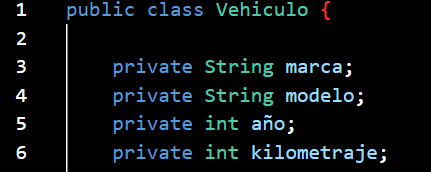
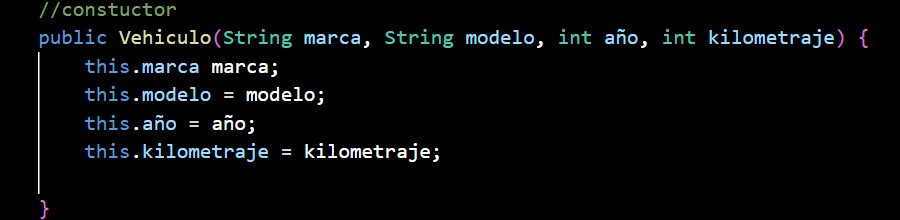
Declaración de atributos y elaboración de constructor



Se define un tipo de objeto que representa un vehículo con ciertas propiedades o atributos. Cada instancia de esta clase (Vehículo) tendrá los siguientes atributos privados: marca (String): Representa la marca del vehículo, modelo (String): Representa el modelo del vehículo, año (int): Representa el año de fabricación del vehículo y kilometraje (int): Que representa la cantidad de kilómetros que el vehículo ha recorrido.

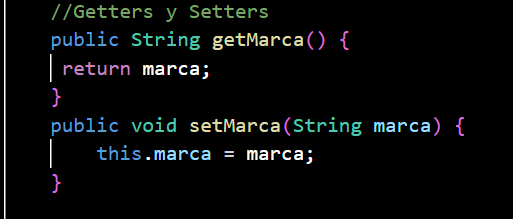
La definición de estos atributos como privados (prívate) significa que no pueden ser accedidos directamente desde fuera de la clase.

El propósito de esta clase es encapsular los datos y proporcionar una estructura para manejar información relacionada con vehículos de manera organizada y segura.



Un constructor es un método especial que se llama automáticamente cuando se crea una nueva instancia de una clase. Su propósito principal es inicializar los nuevos objetos de esa clase.

La palabra clave “this” se usa para referirse al objeto actual de la clase. Se utiliza para diferenciar entre los nombres de los parámetros del constructor y los nombres de los atributos de la clase, ya que ambos tienen los mismos nombres. Sin "this", sería ambiguo saber si se está refiriendo al parámetro o al atributo de la clase.

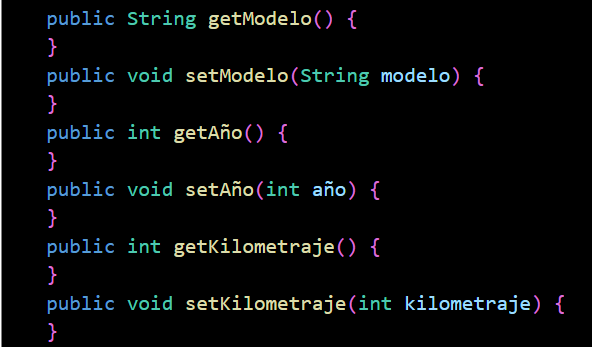


Estos métodos se utilizan para acceder y modificar los atributos privados de una clase, respetando el principio de encapsulación.

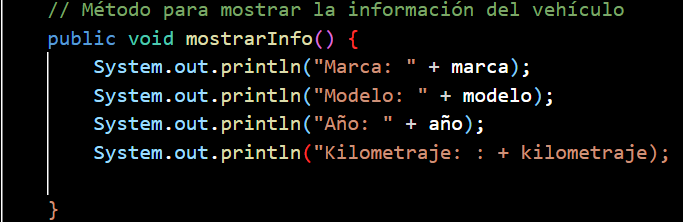
un getter que proporciona una forma controlada de acceder al valor del atributo “marca” desde fuera de la clase.

un setter que proporciona una forma controlada de modificar el valor del atributo “marca” desde fuera de la clase.

