PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 2
10ma práctica (tipo b)
(Segundo Semestre 2025)

Indicaciones Generales:
Duración: **110 minutos.**

- No se permite el uso de apuntes de clase, fotocopias ni material impreso.
- Deberá modular correctamente el proyecto en archivos independientes. Las soluciones deberán desarrollarse bajo un estricto diseño descendente. Cada función no debe sobrepasar las 20 líneas de código aproximadamente. Sólo se permitirá una clase por archivo. La clase Principal solo podrá contener el método main y el código contenido en él solo podrá estar conformado por tareas implementadas como métodos públicos de otras clases. En el archivo Principal.java deberá incluir un comentario en el que coloque claramente su nombre y código, de no hacerlo se le descontarán 0.5 puntos en la nota final.
- El código comentado no se calificará. De igual manera no se calificará el código de una función si esta función no es llamada en ninguna parte del proyecto o su llamado está comentado.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no muestren resultados o que estos no sean coherentes en base al 60%.
- Se tomará en cuenta en la calificación el uso de comentarios relevantes.
- Se les recuerda que, de acuerdo con el reglamento disciplinario de nuestra institución, constituye una falta grave copiar del trabajo realizado por otra persona o cometer plagio.
- No se harán excepciones ante cualquier trasgresión de las indicaciones dadas en la prueba.

Puntaje total: 20 puntos

- La unidad de trabajo será t:\ (Si lo coloca en otra unidad, no se calificará su laboratorio y se le asignará como nota cero). En la unidad de trabajo t:\ colocará el(los) proyecto(s) solicitado(s).
- Cree allí una carpeta con el nombre Lab10_2025_1_CO_PA_PN donde: CO indica: Código del alumno, PA indica: Primer Apellido del alumno y PN indica: primer nombre. De no colocar este requerimiento se le descontarán 3 puntos de la nota final.

Cuestionario

- La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en el capítulo 8 del curso: Introducción al lenguaje de programación Java. En este laboratorio se trabajará con clases abstractas y concretas, métodos abstractos, sobre-escritos y concretos, herencia, polimorfismo, encapsulamiento, lectura de archivos e impresión en pantalla.
- Al finalizar la práctica, comprima la carpeta dada en las indicaciones iniciales empleando el programa Zip que viene por defecto en el Windows, no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares.

Para el diseño de su solución, considere que:

- **Se dispondrá de archivos que contienen todos los datos necesarios para la realización del laboratorio.**
- **Se dispondrá de una solución base que contiene la estructura y clases iniciales implementadas de la solución final.**
- **Únicamente puede emplear las clases Scanner y File para leer los datos de los archivos de entrada.**
- **Los arreglos dinámicos serán manejados únicamente con ArrayList.**
- **La impresión de los resultados se realizará únicamente en pantalla.**

Descripción del caso

La autoridad de transporte, como parte de un plan piloto, busca automatizar el control del tránsito vehicular en un tramo de 300 kilómetros de una de las carreteras más importantes del país. **Este control automatizado se enfoca sólo en dos tipos de infracciones de tránsito, la M20 por exceso de velocidad y la G29 por uso indebido de los carriles**. Esta carretera fue seleccionada debido a que, a lo largo de todo el tramo, cuenta con cuatro tipos de carriles claramente diferenciados:

1. Carril de Emergencia
2. Carril Lento
3. Carril Estándar
4. Carril Rápido

El tramo elegido dispone de la siguiente distribución: un (1) carril de emergencia, un (1) carril lento, dos (2) carriles estándar y un (1) carril rápido.
Con el fin de optimizar el flujo vehicular y garantizar un uso más eficiente de la vía, cada carril está destinado a permitir la circulación de vehículos de diferentes categorías y a distintas velocidades, según se detalla en la siguiente tabla:

