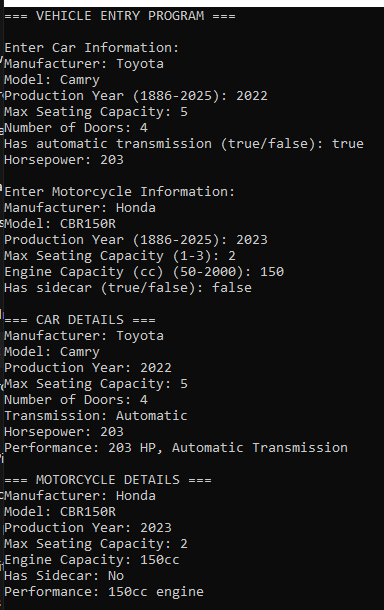
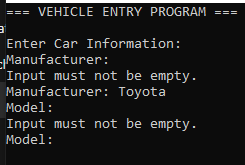
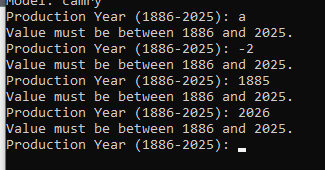


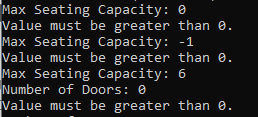
**OUTPUTS**

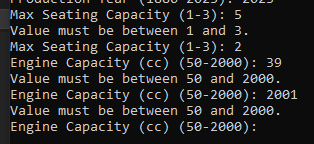
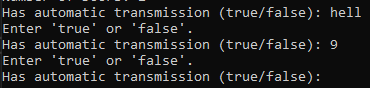
****

**ERROR HANDLINGS**

****

****

****

****

**EXPLANATION**

In my program, I used inheritance by creating a base class called Vehicle, which holds general information like manufacturer, model, production year, and seating capacity. Then I created two child classes, Car and Motorcycle, that inherited from Vehicle and added their own specific features such as HorsePower or EngineCapacity. I also applied abstraction by making Vehicle an abstract class with the abstract method DescribePerformance(), which both subclasses had to define in their own way. This made it easier to describe performance differently for a car and a motorcycle.

Method overriding is shown through the displayInfo() method. The base class has the general version, while each subclass overrides it to include specific details and also calls the DescribePerformance() method. For validation, I used control flow with if statements and do-while loops to make sure input values were valid before creating any objects. I also made helper functions like ReadNonEmpty(), ReadIntInRange(), and CheckBool() to make input checking easier and avoid repeating code.

Constructors were used in each class to assign values to the properties when creating an object. I used : base(...) to send the general vehicle info to the parent constructor, which helped keep things clean. Each vehicle’s info is displayed correctly because the displayInfo() method shows both the inherited and unique properties. I struggled a bit at first with using base and overriding methods, but this activity helped me understand how it all connects. I also enjoyed creating reusable functions to handle user input and validation.