Los getters y setters permiten agregar lógica adicional para controlar cómo y cuándo los atributos pueden ser accedidos o modificados.

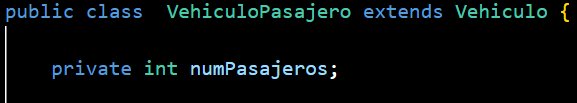


Estos métodos ayudan a mantener la encapsulación, permitiendo un acceso y modificación controlados de los atributos de una clase.



El método “mostrarInfo” está diseñado para imprimir la información del objeto Vehículo en la consola. Este método es útil para visualizar los valores de los atributos del objeto de una manera legible.

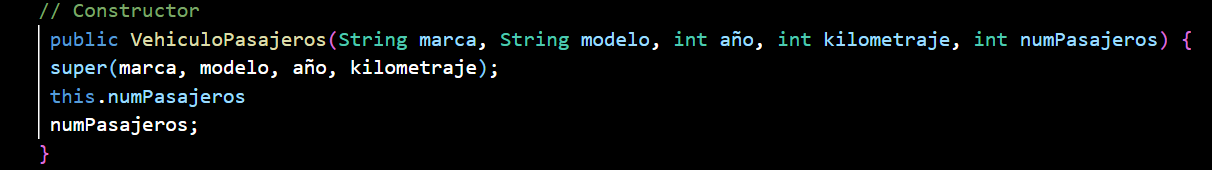
Segunda parte



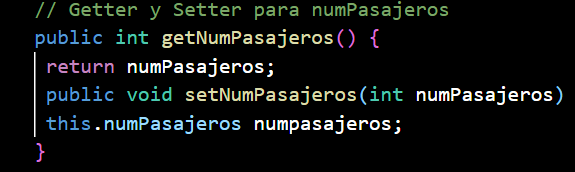
La clase VehiculoPasajeros extiende la clase Vehículo y tiene un atributo privado numPasajeros. Este atributo sirve para almacenar la cantidad de pasajeros que puede llevar el vehículo.

almacena un número entero que representa la cantidad de pasajeros que el vehículo puede transportar.

Al ser privado (prívate), numPasajeros está protegido de modificaciones directas desde fuera de la clase, lo que asegura la integridad de los datos

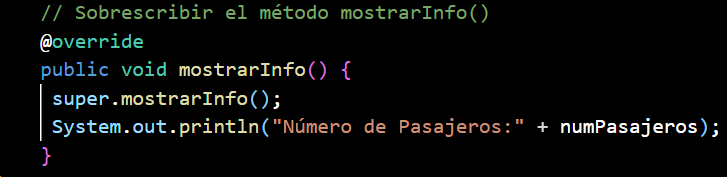


Este constructor permite crear instancias de VehiculoPasajeros con todos los atributos necesarios tanto de la clase base “Vehículo” como de la clase derivada VehiculoPasajeros. Así, cuando creas un objeto de VehiculoPasajeros, puedes proporcionar todos los detalles necesarios sobre el vehículo, incluida la información específica del número de pasajeros.



El método Getter retorna el valor actual del atributo numPasajeros. Permite acceder al número de pasajeros que puede llevar el vehículo desde fuera de la clase.

El método Setter asigna el valor pasado como parámetro al atributo numPasajeros. Permite modificar el número de pasajeros que puede llevar el vehículo desde fuera de la clase.

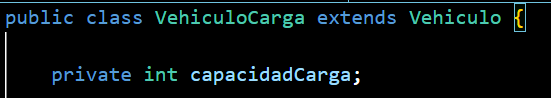


@Override: Indica que el método mostrarInfo está sobrescribiendo un método en la superclase (en este caso, Vehículo). Esta anotación ayuda a evitar errores al asegurarse de que existe un método con la misma firma en la superclase. Proporciona verificación en tiempo de compilación para asegurar que realmente se está sobrescribiendo un método de la clase base.

Llama al método mostrarInfo de la clase base Vehículo. Permite reutilizar la lógica del método mostrarInfo definido en la superclase, asegurándose de que toda la información que muestra el método en la superclase también se muestre en la subclase.