Tipo de Carril	Velocidad Mínima	Velocidad Máxima	Vehículos Permitidos
Carril de Emergencia	NA	80 km/h	Ambulancia Grúa
Carril Lento	NA	60 km/h	Camión Tractor Tráiler
Carril Estándar	NA	80 km/h hasta el kilómetro 100 del tramo. 120 km/h desde el kilómetro 100 y hasta el kilómetro 200 del tramo 100 km/h desde el kilómetro 200 y hasta el kilómetro 300 del tramo	Motocicleta Coupé SUV Camioneta Sedan Ambulancia
Carril Rápido	80 km/h hasta el kilómetro 100 del tramo. 100 km/h desde el kilómetro 100 y hasta el kilómetro 200 del tramo 90 km/h desde el kilómetro 200 y hasta el kilómetro 300 del tramo	120 km/h hasta el kilómetro 100 del tramo. 160 km/h desde el kilómetro 100 y hasta el kilómetro 200 del tramo 140 km/h desde el kilómetro 200 y hasta el kilómetro 300 del tramo	Coupé SUV Camioneta Sedan

Tabla 1 – Reglas de tránsito por Carril

La autoridad de transporte ha dispuesto la instalación de cámaras a lo largo de los 300 kilómetros del tramo seleccionado. En total, se han colocado 100 cámaras en puntos estratégicos, distribuidas aleatoriamente en distintos carriles. Estas cámaras registrarán el movimiento y la ubicación de los vehículos,

independientemente de si se ha cometido una infracción. Posteriormente, las capturas serán procesadas para identificar y registrar las infracciones correspondientes.

La autoridad de transporte cuenta con un único archivo de texto, cuyos datos están separados por espacios. Este archivo se compone de tres secciones claramente diferenciadas separadas por la palabra clave **'FIN'**, las cuales se describen a continuación:

1. La primera sección que contiene la información de los propietarios de los vehículos, incluyendo el DNI, nombres, apellidos y dirección de cada uno, con el propósito de que cualquier infracción que pueda cometerse sea notificada oportunamente en el domicilio registrado.
2. La segunda sección contiene la información de vehículos, incluyendo datos como placa, marca, modelo, año de fabricación, categoría y DNI del propietario, necesarios para su identificación y para gestionar las notificaciones en caso de infracción.
3. La tercera sección contiene las capturas realizadas por cámaras, incluyendo datos como placa, velocidad, carril, latitud, longitud, región, provincia, kilómetro, fecha, hora y código de la cámara. Estas son necesarias para registrar y ubicar el lugar de la infracción.

El programa deberá ser capaz de:

1. **Cargar y procesar la información:** Leer desde el archivo los datos necesarios sobre los propietarios, vehículos y capturas de las cámaras.
2. **Procesar Capturas y Registrar Infracciones:** En base a la información cargada, procesar las capturas de las cámaras y registrar infracciones de ser el caso.
3. **Imprimir Infracciones:** Imprimir en pantalla las infracciones registradas incluyendo datos de la infracción cometida, datos del propietario, vehículo y la captura de la cámara.

Este sistema permitirá a la Autoridad de Transporte agilizar sus procesos de control vial, optimizando los tiempos de procesamiento, registro y notificación de infracciones de tránsito cometidas, especialmente aquellos cometidas en áreas despobladas. Esto contribuirá a una experiencia más fluida y satisfactoria de conducción.

Anatomía del archivo de datos:

La sección "Propietarios" contiene los siguientes campos: **DNI**, **Nombre**, **Apellidos**, **Dirección**.

DNI	Nombre	Apellidos	Dirección
76543210	Carlos	Pérez_García	Av_Arequipa_1234_Lima
72451236	Lucía	Ramírez_Torres	Jr_Carabaya_456_Cercado_de_Lima
…	…	…	…

La sección "Vehículos" incluye los siguientes campos: placa, marca, modelo, año, categoría (tipo de vehículo), DNI del propietario.

