Student Name: Tal Harpaz , ID: 208754622

Student Name: Ariel Ilishayev , ID: 207627985

**Classes and Enums overview:**

**GarageLogic interface:**

Classes:

1. ElectricCar - Represents an electric car. Inherits from the ElectricVehicle class. It has properties for color, number of doors, and a fixed number of wheels (5). The class overrides methods for charging the battery, getting field descriptors (for user input validation), getting vehicle details, and inflating the wheels to their maximum air pressure. It also defines constants for minimum and maximum air pressure in the wheels, as well as minimum and maximum battery time remaining.
2. ElectricMotorcycle - Represents an electric motorcycle. It inherits from the ElectricVehicle class and has properties for license type and engine volume. The class overrides the same functions like ElectricCar, defines the same constants.
3. ElectricVehicle - Is an abstract base class that represents electric vehicle. It defines properties for battery time remaining and maximum battery time. It also includes an abstract method ChargeBattery that must be implemented by derived classes. Additionally, it overrides methods for getting field descriptors and vehicle details.
4. FieldDescriptor - Represents a descriptor for a field or property of a vehicle. It encapsulates the field's name, data type, a validation function to check user input, and an optional action to set the field's value on the vehicle object. This class serves as a metadata container and facilitates dynamic user input handling, validation, and value assignment for vehicle properties.
5. FuelCar - Represents a fuel-powered car with properties like the ElectricCar. It inherits from FuelVehicle and overrides method for refueling, overrides the other functions like all the other vehicles, defines it's constants.
6. FuelMotorcycle - Represents a fuel-powered motorcycle. It inherits from the FuelVehicle class and has properties for license type and engine volume. The class overrides the same functions like FuelCar, defines the same constants.
7. FuelTruck - It inherits from the FuelVehicle class and has properties for transporting hazardous materials and cargo volume. The class overrides the same functions like FuelCar, defines the same constants.
8. FuelVehicle – Very similar to the ElectricVehicle class but an abstract base class that represents electric vehicle, has it's energy related abstract function and fields(fuel).
9. Garage - Manages a collection of vehicles. It has methods to add vehicles, check if the garage is empty or if a specific vehicle is in the garage, get a list of license numbers (optionally filtered by vehicle status), change the status of a vehicle, inflate wheels to maximum pressure, refuel fuel vehicles, charge electric vehicles, and get a specific vehicle by its license number. The class uses a dictionary to store vehicles, with the license number as the key and the vehicle object as the value.
10. InputValidator - A static class with various methods for validating user input in different formats.
11. ValueOutOfRangeException - A custom exception class that is thrown when a value is outside of a specified range.
12. Vehicle - An abstract class that serves as the base for representing vehicles in the garage. It contains properties for the model name, license number, wheels, and owner details. It also has an abstract method InflateWheelsToMax which is implemented by derived classes to inflate the tires of the vehicle.
13. VehicleBuilder - A static class that provides functionality for creating different types of vehicle objects in the. It uses a dictionary VehicleCreators that maps each vehicle type (eVehicleType) to a function that creates an instance of the corresponding vehicle class.
14. Wheel - Represents a vehicle wheel in the. It has properties for the wheel's manufacturer name, current air pressure, and maximum air pressure. The class provides a method Inflate to increase the wheel's air pressure by a specified amount, ensuring that the new pressure does not exceed the maximum allowed value. It also overrides the ToString method.

Enums:

1. eCarColor – A Enum which represents the options of colors for a car vehicle(Fuel and Electric). The options are: Yellow, White, Red, Black
2. eDoorsNumber – A Enum which represents the options of number of doors for a car vehicle(Fuel and Electric). The options are: Two, Three, Four, Five.
3. eFuelType – A Enum which represents the options of a fuel type – Relevant only for the fuel vehicles(Car, motorcycle and truck). The options are: Soler, Ocatn95, Octan96, Octan98. A fuel car has Octan95, a fuel motorcycle has Octan98 and a truck has Soler.
4. eLicenseType – A Enum which represents the options of a license type - Relevant only for the motorcycles(fuel and electric). The options are: A, A1, AA, B1.
5. eVehicleStatus – A Enum which represents the options of a vehicle status. The options are: UnderRepair, Repaired, Paid. Every new vehicle that is added to the garage is at status UnderRepair.
6. eVehicleType – A Enum which represents the options of the vehicles in the garage. The options are: FuelMotorcycle, ElectricMotorcycle, FuelCar, ElectricCar, FuelTruck.

**ConsoleUI interface:**

Classes:

1. UI - Provides a console-based user interface for managing vehicles in a garage. It allows the user to add a new vehicle, list vehicles based on their status, change the status of a vehicle, inflate tires to the recommended pressure, fuel vehicles, charge electric vehicles, and display vehicle details. The class utilizes the VehicleBuilder component to instantiate new vehicles based on their type. The class interacts with the Garage class.
2. Program - Serves as the main entry point for executing the console-based garage management application.

**Inheritance diagram:**

