Examen Recuperativo

Programación Orientada a Objetos – 8 de julio 2024

Nombre		**RUT**	
Paralelo	( ) APaolini ( ) MMoraga ( ) ERoss		

Antecedentes generales:

Puntaje total de la prueba Puntos para nota aprobatoria (4.0)	1000 puntos 600 puntos	Puntaje Obtenido	
Duración de la prueba	3 horas	Nota final	
Resultados de Aprendizaje a evaluar	1. Aplicar técnicas de ingeniería de software en la creación de software legible, mantenible y testeable. 2. Aplicar técnicas de programación orientada al objeto en la resolución de problemas. 3. Crear tipos de datos abstractos con bajo acoplamiento entre la implementación y su comportamiento que permitan la resolución problemas. 4. Analizar las relaciones causa efecto de los procesos en estudio. 6. Seleccionar los procesos, técnicas y herramientas adecuados de acuerdo a los requerimientos.		
Fecha de entrega de resultados	11 julio 2024		

Instrucciones:

1. Esta evaluación tiene 4 páginas (incluyendo la portada). Compruebe que dispone de todas las páginas.
2. Lea la prueba completamente **DOS** veces antes de hacer cualquier pregunta
3. Durante la prueba no se puede utilizar: teléfono móvil, apuntes. Está prohibido intentar conectarse a internet de cualquier manera (excepto a Campus Virtual, y solo para subir su solución). Si es sorprendido obtendrá la calificación mínima. Tampoco puede utilizar dispositivos de almacenamiento externos o cualquier otro dispositivo relojes inteligentes, ábacos, etc.
4. Una prueba respondida correctamente en un 60%, de acuerdo con las ponderaciones asignadas, corresponde a una nota 4,0.
5. La prueba es individual, cualquier sospecha de copia será calificada con la nota mínima y el caso será remitido al comité de ética.
6. En su espacio personal no debe haber nada más que hojas de papel en blanco, lápiz, goma.
7. El resto de sus implementos debe guardarlos dentro de su mochila/bolso y ésta debe posicionarse al frente debajo de la pizarra. Si leyó hasta este punto, felicidades, para saber que lo hizo dibuje una esfera al final de esta página.
8. Los estudiantes quienes se les compruebe falta de honestidad académica o cualquier otro acto contrario a las normas de permanencia universitaria o al espíritu universitario, serán sancionados, según sea la gravedad de la falta, con medidas desde la amonestación verbal hasta la suspensión o pérdida de la condición de estudiante, los estudiantes expulsados no podrán volver a ingresar a ninguna carrera, programa o curso de la institución. El estudiante que incurriere en falta de honestidad, durante la realización de un proceso evaluativo, será calificado con la nota mínima 1,0.

Acepto las condiciones firmando: _____

Problema. Gestión de Estacionamiento UCN (1000 puntos)

La Universidad Católica del Norte desea implementar un sistema para gestionar el estacionamiento de su campus Guayacán. Este sistema debe ser capaz de manejar diferentes tipos de vehículos y calcular las tarifas de estacionamiento de acuerdo con dicho tipo de vehículo. Además, el sistema debe leer la información de los vehículos desde archivos TXT.

Hasta ahora existen 3 tipos de Vehículos: **Auto**, **Moto**, **Camión**.

Los diferentes vehículos tienen características en común, pero también cada tipo tiene alguna característica extra:

Tipo	**Característica**
Auto	Cantidad de asientos (cantidad)
Camión	Peso que puede soportar (en kilos, número decimal)
Moto	Cilindrada (en cc, número entero)

Algo muy importante para la UCN es calcular el precio del estacionamiento de cada vehículo. Este está dado por la siguiente fórmula:

Auto	$\frac{27 * cantAsientos * minutos}{5}$
Camión	Si soporta 2 o más toneladas: $\frac{30*minutos*toneladas}{3}$ Si soporta menos de 2 toneladas: $\frac{28*minutos*toneladas}{4}$
Moto	$10 * cilindrada$

Los vehículos están almacenados en un archivo llamado **vehículos.txt**, con el siguiente formato:

```
rut_dueño,nombre_dueño,patente,tipo,modelo,característica_extra
```

`rut_dueño`: El rut de la persona dueña del vehículo
`nombre_dueño`: El nombre de la persona dueña del vehículo
`patente`: La patente del vehículo
`tipo`: El tipo de vehículo: auto, camion, moto
`modelo`: El nombre del modelo del vehículo
`característica_extra`: La característica extra del vehículo, que depende de su tipo

Además, los tiempos que los vehículos estuvieron en los estacionamientos UCN están guardados en un archivo llamado **estacionamiento.txt,** con el siguiente formato:

```
patente,tiempo_estacionado
```

`patente`: La patente del vehículo
`tiempo_estacionado`: La cantidad de minutos que el vehículo permaneció en la universidad, como número entero.

Si una patente aparece más de una vez en el archivo, la cantidad de minutos se debe acumular.

Requerimientos:

Construya una aplicación en Java que lea todos los archivos indicados y que realice las siguientes operaciones:

1. Calcule y escriba por pantalla el total de dinero que se recibe en los siguientes casos:
    a. Los autos que tienen más de 6 asientos
    b. Camiones que soportan menos de 2 toneladas
    c. Total considerando todos los tipos de vehículos
2. Escribir por pantalla la patente que tenga más letras. Considere que pueden existir vehículos de otros países que tienen más letras que las patentes chilenas. Si existe más de una patente que cumpla la condición, debe mostrarlas todas.