Placa	Marca	Modelo	Año	Categoría	DNI Propietario
RKT-529	Toyota	Corolla	2018	Sedan	76543210
WPM-374	Honda	Civic	2020	Coupe	72451236
…	…	…	…	…	…

La sección de "Capturas" contiene los siguientes campos: placa, velocidad, número del carril (1: Emergencia, 2: Lento, 3: Estándar, 4: Estándar, 5: Rápido), latitud, longitud, región, provincia, km del tramo donde se realizó la captura, fecha y hora de la captura y la cámara que realizó la captura. Esta estructura permite organizar la información de manera clara y detallada para facilitar su uso dentro del sistema.

Placa	Velocidad	N. Carril	Latitud	Longitud	Región	Provincia	KM	Fecha	Hora	Cámara
RKT-529	78.00	3	-12.046422	-77.030000	Lima	Lima	25	29/06/2025	08:15	CAM001
WPM-374	75.00	4	-12.050000	-77.010000	Lima	Lima	30	29/06/2025	14:30	CAM002
…	…	…	…	…	…	…	…	…		

Con esta información se solicita elaborar un proyecto en Netbeans cuya clase Principal estará compuesta por el siguiente código:

```
import java.io.IOException;

public class Principal {
    public static void main(String[] args) throws IOException {
        AutoridadTransporte atu = new AutoridadTransporte("datos.txt");
        atu.cargarDatos();
        atu.procesarCapturas();
        atu.imprimirInfracciones();
    }
}
```

Registro de Infracciones:

Este plan piloto ha sido diseñado únicamente para automatizar la detección de infracciones M20 y G29. Al momento de registrar las infracciones debe tener en cuenta la siguiente tabla.

Código	Monto	Puntos	Descripción
M20	963.00	50	No respetar el límite máximo o minino de velocidad establecido.
G29	428.00	50	Circular en forma desordenada o haciendo maniobras peligrosas

Tabla 2 - Infracciones

El programa sólo registrará dos tipos de infracciones, de acuerdo con las normas de tránsito establecidas en la Tabla 1. La infracción M20 será impuesta a los conductores que excedan los límites de velocidad permitidos, ya sea por superar la velocidad máxima o por circular por debajo de la velocidad mínima establecida. Por otro lado, la infracción G29 se aplicará a aquellos conductores que circulen por carriles no habilitados para su tipo de vehículo o que incumplan las restricciones específicas de circulación. Ambas infracciones se determinarán siguiendo estrictamente las reglas establecidas en la Tabla 1.

Definición de clases:

