IMPORTANTE:
● Colocar un comentario con su nombre y apellido arriba del método main.
● Añadir comentarios claros que ayuden a comprender su código.
● Cada clase debe estar prolija, ordenada y con la identación correspondiente. **Puede restarse puntaje por desprolijidad.**
● Todo aquello que se encuentre comentado o no sea comprendido, **no se corregirá.**

Narrativa:

La empresa de transporte internacional ***GlobalTrans*** necesita un sistema eficiente y escalable para gestionar su flota de vehículos y el registro de los propietarios de estos. Te han asignado el desarrollo inicial de este sistema. Deberás implementar una serie de funcionalidades clave que luego serán la base para que el equipo continúe añadiendo características. A continuación, se describe lo que necesita la empresa en términos generales; deberás interpretar y deducir la implementación adecuada.

La empresa cuenta con diversos tipos de vehículos, cada uno con características particulares y funciones propias. Los vehículos deben organizarse en una estructura que permita almacenar distintos tipos, como Camión, Furgoneta, Bus y Auto. Cada uno de estos tipos de vehículos comparte ciertas propiedades en común (como patente, kilometraje, y anioFabricacion), pero también tiene características únicas:

- Camión: especifica la capacidadCarga en toneladas, y la tarifa por cada tonelada (este valor será siempre 500)
- Furgoneta: indica el volumenCarga en metros cúbicos, y la tarifa por cada metroCubico (este valor será siempre 300)
- Bus: incluye la capacidadDePasajeros.
- Auto: requiere un registro de Accesorios que deben ser almacenados en un ArrayList<Accesorio>. La clase Accesorio deberá ser creada e incluir, al menos, dos atributos: nombre y precio.

Adicionalmente, la empresa requiere que se incluya un método para calcular el **costo del transporte** de los vehículos de carga (Camión y Furgoneta). Este método debe aceptar como parámetros la **distancia** a recorrer y el **peso** a transportar. Debe verificar si el peso a transportar no excede la capacidad de carga del camión o el volumen de carga de la furgoneta. Si el peso es válido, el método calculará el costo del viaje (distancia * peso * tarifa) y lo retornará; si no, deberá indicar el excedente de forma adecuada.

Por otro lado, el sistema debe incluir una clase de administración llamada Registro, que permita llevar un registro general de vehículos y, de manera separada, un registro de Propietarios. Esta última deberá permitir persistir el nombre de cada propietario, su celular y su DNI(colocar este último como String).

La clase Registro deberá permitir:

1. Agregar vehículos y propietarios, asegurándose de que no haya duplicados. Para esto:
    - No se deben permitir duplicados de vehículos en función de su patente.
    - No se deben permitir duplicados de propietarios en función de su dni.
2. Buscar un vehículo o un propietario específico e indicar si se encuentra dentro de la lista. En ambos casos, debe indicar de forma adecuada si no se encuentra el objeto en cuestión.
3. Agregar un método que busque un elemento por posición. El método deberá recibir la posición por parámetro y retornar el elemento encontrado.
4. Eliminar un vehículo o un propietario específico. En ambos casos, debe indicar de forma adecuada si no se encuentra el objeto que se desea eliminar.
5. Exportar todos los propietarios a un archivo JSON llamado “registroPropietarios.json”, registrando toda la información en un formato adecuado. Pueden realizarlo en el main si lo desean.
6. Traer la información de los propietarios desde su correspondiente archivo JSON. Pueden realizarlo en el main si lo desean.

La empresa requiere que uses **genéricos** en la clase Registro, de modo que sea posible gestionar los registros de Vehículos y Propietarios con independencia. Además, deberás usar **paquetes** para organizar el sistema de clases y capturar posibles **errores específicos** de forma adecuada. Implementa las validaciones necesarias para gestionar situaciones como la duplicación de vehículos y propietarios, o la ausencia de un objeto específico cuando se busca por su patente o dni ya sea para retornar o para eliminar.

Para probar los métodos recuerda “**hardcodear”** información en el main. Es decir, **no** se solicita la clase Menú ni el ingreso de datos por parte del usuario en ningún momento.

¡Exitos!