**EVALUACIÓN 1**

**ASIGNATURA: PGY2121 – DESARROLLO DE SOFTWARE**

|  |  |  |
| --- | --- | --- |
| **PUNTAJE MÁXIMO** | **27 puntos** | **NOTA** |
| **PUNTAJE OBTENIDO** |  |
| **FECHA** |  |
| **DURACIÓN** | **120 minutos** |

**INDICACIONES GENERALES**

La nota 4,0 se obtiene logrando un 60% del puntaje total.

Cuando termine la evaluación, comprima la carpeta donde ha dejado los trabajos de la prueba y siga las instrucciones del docente.

**LEA ATENTAMENTE el documento antes de iniciar el desarrollo.**

**ENUNCIADO**

Un taller mecánico ha decidido realizar una aplicación, que permita agilizar la atención de sus clientes. Para esto se requiere crear un sistema que almacene información de clientes y sus vehículos.

Determine 5 atributos para el cliente y 5 atributos para el vehículo (como mínimo), que permitan guardar la información de los clientes y sus vehículos. Cada vehículo tendrá 1 dueño. Debe aplicar colaboración y además debe cumplir con los siguientes requerimientos planteados por el dueño del taller, los que se deberán implementar en la clase vehiculo:

1. Debe existir un método que permita mostrar la **información** del vehículo.
2. Debe existir un método que permita realizar un **descuento** al total de los servicios prestados, según las siguientes las condiciones serán: Si el cliente es nuevo o lo(s) servicios prestados tienen un costo de $100.000 o más, se aplicará un descuento de x%. Para los servicios solo se necesita el monto a pagar.
3. Debe existir un método que permita enviar un **correo** avisando que el vehículo ya está listo para el pago y retiro. (simular mediante un mensaje por la consola)

Implemente las siguientes reglas de negocio:

Para la clase vehículos:

* El kilometraje debe estar en el rango de 0 a 500.000 km.
* La patente debe tener largo 6: CB3366 y no se permiten vacíos o en blanco.

Para la clase clientes:

* El nombre y rut del cliente no debe estar vacíos o en blanco.
* Código de área debe ser solo 2 dígitos.

Ambas clases deben tener **constructores, accesadores y mutadores.**

Desde una clase que contiene el main, crear un menú que permita:

1. Crear un cliente y vehículo.
2. Mostrar la información de un vehículo y su dueño (usar método)
3. Agregar monto de los servicios y aplicar el descuento.
4. Enviar Correo.
5. Salir.

Observaciones:

* Debe crear las variables de objeto fuera del menú.
* Los datos a guardar son para 1 cliente y 1 vehículo.
* El vehículo debe tener asociado un dueño.
* Solo se debe considerar el valor a total a pagar de los servicios.

**Recordar que se debe programar en java aplicando todas las reglas y convenciones vistas en clases.**

CREAR LA CARPETA en el disco D: **ApellidoNombre**

DENTRO DE ELLA CONSTRUIR SU PROYECTO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Aspectos a Evaluar** | **TL 3** | **ML 2** | **PL 1** | **NL 0** | **Pto.** |
| 1. Identifica correctamente los atributos de las clases | Identifica y define correctamente los atributos de las clases con los tipos de datos adecuados (primitivos y referenciados) y encapsulando los atributos | Define la mayoría de los atributos (considerando el tipo referenciado) con los tipos de datos correctos y encapsulando los atributos | Define sólo la mitad de los atributos | No define los atributos |  |
| 1. Definir correctamente los métodos constructores, accesadores y mutadores de las clases | Define correctamente los constructores, accesadores y mutadores | Define los constructores, accesadores y mutadores  con pequeños errores | Define los constructores, accesadores y mutadores con varios errores | No define los métodos |  |
| 1. Definir las reglas de negocio | Define correctamente las reglas de negocio | Define las reglas de negocio con pequeños errores | Define las reglas de negocio con varios errores | No define las reglas de negocio |  |
| 1. Definir el primer método customer | Define correctamente el método customer | Define el método customer con pequeños errores | Define el método customer con varios errores | No define el método customer |  |
| 1. Definir el segundo método customer | Define correctamente el método customer | Define el método customer con pequeños errores | Define el método customer con varios errores | No define el método customer |  |
| 1. Definir el tercer método customer | Define correctamente el método customer | Define el método customer con pequeños errores | Define el método customer con varios errores | No define el método customer |  |
| 1. Definir el main | Define el main correctamente,  crea los objetos y el menú | Define el main,  crea los objetos y el menú con pequeños errores | Define el main,  crea los objetos y el menú con varios errores | No define el main |  |
| 1. Invocar a métodos del main | Invoca a los tres métodos del main en forma correcta | Invoca a dos métodos del main en forma correcta | Invoca a un método del main en forma correcta | No invoca a métodos del main |  |
| 1. Convenciones | Utiliza las convenciones: nombres correctos de clases, atributos, métodos, comentarios e indentación | Utiliza la mayoría de las convenciones | Utiliza algunas de las convenciones | No utiliza convenciones |  |

|  |  |
| --- | --- |
| **Puntajes** | **Nota** |
| Puntaje Total: 27 | 7.0 |
| Puntaje: 25 | 4.0 |

Escala de Notas

La siguiente escala permite asociar el puntaje obtenido con una nota. En caso de puntajes con decimales, se debe aproximar al número entero de la siguiente manera:

\* Decimal igual o mayor a 0.5 se aproxima al entero superior.

\* Decimal menor a 0.5 se aproxima al entero inferior.