Imprime en la consola la información específica del número de pasajeros del vehículo. Añade información adicional específica de la subclase VehiculoPasajeros que no está presente en la superclase Vehículo.

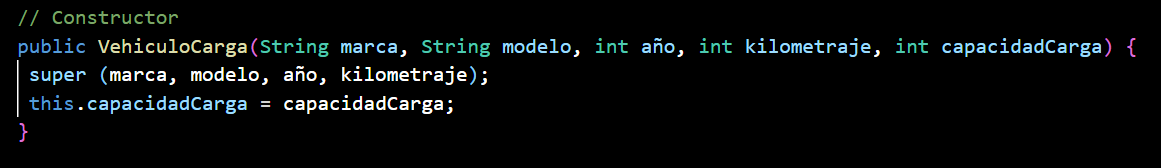
Tercera parte



La clase VehiculoCarga extiende la clase Vehículo y tiene un atributo privado capacidadCarga.

Herencia (extends Vehiculo): La clase VehiculoCarga hereda todos los atributos y métodos de la clase “Vehículo”. Esto significa que VehiculoCarga tiene todas las características de un Vehículo además de las que se le añadan específicamente.

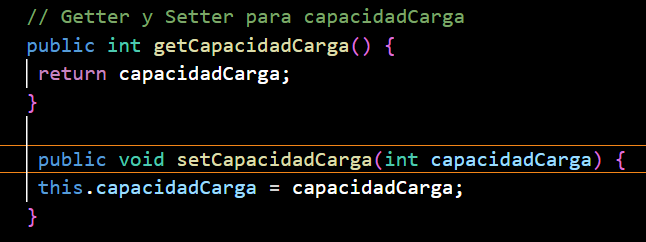
La clase VehiculoCarga tiene un atributo privado llamado capacidadCarga de tipo int. Este atributo representa la capacidad de carga del vehículo en cuestión, probablemente en unidades como kilogramos o toneladas. Al ser prívate, este atributo solo es accesible directamente desde dentro de la clase VehiculoCarga.



super (marca, modelo, año, kilometraje): Esto llama al constructor de la clase base “Vehículo” con los parámetros marca, modelo, año y kilometraje. La palabra clave super se utiliza para llamar al constructor de la clase base. Esto inicializa los atributos heredados de la clase Vehículo.

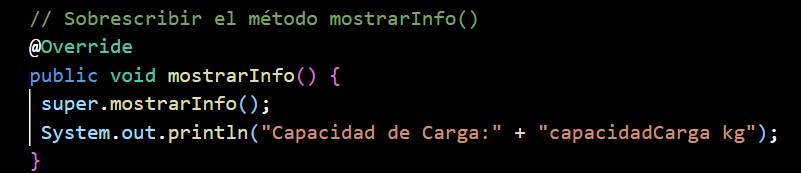
this.capacidadCarga = capacidadCarga;: Esto asigna el valor del parámetro capacidadCarga al atributo capacidadCarga de la clase VehiculoCarga. El this se utiliza para referirse al objeto actual de la clase, y se utiliza para distinguir entre el atributo de la clase y el parámetro del constructor que tienen el mismo nombre.

este constructor permite crear un objeto VehiculoCarga con la marca, modelo, año, kilometraje y capacidad de carga especificados. El constructor de la clase base se utiliza para inicializar los atributos heredados, mientras que el parámetro capacidadCarga se utiliza para inicializar el atributo específico de VehiculoCarga.



son métodos de acceso que te permiten obtener y establecer el valor del atributo capacidadCarga. getCapacidadCarga(): Este método es un getter que devuelve el valor actual de capacidadCarga. Al llamar a este método, obtienes el valor actual de la capacidad de carga del vehículo. setCapacidadCarga(int capacidadCarga): Este método es un setter que te permite establecer el valor de capacidadCarga. Al llamar a este método y pasarle un nuevo valor como parámetro, actualiza el valor de la capacidad de carga del vehículo.

