OCPJP Ing. Adrian Gomez

Assignment # 3 OO Programming

1.- Car Shop(1.5 puntos)

Eres seleccionado para desarrollar una aplicación para un concesionario, primer paso sería pensar en cómo representar la lógica de negocios. Debes crear una clase llamada Car, la cual representará la base de todos los carros del concesionario. La clase Car debe tener los siguientes atributos : price(double) y color(String).

Luego debes crear una clase para representar las camionetas(Truck) . La diferencia entre las camionetas y los carros es que para las camionetas consideramos el atributo peso(weigth(double)). La otra diferencia seria que si el peso de la camioneta es menor a 2000 tendrías un descuento del 20%, de lo contrario tendrás un descuento de 10%.

También se debe poder representar los carros de lujo(LuxuryCar). Los carros de lujo tendrán otro atributo propio que serían la cantidad de accesorios(accessories(int)), cada accesorio aumentará el precio del carro en un 10%(sobre el precio base).

Ejemplo:

a)Car ford = new Car(1000.0,”red”);
ford.getPrice()// debe ser 1000.0
b) Car fordExplorer = new Truck(2000.0,”blue”,1900);// debería ser 1600
c) Car ferrari = new LuxuryCar(3000.0,”black”,5);// debería ser 4500

Debes enviarme las clases Car, Truck y LuxuryCar.

2.-Shapes(1.5 puntos)

Debes diseñar un modelo que permita representar las figuras geométricas de una forma que sea mantenible y extensible. Debes poder permitirle a otros programadores poder agregar nuevas figuras geometricas al programa sin modificar el código de las demás figuras.

El punto de las figuras geométricas que vamos a considerar es el área, entonces todas las figuras deben estar obligadas a tener una implementación de un método que te devuelva el área.

Ejemplo el triángulo para calcular el área necesitamos la base y la altura; es decir, que debes crear una clase para representar el triángulo con los atributos base y altura.

Debes crear la implementación de triangulo, cuadrado y rectángulo. Si yo quiero crear una nueva figura como por ejemplo el pentágono debes permitirlo facilmente.

En este ejercicio serán evaluados aspectos de diseño que antes no habían sido considerados como: nombre de las variables, encapsulación, etc.

Debes enviarme la clase Triangle, Square y Rectangle aparte de la clase Padre o Interfaz que hayas utilizado, en caso de que formen parte de su implementación. También debes en el correo justificar brevemente porque tu código es extendible.

3.-MovieDAO(2 puntos)

Se te ha asignado la tarea de crear una aplicación para llevar el control de las películas que se venden en una tienda. Otro programador fue encargado de desarrollar el modelo y tu trabajo es persistir el estado de esos objetos, la empresa todavía no ha podido decidir qué gestor de bases de datos utilizar pero no quiere retrasar tanto el desarrollo del proyecto, por lo cual se requiere que implementes una persistencia en memoria.

El otro programador pone a tu disposición la clase Movie y Actor las cuales **no debes modificar.** El equipo de programación se pone de acuerdo en una capa para abstraer la lógica de la persistencia y desarrollan una interfaz. Debes implementar todos los métodos que aparecen declarados en la interfaz.

Solo debes enviarme la clase de la implementación.

4.-RecommendingMovies(2 puntos OPCIONAL)

La aplicación para la tienda de películas en la que has trabajado ha tenido más éxito de lo esperado y su página web es constantemente visitada por sus clientes en busca de películas. Tu jefe quiere que le des una ayuda a los visitantes de la página para saber qué películas les pueden gustar.

Vamos a desarrollar una forma muy sencilla de recomendar una serie de películas a un cliente, tendrás acceso a un arreglo de clientes. La clase Customer tiene un arreglo de películas las cuales son las que ha comprado antes, debes buscar al cliente que tenga los gustos más parecidos a él; es decir, tengan en común la mayor cantidad de películas compradas .

Una vez encontrado el cliente que más se parece le vamos a recomendar todas las películas que él no ha comprado. Si te encuentras con que hay más de un cliente que coincide con sus características deben recomendar las películas de todos los que coincidan.

Si el cliente no ha comprado ninguna película o no hay coincidencias debes regresar un arreglo vacío.

Debes tener cuidado de que el cliente que coincida no sea el mismo al que le vas a recomendar las películas.

Si el cliente que coincide tiene exactamente las mismas películas debes buscar el segundo que coincida.

Se te dará una interfaz llamada CustomerDAO con un método findAll, el cual te dará todos los clientes de la tienda. Debes crear una implementación para poder probar.

Debes implementar la interfaz RecommendingMovies y enviar la implementación.

ENTREGA: Antes del sábado 30 de noviembre.