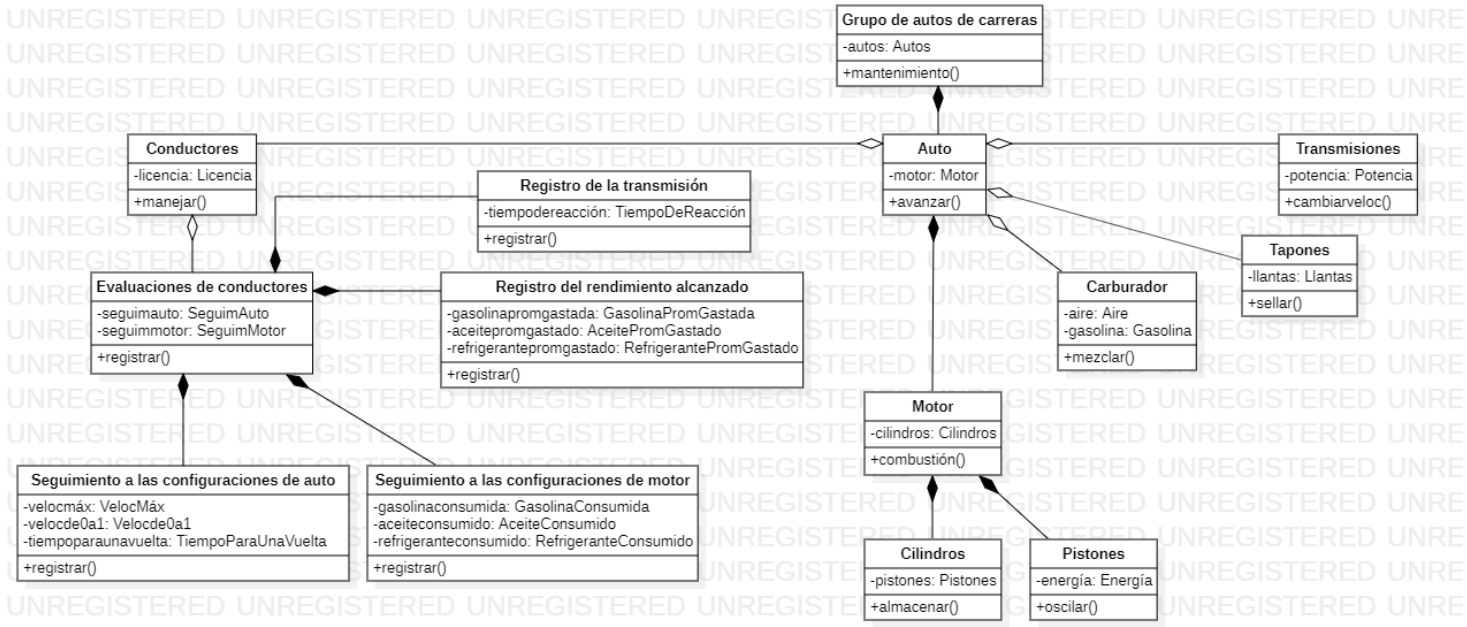
UNIVERSIDAD AUTÓNOMA METROPOLITANA
Casa abierta al tiempo
Unidad Iztapalapa

Práctica semana 8
Trimestre 21-O

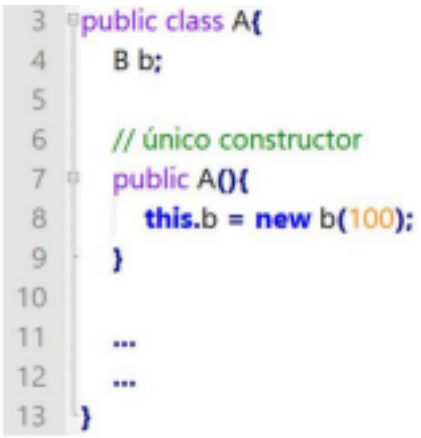
Análisis y diseño de sistemas de cómputo

Prof. Orlando Muñoz Texzocotetla

1. **Una compañía de mantenimiento a un grupo de autos de carreras. Estos autos utilizan algunos de los nuevos motores de 8 cilindros y las nuevas transmisiones. Una vez que los motores son ensamblados, los pistones, el carburador y los tapones no pueden cambiarse con otros motores debido a los cambios que causan altas temperaturas. Se desea mantener el registro del rendimiento alcanzado por cada motor en cada auto, y de cada transmisión en combinación con cada motor. Los conductores dan su evaluación después de manejar su correspondiente auto. Para ello se necesita de un sistema que lleve el seguimiento a las configuraciones de cada auto (y de cada motor), así como de las evaluaciones dadas por los conductores. Crear un diagrama de clases UML que representa la información dada en el párrafo anterior. El diagrama debe tener al menos 5 clases, con sus correspondientes atributos y métodos, así como sus correspondientes relaciones entre ellas: asociación, todo-parte y herencia.**



2. **Indique si el siguiente código en Java representa una relación de agregación o de composición, y explique por qué.**



```
3  public class A{
4      B b;
5
6      // único constructor
7      public A(){
8          this.b = new b(100);
9      }
10
11     ...
12     ...
13 }
```

Representa una relación de composición, ya que para que sea agregación, el objeto en cuestión debe de ser agregado mediante un método siendo de otra clase, pero, en este caso es agregado mediante un único constructor. Por lo tanto, el objeto en cuestión es de dentro de la misma clase lo que hace que la relación sea de composición.

5. **Se desea crear una clase llamada vector3D, y se decide que esta clase derive de otra clase llamada vector2D. La explicación de esa decisión es que un vector 3D es un vector 2D pero con una dimensión más (añadida). Expliqué por qué esa decisión (y explicación) es un error conceptual de herencia en programación orientada a objetos.**

Para este problema debemos saber con claridad que es la **herencia** en **OOP** (programación orientada a objetos).

La ***herencia*** es el mecanismo en Java por el cual una clase permite heredar las características (atributos y métodos) de otra clase.

En el lenguaje de Java, una clase que se hereda se denomina superclase. La clase que hereda se llama subclase. Por lo tanto, una ***subclase*** es una versión especializada de una ***superclase***. Hereda todas las variables y métodos definidos por la superclase y agrega sus propios elementos únicos.

Dada la información anterior entendemos que la clase vector2D es nuestra superclase y el texto nos dice que se desea crear una clase llamada vector3D la cual deriva de la clase vector2D por lo que la podemos entender como nuestra subclase.

Un vector 3D está formado por 3 variables (x, y, z) y el vector 2D por 2 variables (x, y), pero el argumento nos indica que *“un vector 3D es un vector 2D, pero con una dimensión más (añadida)”*.

Esto anterior es incorrecto dado que la superclase debe contener atributos y métodos que sean utilizados por la subclase y esta subclase no puede añadirse con más dimensiones o atributos dado que se utilizara un extends que llame a nuestra superclase la cual nos dará los atributos (x, y) y no nos dará la variable (z) para tener un vector3D por lo que nuestra clase vector3D estaría incorrecta.