Estos métodos de acceso (get y set) proporcionan un mecanismo seguro y controlado para obtener y modificar el valor del atributo capacidadCarga, permitiendo así el encapsulamiento y la protección de los datos. Esto significa que el acceso directo al atributo capacidadCarga está restringido y solo se puede realizar a través de estos métodos, lo que ayuda a mantener la integridad de los datos y a facilitar la futura modificación del comportamiento del atributo si es necesario.



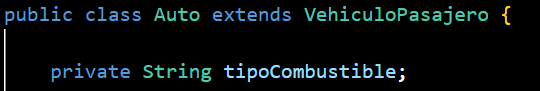
El método mostrarInfo() en la clase VehiculoCarga es una anulación del método mostrarInfo() de la clase base “Vehículo”.

@Override: Esta anotación indica que el método mostrarInfo() está anulando un método de la clase base. Ayuda a garantizar que estés anulando correctamente un método de la clase padre y no creando accidentalmente un nuevo método

super.mostrarInfo(): Esto llama al método mostrarInfo() de la clase base “Vehículo”. Esto permite que la clase VehiculoCarga utilice la implementación existente del método mostrarInfo() de la clase “Vehículo” y luego añada su propia funcionalidad adicional.

Esto imprime la información adicional específica de la clase VehiculoCarga, que en este caso es la capacidad de carga del vehículo en kilogramos. Esta línea se ejecuta después de imprimir la información básica del vehículo utilizando super.mostrarInfo(), lo que permite que la capacidad de carga se muestre junto con la otra información del vehículo.

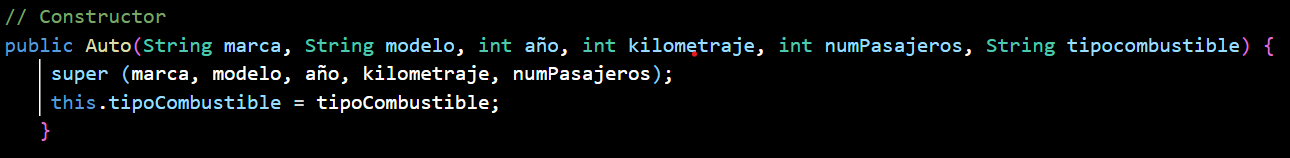
Cuarta parte



La clase Auto extiende la clase VehiculoPasajeros y añade un atributo privado tipoCombustible.

La clase Auto hereda todos los atributos y métodos de la clase VehiculoPasajeros. Esto significa que Auto tiene todas las características de un VehiculoPasajeros, además de las que se le puede añadir específicamente.

La clase Auto tiene un atributo privado llamado tipoCombustible de tipo String. Este atributo representa el tipo de combustible que usa el automóvil (por ejemplo, gasolina, diésel, etc.). Al ser private, este atributo solo es accesible directamente desde dentro de la clase Auto

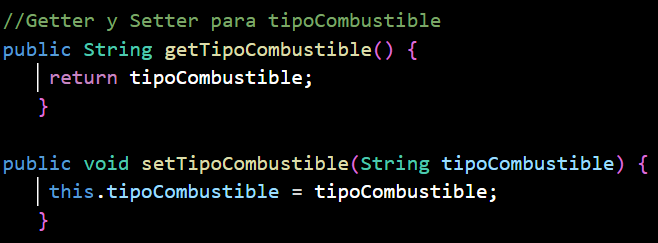


en la clase Auto inicializa un objeto Auto con los parámetros proporcionados.

La palabra clave “super” se utiliza para invocar el constructor de la clase base VehiculoPasajeros, inicializando los atributos heredados de esa clase.

La palabra clave “this” se usa para referirse al atributo de la instancia actual de la clase Auto, diferenciándolo del parámetro del constructor que tiene el mismo nombre.