Nombre	Abstracta	Clase Base	Atributos	Métodos
Registro	Si	N/A	N/A	cargar(archivo: Scanner): void -> **abstracto** imprimir(): void -> **abstracto**
Propietario	No	Registro	dni: int nombre: String apellidos: double dirección: String	cargar(archivo: Scanner): void -> **sobrescribe** imprimir(): void -> **sobrescribe** getters y setters **Opcional:** toString(): String -> **sobrescribe**
Vehiculo	No	Registro	placa: String marca: String modelo: String anhoFab: int categoría: String propietario: Propietario	cargar(archivo: Scanner): void -> **sobrescribe** imprimir(): void -> **sobrescribe** getters y setters **Opcional:** toString(): String -> **sobrescribe**
Carril	Si	N/A	N/A	getTipo(): String -> **abstracto** velocidadMaxima(km: int): double -> **abstracto** velocidadPermitida(km: int, velocidad: double): boolean -> **abstracto** vehiculoPermitido(categoria: string): boolean -> **abstracto**
CarrilEmergencia	No	Carril	N/A	getTipo(): String -> **sobrescribe** velocidadMaxima(km: int): double -> **sobrescribe** velocidadPermitida(km: int, velocidad: double): boolean -> **sobrescribe** vehiculoPermitido(categoria: string): boolean -> **sobrescribe**
CarrilEstandar	No	Carril	N/A	getTipo(): String -> **sobrescribe** velocidadMaxima(km: int): double -> **sobrescribe** velocidadPermitida(km: int, velocidad: double): boolean -> **sobrescribe** vehiculoPermitido(categoria: string): boolean -> **sobrescribe**
CarrilLento	No	Carril	N/A	getTipo(): String -> **sobrescribe** velocidadMaxima(km: int): double -> **sobrescribe** velocidadPermitida(km: int, velocidad: double): boolean -> **sobrescribe** vehiculoPermitido(categoria: string): boolean -> **sobrescribe**
CarrilRapido	No	Carril	N/A	getTipo(): String -> **sobrescribe** velocidadMaxima(km: int): double -> **sobrescribe** velocidadMinima(km: int): double velocidadPermitida(km: int, velocidad: double): boolean -> **sobrescribe** vehiculoPermitido(categoria: string): boolean -> **sobrescribe**
Captura	No	Registro	placa: String velocidad: double carril: Carril latitud: double longitud: double region: String provincia: String km: int fecha: String hora: String codigoCamara: String	cargar(archivo: Scanner): void -> **sobrescribe** imprimir(): void -> **sobrescribe** getters y setters **Opcional:** toString(): String -> **sobrescribe** **Sugerencia:** Se sugiere que el getter getCarril sea sobrecargado y exista una sobrecarga que reciba un parámetro con el nro de carril y que a través de un switch/case retorne la clase derivada apropiada, y otra sobrecarga sin parámetros que retorne el atributo carril. La sobrecarga que recibe el nro de carril puede ser usada para la carga de datos.
RegistroInfraccion	No	N/A	codigoInfraccion: String monto: double puntos: puntos captura: Captura vehiculo: Vehiculo	getters y setters **Opcional:** toString(): String -> **sobrescribe**
AutoridadTransporte	No	N/A	propietarios: List<Propietario> vehiculos: List<Vehiculo> capturas: List<Captura> reglnfracciones: List<RegistroInfraccion> archivo: Scanner	cargarDatos(): void procesarCapturas(): void imprimirInfracciones(): void
Principal	No	N/A	N/A	main(args: String[])

Tabla 3 - Definición de clases

Nota:

Se permite la creación de métodos adicionales cuando se considere necesario, por ejemplo, para lograr una adecuada modularización del código. Cabe precisar que en la Tabla 3 no se han incluido métodos privados ni protegidos; sin embargo, usted puede implementarlos si lo estima conveniente para mejorar la estructura y organización del programa. No obstante, es importante tener en cuenta que este laboratorio ha sido diseñado y probado exclusivamente en base a las definiciones proporcionadas anteriormente. Por ello, se recomienda limitarse estrictamente a dichas definiciones para asegurar la compatibilidad, facilitar la implementación y evitar inconvenientes durante la evaluación.

Parte 1 (5 puntos)

Implementación del método **cargarDatos** de la clase AutoridadTransporte**.**

- **carga de propietarios (1 puntos)**: Lea la sección “Propietarios” del archivo **datos.txt** y cargue los objetos resultantes en la lista de propietarios.
- **carga de vehículos (1 puntos)**: Lea la sección “Vehículos” del archivo **datos.txt** y cargue los objetos resultantes en la lista de vehiculos.
- **carga de capturas (2 puntos)**: Lea la sección “Capturas” del archivo **datos.txt** y cargue los objetos resultantes en la lista de capturas.

** Puede implementar métodos privados en la clase AutoridadTransporte para cada una de las cargas de datos.

Se evaluará el correcto uso del encapsulamiento, la herencia y el polimorfismo.

Parte 2 (10 puntos)

Es importante destacar que esta etapa del laboratorio depende directamente de la correcta implementación de la primera parte. Por lo tanto, antes de continuar, asegúrese de haber completado satisfactoriamente dicha fase, ya que, de lo contrario, el funcionamiento de esta sección no podrá ser validado de manera adecuada.

