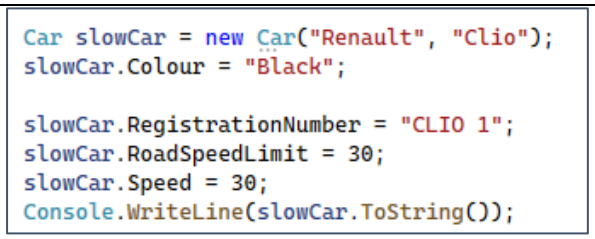
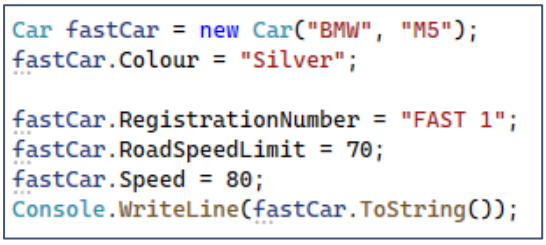
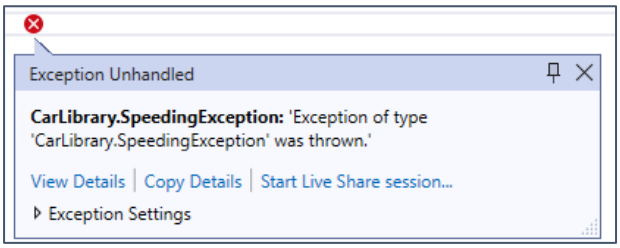
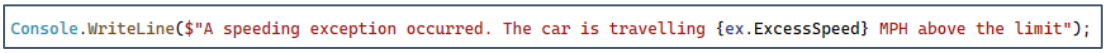
Exception Handling

The objective of this exercise is to consolidate your understanding of exception handling and creating and throwing custom exceptions.

1	Open the **CarLibrary** solution in: **C:\Courseware\QACS\Labs\13_Exception_Handling\Begin**
2	Comment out the existing code in **Program.cs**
3	Open **Car.cs** and override the **ToString** method to display detailed car information: `return $"Car Make is {Make}, Model is {Model}, Colour is {Colour}, Speed is {Speed} MPH";`
4	Add a new auto-implemented property called **RoadSpeedLimit**.
5	You will change the logic of the Speed setter to account for the road's speed limit and whether or not the value that you are setting is a legal driving speed for the current road. • If it is, set the value • If it is not, you will raise a custom exception
6	In a separate file in the class library project, create an exception class called **SpeedingException**. Don't forget to use inheritance.
7	Within the new custom exception class, create an auto-implemented property called **ExcessSpeed** and set this value within the constructor.
8	In **Car.cs**, if the car is not travelling at a legal speed, throw a new instance of **SpeedingException,** ensuring you pass in the excess speed value.
9	In **Program.cs**, create two new car instances: **slowCar** and **fastCar**
10	Set the following values for **slowCar**:

	
11	Set the following values for **fastCar**: 
12	Compile and run your application. You should see an unhandled exception: 
13	Uncomment the existing code in **Program.cs**. Set the **RoadSpeedLimit** to **50** for Car **c2.** Wrap the code in this file with **try…catch…finally** blocks to handle the exceptions that are thrown. Add a catch block for **Exception** as well as for **SpeedingException**. Utilise the properties of the exception class to display useful messages to the console: 
14	Compile and run your application. Observe the exceptions that are thrown and caught.

15	Add a property to store the *Car instance* within the **SpeedingException** and use this to access information that can be output to the console to help identify the Car that is speeding:
	
	``
	catch (SpeedingException ex)
	{
	    Console.WriteLine($"A speeding exception occurred. The car is travelling {ex.ExcessSpeed} MPH above the limit");
	    Console.WriteLine($"A speeding exception occurred. Car {ex.Car.RegistrationNumber} is travelling {ex.ExcessSpeed} MPH above the limit");
	}
	``

If you have time

16	Create a list of valid colours within Car.cs and create a custom InvalidColourException that is thrown if the colour is not in the list.
17	Observe how this exception is caught by the generic Exception event handler.
18	Add a custom catch block to handle this specific type of exception.
19	A suggested solution is provided in the End folder for your reference.