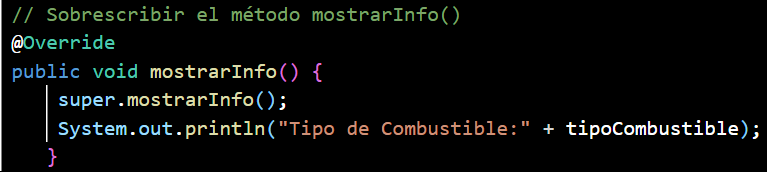
Este constructor permite crear un objeto Auto inicializando tanto los atributos heredados de VehiculoPasajeros como los atributos específicos de Auto.



en la clase Auto esto son métodos de acceso (getter y setter) para el atributo tipoCombustible.

getter que devuelve el valor actual del atributo tipoCombustible. Se utiliza para obtener la información sobre el tipo de combustible del objeto Auto sin permitir la modificación directa del atributo. Permite acceder al valor del atributo tipoCombustible de manera controlada y segura. Mantiene el principio de encapsulamiento al no permitir el acceso directo al atributo privado desde fuera de la clase.

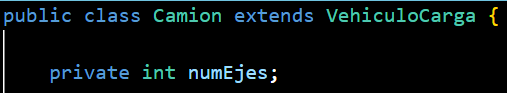
setter que establece el valor del atributo tipoCombustible. Se utiliza para modificar la información sobre el tipo de combustible del objeto Auto. Permite cambiar el valor del atributo tipoCombustible de manera controlada. Permite mantener el atributo tipoCombustible privado, asegurando que cualquier cambio en el valor del atributo pase a través de este método, lo que facilita la validación o el procesamiento adicional si es necesario en el futuro.



El método mostrarInfo() en la clase Auto está diseñado para proporcionar información detallada sobre un objeto Auto, incluyendo tanto la información heredada de la clase VehiculoPasajeros como la información específica del Auto.

@Override: Esta anotación indica que este método está sobrescribiendo un método de la clase base.

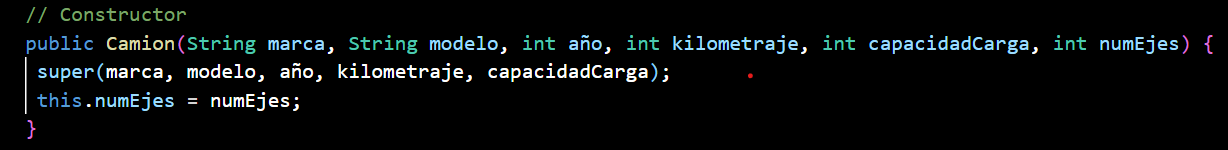
El propósito de sobrescribir el método mostrarInfo() es extender la funcionalidad del método de la clase base para incluir información adicional específica del Auto. Esto es útil ya que proporciona una visión completa del objeto, combinando la información heredada con la nueva información específica de la subclase.



La clase camión extiende la clase VehiculoCarga y añade un atributo privado numEjes.

La clase Camion hereda todos los atributos y métodos de la clase VehiculoCarga. Esto significa que camión tiene todas las características de un VehiculoCarga además de las que se le pueden añadir específicamente.

La clase camión tiene un atributo privado llamado numEjes de tipo int. Este atributo representa el número de ejes del camión.

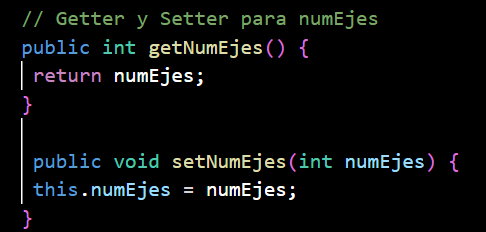


en la clase camión tiene la función de inicializar un objeto camión con los valores

La palabra clave “super” se utiliza para invocar el constructor de la clase base (VehiculoCarga), inicializando los atributos heredados (marca, modelo, año, kilometraje, capacidadCarga).

La palabra clave “this” se usa para referirse al atributo de la instancia actual de la clase camión, diferenciándolo del parámetro del constructor que tiene el mismo nombre.

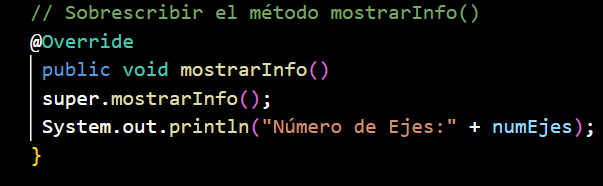
La función del constructor es inicializar un nuevo objeto camión con los valores especificados para cada atributo. Al hacerlo, establece todos los atributos heredados de la clase VehiculoCarga y el nuevo atributo específico numEjes.