A continuación, proceda a implementar el método procesarCapturas en la clase AutoridadTransporte:

- Este método será responsable de procesar toda la información contenida en la lista de capturas. Para ello, deberá extraer datos como el kilómetro y el carril en el que se realizó la captura, así como la velocidad a la que circulaba el vehículo.
- Dado que en la captura únicamente se dispone de la placa del vehículo, es necesario implementar un método de búsqueda por placa dentro de la misma clase AutoridadTransporte. Este método permitirá localizar el objeto correspondiente al vehículo involucrado en la captura.
- Al obtener el carril desde la captura, se evaluará rigurosamente el uso correcto del polimorfismo. Es requisito obligatorio que el carril recuperado sea la instancia de la clase derivada adecuada, correspondiente al tipo específico de carril involucrado.
- Una vez obtenido el objeto carril, deberá determinarse si el vehículo está autorizado a circular por dicho carril y si la velocidad registrada en la captura cumple con los límites establecidos para ese carril, considerando el kilómetro de la carretera donde ocurrió la captura.
- En base a esta evaluación, si corresponde, se deberán registrar las infracciones detectadas en la lista regInfracciones.

** Puede implementar un método privado en la clase AutoridadTransporte para procesar capturas individualmente.

Recuerde seguir rigurosamente los principios de programación orientada a objetos, haciendo especial énfasis en la correcta aplicación del polimorfismo y en el diseño limpio y estructurado de los métodos.

Parte 3 (5 puntos)

A continuación, deberá implementar un reporte detallado del registro de infracciones. Para ello, se solicita desarrollar el método imprimirInfracciones dentro de la clase AutoridadTransporte.

Este método tendrá la responsabilidad de recorrer la lista regInfracciones, la cual contiene todas las infracciones de tránsito que han sido detectadas y registradas durante el procesamiento de las capturas.

Durante la iteración, deberá acceder a todos los atributos de cada infracción utilizando de manera adecuada sus métodos *getters*, respetando los principios de encapsulamiento de la programación orientada a objetos.

Con esta información, se generará un reporte estructurado que muestre de forma completa los detalles de cada infracción, sin omitir ningún dato relevante. La información que debe incluir en el reporte comprende:

- Placa del vehículo involucrado.
- Código de la infracción cometida, monto y puntos en la licencia.
- Kilómetro exacto y ubicación geográfica donde ocurrió la infracción.
- Identificación y tipo del carril por el que transitaba el vehículo.
- Fecha, hora y velocidad registrada.
- Información del propietario y el vehículo y cualquier otra información relevante contenida en el objeto RegistroInfraccion.

El formato del reporte debe ser claro, ordenado y legible, de modo que permita identificar fácilmente cada infracción y todos los detalles asociados a la misma. Se recomienda incluir un encabezado general para el reporte y separar las infracciones mediante líneas divisorias u otro formato visual que facilite la lectura.

Asimismo, este método debe limitarse exclusivamente a la generación y visualización del reporte, sin modificar la lista de infracciones ni alterar el estado interno de los objetos involucrados.

Las Figuras 1 y 2 presentan ejemplos referenciales de impresiones de infracciones de tránsito. No se requiere que las impresiones generadas en esta parte coincidan exactamente con el formato mostrado, sin embargo, se espera que la información presentada sea clara, agrupada de forma coherente y concisa, permitiendo identificar fácilmente los detalles de cada infracción.

