Class: Wheel

Explanation: The Wheel class represents a tire of a vehicle. It contains information about the manufacturer of the tire, the current air pressure, and the maximum permissible air pressure. The class also provides functionalities to pump the tire to its maximum air pressure and add a specified amount of air. In addition, there are methods to get the current and maximum air pressure values.

Class: Engine

Explanation: The Engine class represents an abstract concept of an engine that can be powered by various types of energy This class maintains the current energy level of the engine and its total capacity. There's also functionality to add energy to the engine.

Enum: eFuelType

Explanation: The eFuelType enumeration represents the type of fuel an engine can use. It lists different types of fuel such as Soler, Octan95, Octan96, and Octan98.

Class: FuelEngine

Explanation: The FuelEngine class is derived from the Engine class and represents engines that run on different types of fuel as specified by the eFuelType enumeration. It contains information about the type of fuel used. The class also has functionality to add fuel, ensuring that the fuel type matches the engine's designated type.

Class: ElectricEngine

Explanation: The ElectricEngine class is derived from the Engine class and represents engines that run on electric energy. The energy source is electricity and the unit can be thought of as "hours of charge". The class offers functionality to charge the engine by a specified number of hours.

Class: ValueOutOfRangeException

Explanation: The ValueOutOfRangeException class is derived from the built-in Exception class. It represents a specific type of exception that is thrown when a value is out of a specified range. The class holds information about the acceptable minimum and maximum values, and it provides a custom error message that details the parameter name and its acceptable range.

Class: Vehicle

Explanation: The Vehicle class encapsulates the basic attributes and behaviors of a vehicle. It includes details about the vehicle model, license number, type of engine (whether fuel or electric), a list of wheels, and the current status of the vehicle regarding its repairs or maintenance.

Enum: eCurrentCarStatus

Explanation: The eCurrentCarStatus enumeration contains the possible status of a vehicle. A vehicle can be in one of three states: NotFixedYet, Fixed, or Paid.

Class: Car

Explanation: The Car class is a specialized version of the Vehicle class, specifically for cars. It inherits from the Vehicle class and introduces two additional enumerations: eCarColor (represents car colors) and eNumOfDoors (represents the number of doors a car can have).

Enum: eCarColor

Explanation: Represents the possible colors a car can have.

Enum: eNumOfDoors

Explanation: Represents the number of doors a car can have. The values are also represented numerically to ensure they lie between 2 and 5.

Class: Motorcycle

Explanation: The Motorcycle class is a specialized version of the Vehicle class, specifically for motorcycles. It inherits from the Vehicle class and introduces an additional enumeration: eLicenseType (represents motorcycle license types) and a property for engine capacity.

eLicenseType MotorcycleLicenseType

Explanation: Represents the license type of the motorcycle. There's a check in the setter to ensure the value being set is valid as per the eLicenseType enumeration.

Class: Truck

Explanation: The Truck class inherits from the Vehicle class and introduces properties specific to trucks, such as RefrigeratedTruck and CargoVolume.

Class: CustomerCard

Explanation: The CustomerCard class is designed to hold information about a customer, including the customer's name, phone number, and a list of vehicles they own.

Class: Garage

Explanation: The Garage class is responsible for managing the operations within the garage as well as handling customer cards.

ConsoleUI

Explanation: The user interface for the garage operations. It provides methods to interact with the garage's functions and display data to the user.

|  |
| --- |
| UI |
| +AddVehicleUI() +LicenceNumberOfVehiclesInGarageUI() +ChangeStateOfVehicleUI() +PumpToMaximumAirUI() +AddFuelUI() +chargeElectricalVehicleUI() +VehiclesInformationUI() |

|  |
| --- |
| program |
| Main() |

|  |
| --- |
| **Garage** |
| m\_CustomerCards- |
| Garage() : void+  +AddVehicle() : bool  + +AddCustomer() : void  AddElectricCar() : string+  AddFuleCar() : string+  AddEclectricMotorcycle() : string+  AddFuleMotorcycle() : string+  AddTruck() : string+  + IsCustomerExist() : bool+  LicenceNumberOfVehiclesInGarage() : string+  LicenceNumberOfVehiclesInGarage() : string+  ChangeStateOfVehicle() : void+  PumpToMaximumAir() : void+  AddFuel() : void+  ChargeEngine() : void+  VehiclesInformation() : string+  IsVehicleInGarage() : bool+ |

|  |
| --- |
| CustomerCard |
| - string OwnerName  - string OwnerPhone  - List<Vehicle> VehicleList |
| + CustomerCard() : void  + ToString() : string |

|  |
| --- |
| Vehicle |
| - m\_ModelName: string  - m\_LicenseNumber: string  - m\_Engine: Engine  - m\_WheelsList: List<Wheel>  - m\_CurrentStatus: |
| + Vehicle(string, string, float, float, int): constructor  + Vehicle(string, string, float, float, eFuelType, int): constructor  + ModelName: property get/set  + LicenseNumber: property get/set  + Engine: property get/set  + Wheels: property get/set  + NumOfWheels: property get  + CurrentStatus: property get/set  + ToString(): string override  + Equals(object): bool override  + AreEqual(Vehicle, Vehicle): bool  + GetHashCode(): int override |

|  |
| --- |
| Car |
| - eCarColor CarColor  - eNumOfDoors NumberOfDoors |
| + Car() : void  + Car() : void  + ToString() : string |

|  |
| --- |
| Motorcycle |
| - m\_LicenseType: eLicenseType  - m\_EngineCapacityInCubicCentimeter: int |
| Motorcycle |
| + Motorcycle(string, string, float, float, eLicenseType, int, int): constructor  + Motorcycle(string, string, float, float, eFuelType, eLicenseType, int, int): constructor  + MotorcycleLicenseType: property get/set  + EngineCapacityInCubicCentimeter: property get/set  + ToString(): string override |

|  |
| --- |
| Whell  Wheel |
| - m\_Manufacturer: string  - m\_CurrentAirPressure: float  - m\_MaxAirPressure: float |
| + Wheel()  + CurrentAirPressure: float  + MaxAirPressure: float  + PumpToMax(): void  + AddAirPressure(): void |  + To String(): string |

|  |
| --- |
| ElectricEngine |
| + ElectricEngine() : void  + ChargeEngine() : void  + ToString() : string |

|  |
| --- |
| Engine |
| - float CurrentEnergy  - float EngineCapcity |
| + Engine() : void  + AddEnergy() : void |

|  |
| --- |
| Truck |
| - m\_RefrigeratedTruck: bool  - m\_CargoVolume: float |
| + Truck(): constructor  + RefrigeratedTruck: property get/set  + CargoVolume: property get/set  + ToString(): string override |

|  |
| --- |
| FuelEngine |
| - enum eFuelType  + eFuelType FuelType { get; set; } |
| + FuelEngine() : void  + AddFuel() : void  + ToString() : string |

|  |
| --- |
| ValueOutOfRangeException |
| + ValueOutOfRangeException(): constructor  + MaxValue: property get  + MinValue: property get |