Los métodos getNumEjes() y setNumEjes(int numEjes) en la clase Camion son métodos de acceso, conocidos como getter y setter, para el atributo privado numEjes.

El método getter devuelve el valor actual del atributo numEjes. Permite obtener el valor del atributo numEjes de manera segura y controlada. Mantiene el principio de encapsulamiento al no permitir el acceso directo al atributo privado desde fuera de la clase. Esto protege el dato y permite cambiar su representación interna sin afectar al código externo que lo usa.

El método setter establece el valor del atributo numEjes. Permite modificar el valor del atributo numEjes de manera segura. Permite mantener el atributo numEjes privado, asegurando que cualquier cambio en el valor del atributo pase a través de este método. Esto permite incluir validación o procesamiento adicional si es necesario en el futuro.



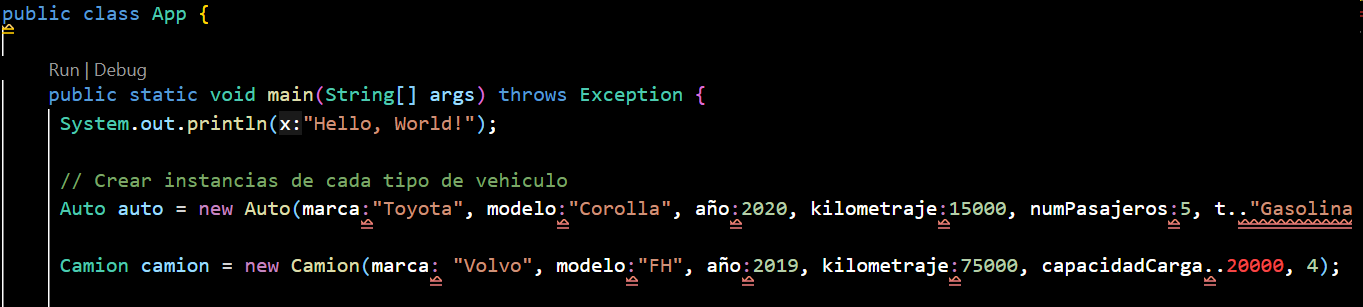
El método mostrarInfo() sobrescrito en la clase camión está diseñado para proporcionar información detallada sobre un objeto camión, incluyendo tanto la información heredada de la clase VehiculoCarga como la información específica del Camión.

@Override: Esta anotación indica que este método está sobrescribiendo un método de la clase base. En este caso, mostrarInfo() está sobrescribiendo un método con el mismo nombre en VehiculoCarga.

super.mostrarInfo();: Esta línea llama al método mostrarInfo() de la clase base (VehiculoCarga). Al hacerlo, se asegura de que la información general del vehículo de carga (como marca, modelo, año, kilometraje y capacidad de carga) sea impresa antes de añadir la información específica del Camión.

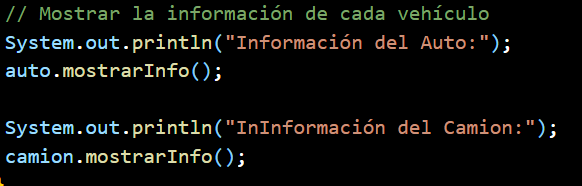
Luego se imprime la información específica del Camión, en este caso, el número de ejes.

Quinta parte



Crea una nueva instancia de la clase Camión y la asigna a la variable camión.

Estos son los argumentos que se pasan al constructor de la clase Camión para inicializar sus atributos. Crea una instancia específica de los objetos Auto y Camión con valores específicos para sus atributos. Esto permite trabajar con objetos concretos que representan vehículos de tipo auto y camión. Estos objetos pueden ser utilizados para realizar operaciones, como mostrar información sobre los vehículos, calcular costos, etc.



Muestra toda la información por consola de los vehículos de la clase auto y la clase camión.