**PE11 – Defining Classes**

**Due Saturday 19-Sept-2020 by 11:59pm**

* Define the following terms.

inheritance: When a child class keeps the same methods and variables of its parent class. The child class can modify some of the same methods it shares and it can also add new data. For example, a parent class is Animal and has a method called makeNoise(). A sub class can be Cow and its makeNoise() method can be modified to return the string "moo".

parent/base class: This is the main class that sub classes will be derived from. It creates a basic structure that the sub classes will follow and be modified to their specific needs. Many sub classes can be derived from the parent class. To create

child/derived class: A child/ derived class is a class that inherits properties from a parent class. It will share the same methods and variables that are declared by the parent class. It can also hold its own methods/functions and variables.

* What is wrong with the following code?

public sealed class MyClass

{

// class members

}

public class myDerivedClass : MyClass

{

// class members

}

- **Public sealed classes prohibit inheritance so myDerivedClass cannot inherit MyClass's properties.**

* How would you define a non-creatable class?

**- A class where an object cannot be created using the New keyword. The class has to be referenced in other objects.**

* Why are non-creatable classes still useful? How do we make use of their capabilities? Give an example of a non-creatable class that we have used.

**- Non-creatable classes are still useful because they give us the ability to make a framework from which child classes can work from. Basically like a shapeless cookie cutter that cookie cutters made from it fill in their own shape like a circle or triangle.**

* Write code in a class library project (.NET .DLL) called Vehicles that implements the Vehicle family of objects as outlined below (\_424DoubleBogey as the class identifier). There are nine objects and two interfaces (IPassengerCarrier and IHeavyLoadCarrier) that require implementation. The virtual method should be an empty function:

public virtual void LoadPassenger() { }



GitHub URL:

* Create a console application project, Traffic, which references Vehicles.dll (created in Q5 above). Include a function called AddPassenger() that accepts any object with the IPassengerCarrier interface. Within the AddPassenger() function, call the LoadPassenger() method using a reference to the interface. Also add a line to call the ToString() method inherited from System.Object (ie. if vehicleObject is passed to the function, call Console.WriteLine(vehicleObject.ToString()). Also try passing an object that did not inherit the IPassengerCarrier interface and see what happens.

GitHub URL: <https://github.com/jag8932/myIGME-201/tree/master/PE11_Goodwillie>

***Submission***

Upload this completed document and GutHub URL's for #5 and #6 to the corresponding MyCourses dropbox.