3. Listar todos los vehículos y su dueño respectivo. Escriba la patente, el nombre del dueño y los minutos.

4. Construya una versión alternativa del `Sistema`, que siga respetando las operaciones originales. En este caso, el estacionamiento solo contabilizará los montos a los vehículos que tengan a lo menos 10 minutos de tiempo estacionado. En caso contrario, los vehículos igualmente entrarán al estacionamiento, pero se considerará que estuvieron cero minutos estacionados. Con este nuevo sistema, responda las mismas preguntas de los puntos (1), (2) y (3). Construya un `SistemaFactory` para instanciar el tipo correcto de `Sistema`:

```java
/**
 * Retorna un tipo de Sistema de acuerdo al parámetro
 *
 * @param tipo Indica el tipo de Sistema que se quiere instanciar:
 *             0: Sistema "normal"
 *             1: Sistema "alternativo"
 * @return Una instancia de sistema
 */
public static Sistema getSistema(int tipo)
{
    return null;
}
```

Ejemplo de ejecución:

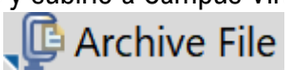
```
NORMAL
-----------------------------
GananciaAutosMasSeisAsientos: 437.4
GananciaCamionesMenosDosToneladas: 32.34924
GananciaTotal: 3469.74924
PatentesMásLargas: XY12345 AUX3993
ListaVehículosConDueños:
AUTO: XY12345 1111 JUAN 9 minutos
CAMIÓN: AUX3993 2222 XIMENA 3 minutos
MOTO: LP12 3333 FRAN 170 minutos

ALTERNATIVA
-----------------------------
GananciaAutosMasSeisAsientos: 0.0
GananciaCamionesMenosDosToneladas: 0.0
GananciaTotal: 3000.0
PatentesMásLargas: XY12345 AUX3993
ListaVehículosConDueños:
AUTO: XY12345 1111 JUAN 0 minutos
CAMIÓN: AUX3993 2222 XIMENA 0 minutos
MOTO: LP12 3333 FRAN 170 minutos
```

Debe entregar:

- Diagrama de clases (20%)
- Código Java (80%)

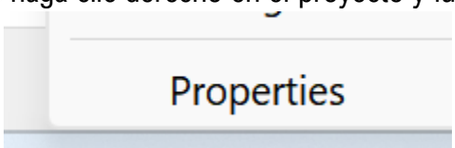
Consideraciones:

- Considere que un vehículo tiene un sólo dueño. Si un dueño ya tiene vehículo, y se lee otro vehículo a su nombre, este vehículo debe omitirse.
- **Debe** usar orientación al objeto. **Debe** usar herencia.
- Debe utilizar el patrón de diseño **Visitor** para responder todas las preguntas del punto 1 y 2.
- Implementar el patrón **Factory** para crear instancias de diferentes tipos de vehículos.
- Otros patrones serán bienvenidos, pero no son obligatorios ni llevan puntaje.
- Use la arquitectura vista en clase: **DEBE** crear un "Sistema" y respetar las reglas de operación de éste.
- Las operaciones del Sistema deben estar documentadas usando javadoc.
- No se deben utilizar ciclos dentro de ciclos. Use funciones para hacerse la vida más fácil.
- Considere que su código debe ser escrito pensando en el futuro. Por ejemplo, debe ser fácil agregar un nuevo tipo de vehículo.
- El código fuente debe exportarlo como .zip y subirlo a Campus Virtual. Use esta opción en el menú exportar:

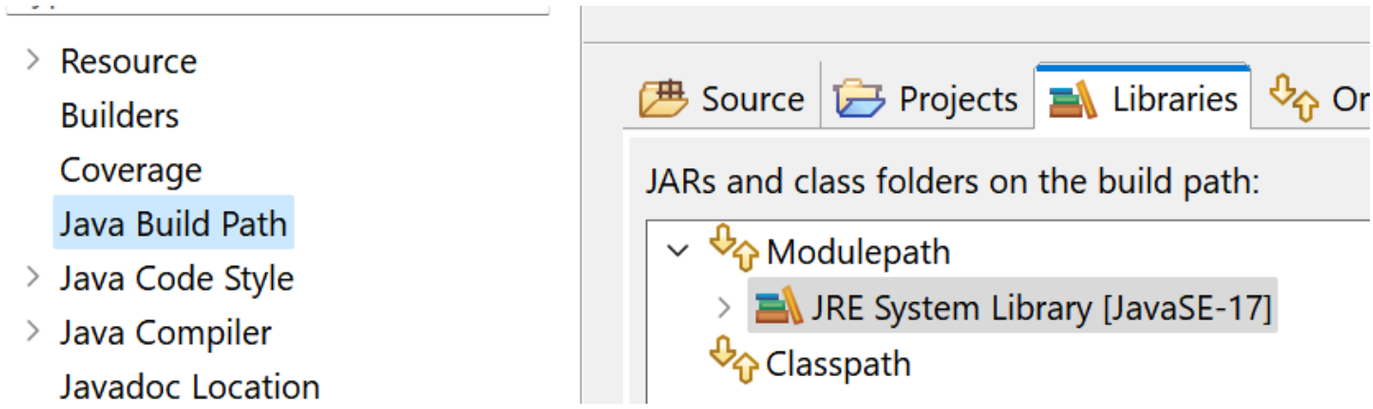


- Archivos .txt de ejemplo se encuentran en CampusVirtual.
- Los diagramas de clases debe escribirlos en papel y entregarlos junto a la prueba.
- Hojas sin nombre no se revisarán.

- Si el IDE no funciona correctamente haga clic derecho en el proyecto y luego en propiedades



En **Java Build Path**, luego **Libraries** y luego doble clic en **JRE System Library.**



Por último, selecciones **Alternate JRE** y aplique los cambios.