new Persona("Pepe", 213);
6          Estudiante e1 = new Estudiante(91, "Sara", 1431);
7          Profesor pf1 = new Profesor("Calculo", "Roger", 412);
8          Persona p2 = new Estudiante(87, "Sofia", "4");
9          Persona e2 = new Profesor("Bases", "Adrian", 231);
10
11     }
12 }
13
14 class Persona {
15
16     private String nombre;
17     protected long cedula;
18
19     public Persona(String nombre, long cedula) {
20         this.nombre = nombre;
21         this.cedula = cedula;
22     }
23
24     public final int getEdad() {
25         return 18;
26     }
27 }
28
29 class Profesor extends Persona{
30
31     String materia;
32     private int cedula;
33
34     public Profesor(String materia, String nombre, int cedula) {
35         super(nombre, cedula);
36         this.materia = materia;
37     }
38
39     public Profesor() {
40         this("Fisica");
41         super("Juan", 20123);
42     }
43
44     public Profesor(String materia) {
45         this(materia, "Juan", 20123);
46     }
47
48     public int getCedula() {
49         return cedula;
50     }
```

```
51
52      public long getCedulaS () {
53          return super.cedula;
54      }
55
56  }
57
58  class Estudiante extends Persona {
59
60      int codigo;
61
62      public void combinarNombre(String n) {
63          super.nombre += n;
64      }
65
66      public final int getEdad() {
67          return 20;
68      }
69
70  }
```

3. EJERCICIO PYTHON

```
1   from enum import Enum
2
3
4   class Nombre(Enum):
5       PEPE = "Pepe"
6       JAIME = "Jaime"
7       HERNAN = "Hernan"
8
9
10  class Persona:
11      numPersonas = 0
12
13      def __init__(self):
14          Persona.numPersonas += 1
15
16
17  class Padre(Persona):
18      pass
19
20
21  class Abuelo(Padre):
22      def __init__(self, nombre, hijo):
23          self._nombre = nombre
24          self._hijo = hijo
25
26      def getNombre(self):
27          return self._nombre
28
29
30  class Bisabuelo(Abuelo):
```

```
31      def __init__(self, nombre, hijo, edad):
32          self._edad = edad
33          self._nombre = nombre
34          super().__init__(Nombre.PEPE.value, hijo)
35
36
37  if __name__== "__main__":
38
39      p1 = Persona()
40      p2 = Padre()
41      p3 = Abuelo(Nombre.JAIME.value, p2)
42      p4 = Bisabuelo(Nombre.HERNAN.value, p3, 89)
43
44      print(Persona.numPersonas)
45      print(p4.getNombre())
```

En base al código anterior responda:

1. ¿Cuál es el constructor de la clase Padre?
2. ¿Por qué al imprimir el número de personas su resultado no es 4? ¿Como solucionaría usando super() para que se cuenten los objetos de las clases que heredan de Persona?
3. ¿Por qué el imprimir el nombre de p4, este no es “Hernan” si se le asigno este al objeto?
4. ¿Qué particularidad se observa al usar Enum en el Código? ¿Qué ocurre si se quita el llamado al atributo value, en la línea 34, 41 y 42?

4. EJERCICIO PRACTICO GITHUB EN JAVA

Ingrese al siguiente enlace para aceptar y empezar el ejercicio:
https://classroom.github.com/a/5-Mz2ekU

El fabricante de autos

Luego de tomarnos un respiro y seguir adelante, incursionamos en un nuevo dominio, los vehículos.

Queremos entonces crear un software que lleve el control y modelado de la adquisición de autos por fabricantes, países, etc, para esto se le dieron ciertas indicaciones:

- Se le pide en primer lugar crear las clases Automóvil, Camion, Camioneta, Vehículo, País y Fabricante. Automóvil, Camion y Camioneta heredan de Vehículo. Todas las clases estarán en el paquete vehiculos.
- Para la clase País tendrá un atributo nombre, donde almacenará el valor del nombre del país.
- Para la clase Fabricante tendrá los atributos nombre y pais, donde almacenará el valor del nombre del fabricante y el país donde se le ubica a la fábrica.
- Para la clase Vehículo tendrá un atributo placa donde se almacena el texto de la placa del vehículo, un atributo puertas donde almacena la cantidad de estas, un atributo velocidadMaxima donde se especifica el valor máximo que alcanza el vehículo en su velocidad, un atributo nombre donde almacena el texto con el que llaman al vehículo, el atributo precio almacena la cantidad necesaria para comprar el vehículo, el atributo peso almacenara el valor del peso del vehículo, el atributo traccion almacenará el texto del

tipo de tracción que usa el vehículo, el atributo fabricante que almacena el valor del fabricante del auto, por último el atributo CantidadVehiculos el cual almacena el valor entero de la cantidad de vehículos que se han creado hasta el momento.

- Para la clase Automóvil esta tendrá un atributo puestos donde almacenará la cantidad de puestos o asientos para personas, que el auto tiene permitido movilizar. Esta clase para los atributos puertas, velocidadMaxima y traccion tomaran los siguientes valores cuando se crea un nuevo Automóvil 4, 100 y ”FWD” respectivamente. El resto de los atributos se recibirán como parámetros.
- Para la clase Camioneta esta tendrá un atributo volco donde se especifica si la camioneta tiene o no tiene volcó (use el atributo boolean). Igualmente, el valor para los atributos velocidadMaxima y traccion tomarán los siguientes valores cuando se crea una nueva Camioneta 90 y ”4X4” respectivamente. El resto de los atributos se recibirán como parámetros.
- Para la clase Camion esta tendrá un atributo ejes donde se almacena la cantidad de ejes que tiene el camión. Además, el valor para los atributos puertas, velocidadMaxima y traccion tomarán los siguientes valores cuando se crea un nuevo Camión 2, 80 y ”4X2” respectivamente. El resto de los atributos se recibirán como parámetros.

Nota1: Cree los métodos get y set para los atributos de las clases. Tenga en cuenta que cuando un valor es de tipo boolean el estándar para obtener el valor(get) se representa como isNombreAtributo, y cree los constructores con el orden de los parámetros que sigue el mismo orden de los atributos.

Nota2: Tenga en cuenta incluir los atributos de las clases que hereda en su constructor.

- Cree un método llamado vehiculosPorTipo en la clase Vehículo que retornara la cantidad de objetos creados por cada subclase de Vehículo. Este devolverá el valor siguiendo el siguiente formato:

  “Automoviles: #
  Camionetas: #
  Camiones: #”

  # es el número de objetos por cada subclase.
- Por último, cree las siguientes funcionalidades:
  - Cree un método llamado paisMasVendedor que se encargue de retornar el país que ha vendido más unidades de vehículos entre todos los demás. (El que ha creado instancias de vehículos cuyo fabricante es de un país)
  - Cree un método llamado fabricaMayorVentas que se encargue de retornar la fábrica que ha vendido más unidades de vehículos entre todas las demás. (El que ha creado instancias de vehículos asociado a un fabricante)

  Nota3: Piense en que clase deberían estar ubicados estos métodos.

Nota4: Puede crear más atributos y métodos diferentes a los que se mencionó anteriormente, si lo desea, siempre y cuando estos no se incluyan en los parámetros de los constructores y de las funcionalidades/métodos.