**Ejercicio**

**Lea el documento atentamente antes de iniciar la creación de la solución.**

**ENUNCIADO**

Un taller mecánico ha decidido realizar una aplicación, que permita agilizar la atención y el cobro de los servicios realizados a cada uno de los vehículos que reciben en sus dependencias.

En la actualidad, el taller recibe 2 categorías o tipos de vehículos: **Automóviles** y **Motocicletas**.

Algunos datos adicionales a considerar para cada uno de estos tipos son: los **Automóviles** tienen cantidad de puertas y capacidad del maletero (lts) y las **Motocicletas** tienen un estilo (deportiva, cross, etc) y medida del manillar.





Determine la **SUPERCLASE** y sus **SUBCLASES**, que permitan resolver el problema planteado considerando además, los siguientes requisitos:

* La SUPERCLASE debe considerar:
  + Un mínimo de **4** atributos.
  + Debe ser abstracta.
  + Un método abstracto que permita obtener el total del cobro (**totalPorPagar**), considerando los impuestos y descuentos.
  + Implementar la Interface que se describe más abajo.
* Las SUBCLASE deben considerar:
  + 2 atributos cada una.
  + No deben tener subclases.
  + Debe existir sobreescritura en un método por lo menos.
* Debe existir sobrecarga en un método por lo menos.
* Calcular el **descuento** (**obtenerDescuento**), según el valor que determine el usuario por parámetro.
* Debe implementar una **INTERFACE** en la súper clase con 3 porcentajes:
  + **IVA**: 19%
  + **DESCUENTO**\_AUTOMOVIL: 5%
  + **DESCUENTO\_MOTOCICLETA**: 10%
  + Métodos:
  + Calcular el total a pagar (**totalConIVA**), agregando al costo de los servicios el impuesto.
  + Calcular el descuento (**obtenerDescuento**), según el tipo de automóvil.
* Debe existir una clase que tenga una colección de vehículos, el cual deberá permitir:
* Debe permitir agregar (**agregar**) vehículos. Debe validar que el vehículo agregado no exista (patente).
* Mostrar (**listar**) todos los vehículos (todos sus datos).
* Contar (**contabilizar**) los vehículos de cada categoría, o sea automóvil y moto.

Todas las clases deben tener los siguientes métodos implementados: constructores, accesadores y mutadores.

Desde la clase que contiene el main debe:

* Realizar un registro de ingreso para un vehículo y una moto.
* Mostrar el total a pagar para una patente.
* Listar todos los vehículos.
* Mostrar los vehículos según su tipo o categoría.

Recordar que se debe programar en java aplicando todas las convenciones aprendidas en clases y **encapsulando** la clase.

CREAR LA CARPETA **NombreApellido** EN EL DISCO DE RESPALDO, DENTRO DE ELLA CONSTRUIR SU PROYECTO. NO OLVIDE COLOCAR SU NOMBRE EN EL COMENTARIO JAVADOC CORRESPONDIENTE.

Rúbrica

|  |  |
| --- | --- |
| **PGY2121** | **Desarrollo de Software** |

|  |  |
| --- | --- |
| **Sigla** | **Leyenda** |
| TL | TOTALMENTE LOGRADO |
| ML | MEDIANAMENTE LOGRADO |
| PL | PARCIALMENTE LOGRADO |
| NL | NO LOGRADO |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Aspectos a Evaluar | TL  3 | ML  2 | PL  1 | NL  0 | Ptos |
| 1. Definir correctamente las clases en la herencia con sus atributos y métodos | Define los atributos, constructores, accesadores y mutadores correctamente con sus modificadores de acceso en la superclase y subclases | Define los atributos, constructores, accesadores y mutadores con pequeños errores | Define los atributos, constructores, accesadores y mutadores con algunos errores | No define los atributos, constructores, accesadores y mutadores |  |
| 1. Definir las restricciones de las clases | Define todas las restricciones de las clases: extends, abstract y final | Define las restricciones de las clases: extends, abstract y falta un final | Define las restricciones de las clases: extends, falta abstract y final | No define las restricciones |  |
| 1. Implementar sobreescritura y sobrecarga | Implementa la sobreescritura y sobrecarga | Implementa la sobreescritura y sobrecarga con un pequeño error | Implementa sobreescritura o sobrecarga | No define sobreescritura ni sobrecarga |  |
| 1. Definir el método abstracto e implementar en las subclases | Define el método abstracto e implementa en las subclases | Define el método abstracto e implementa en las subclases con pequeños errores | Define el método abstracto e implementa en las subclases con algunos errores | No implementa el método |  |
| 1. Implementar interfaz | Define la interfaz con sus contantes y métodos y es implementada por la superclase | Define la interfaz con sus contantes y métodos y es implementada por la superclase con pequeños errores | Define la interfaz con sus contantes y métodos y es implementada por la superclase con algunos errores | No implementa la interfaz |  |
| 1. Implementar colección | Crea la colección y sus métodos | Crea la colección y le falta un método | Crea la colección y le faltan dos método | No define la colección |  |
| 1. Aplicar polimorfismo | Aplica polimorfismo |  |  | No aplica polimorfismo |  |
| 1. Crear el main e invocar a todos los métodos | Crea objetos, invocando a todos los métodos | Crea objetos, invocando a tres métodos | Crea objetos, invocando a dos métodos | No crea los objetos ni invoca a más de un método |  |

|  |  |
| --- | --- |
| **Puntajes** | **Nota** |
| Puntaje Total: 24 | 7.0 |
| Puntaje: 14 | 4.0 |

Tabla

Descripción generada automáticamente