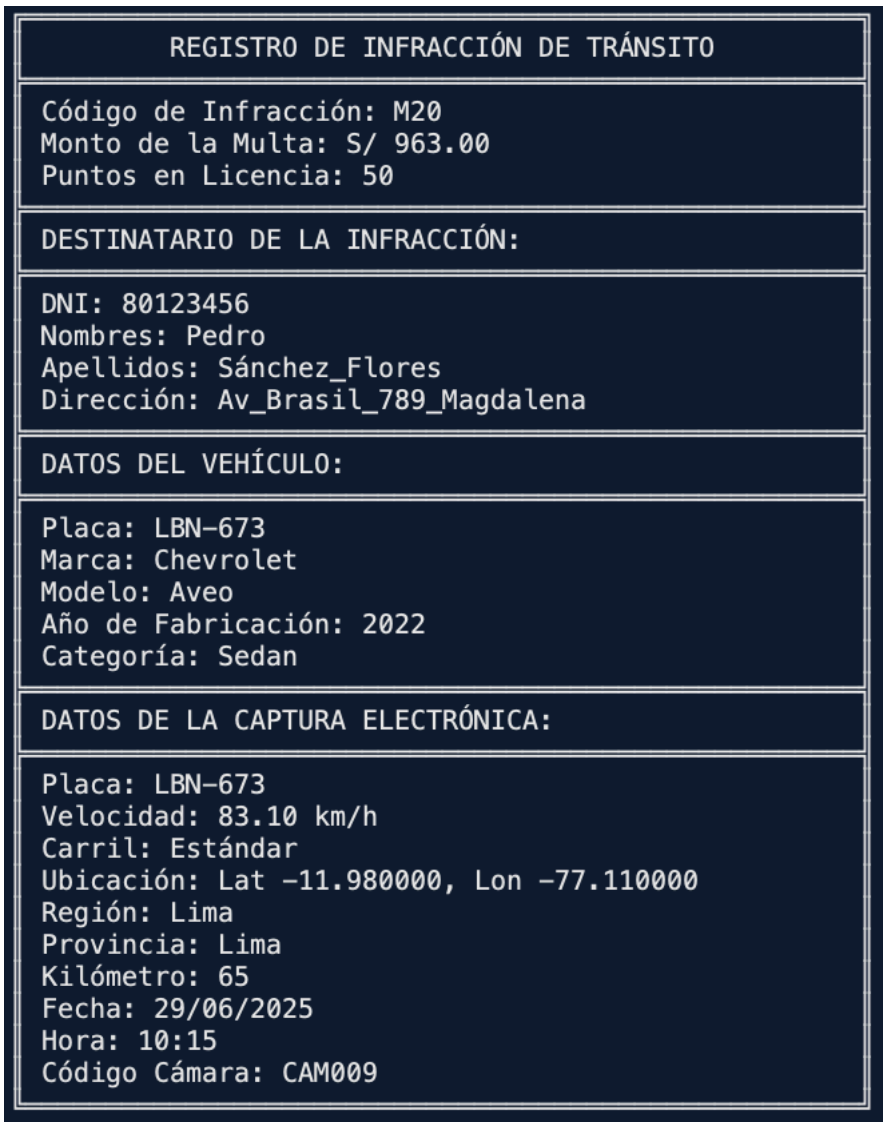

Figura 1 - Impresión de infracción M20

```
        REGISTRO DE INFRACCIÓN DE TRÁNSITO

Código de Infracción: G29
Monto de la Multa: S/ 428.00
Puntos en Licencia: 50

DESTINATARIO DE LA INFRACCIÓN:

DNI: 85678923
Nombres: Carmen
Apellidos: Flores_Sánchez
Dirección: Calle_Grau_321_Arequipa

DATOS DEL VEHÍCULO:

Placa: MKN-940
Marca: Volkswagen
Modelo: Saveiro
Año de Fabricación: 2020
Categoría: Camioneta

DATOS DE LA CAPTURA ELECTRÓNICA:

Placa: MKN-940
Velocidad: 55.00 km/h
Carril: Lento
Ubicación: Lat -11.430000, Lon -77.420000
Región: Lima
Provincia: Huacho
Kilómetro: 135
Fecha: 29/06/2025
Hora: 13:20
Código Cámara: CAM027
```

Figura 2 - Impresión de Infracción G29

Profesores del curso: Miguel Guanira, Andrés Melgar, Rony Cueva, Eric Huiza, Erasmo Gómez

Pando, 4 de julio de 2025