**Name: Patrick O’Brien**

For each of the below questions, write a short sentence or two to express (in your own words) your answer. Keep the answers short, but use complete, correct, English sentences.

If it helps to clarify the questions, feel free to mentally prefix all the questions with the phrase "According to the video…"

1. After you’ve watched all the videos, please answer this question:  
   Of all the videos that you watched, if you could pick one video to be re-recorded by the instructor outside of class which would you choose? Why?  
   (Keep in mind the recording outside of class will omit any pauses from the instructor answering student questions, have less hemming and hawing, etc, and generally be more concise)

|  |
| --- |
| **VIDEO: Polymorphism: An Array Of Objects With An Overridden Method because the mic rubs all over and sounds horrible.** |

**VIDEO: Inheritance: Conceptual overview**

1. What is the purpose behind inheritance? (Give a very brief, intuitive explanation)

|  |
| --- |
| Inheritance is the ability to create a class that inherits attributes and behaviors from an existing class (class SpyCar inheriting attributes from class Car). The new SpyCar class inherited the attributes of the Car class, and then added additional attributes to it to make it a SpyCar and not just an ordinary car. |

1. It is bad to copy-and-paste the Car code and then tweak it in order to create a more specialized subclass/subcategory (such as, for example, a SpyCar). Briefly, intuitively explain one reason why this is bad / will introduce problems later on.

|  |
| --- |
| You will have to remember several months later why SpyCar is important, how it differs from a normal car. When using Car as the base and inheriting its attributes, it would be easy to read the code and understand what the differences are. |

1. In order to tell C# that a given class extends another, existing class, what do add to the class declaration?

|  |
| --- |
| Class SpyCar : Car (c#) or class SpyCar extends Car (java) |

**VIDEO: Inheritance: vocabulary**

1. A **SpyCar** has a copy of what?   
   (This will be covered in more detail in the next video)

|  |
| --- |
| SpyCar is a copy of the superclass/base class Car. |

1. What are the two pairs of words we use to talk about the Car & SpyCar classes? Make sure that you’re clear which word describes Car and which word describes SpyCar, and which two words typically go together in which pair:

|  |
| --- |
| A SpyCar is a copy of all the instance variables that are declared in its base class (Car).  Car would be considered the super class, and SpyCar would be considered the subclass. **Also base class 🡪 derived class (more specialized one).** |

1. If you have a class, then create a subclass of it, and you then create a further subclass of the first subclass?   
   What advantage(s) would this give you?

|  |
| --- |
| FlyingSpyCar and BoatSpyCar, it would give the advantage of having all the changes run down from the classes above, and it has all the subclasses use that code. So if you had a bug in your Car class, and you fixed it, you would be fixing the same bug in all the subclasses simultaneously. |

**VIDEO: Inheritance: Inheriting data**

1. What data fields does an instance of the SpyCar class have? For each field, explain where it came from / why the SpyCar has it.

|  |
| --- |
| TankSize 🡪 only accessible form within the Car class. SpyCar will have this method, but will not have access to change it without the getTankSize getter method. Change the size of the fuel tank in gallons  numSmokeScreen 🡪 how many charges for smoke screen are required. |

1. Can a derived class directly access a data field that was declared to be private in a base class?  
   If not, then how does the derived class make use of it (explain this briefly and intuitively)?

|  |
| --- |
| No, it must use a getter method declared in the base class, otherwise the base class has to be changed from private to protected. This allows only derived classes to access it directly.  The derived class has a copy of the method but cannot change it. |

1. What does the C# keyword **protected** do? How is this different from **private**?

|  |
| --- |
| This allows only derived classes to access it directly. Private means that only the base class can access the method, nothing else. |

1. Can an instance of the **Car** class ever access the fields of the derived **SpyCar** class? Explain why this does (or does not) make sense, briefly and intuitively.

|  |
| --- |
| It cannot, there’s no guarantee that a car is a SpyCar. Can’t call any class the subclasses define. Because Car is itself defining SpyCar, and any other subclasses. it would break things. |

**VIDEO: Inheritance: Constructors**

1. In C#, how do you call a base-class constructor? If there are multiple (possibly overloaded) constructors in the base class, how do you indicate which one you want to use?

|  |
| --- |
| Public SpyCar (double ts, int sm) : base( ts)  Says that SpyCar constructor has two parameters, base class needs ts (tank size) + everything that the new more specialized class needs sm (smokescreen charges), and then pass everything up to the base class constructor base(ts).  You indicate which one you want to use by changing the parameters to match the constructor in base class that you want to use. |

1. How can you set things up so that all instances of the **SpyCar** class always has, say, zero backseats.

|  |
| --- |
| Public SpyCar (double ts, int sm) : base(ts, 0); 🡪 0 sets the backseats to 0 in each instance. |

1. If you don’t specify a base-class constructor (to use in a derived class’s constructor), which one will C# use?

|  |
| --- |
| It will use the default constructor. You have to use some constructor in the Car class to make sure all the components of a car are built up before everything in SpyCar is built. Basic car has to be set up before the more specialized derived classes are. |

1. How can you provide a default constructor in the base class without duplicating code amongst the constructors?

|  |
| --- |
| Public Car() : this(10.0, 2); constructor inside the Car() class. References the other constructor. Then all you need to do is change the derived classes to work with the new format. |

1. Building off your answer for the previous question, what is a very typical pattern when a class provides multiple constructors?

|  |
| --- |
| It is a very common pattern to have one real constructor and have everything else jump into the other constructor. Everything else is provided for convenience, and then dumped into the main constructor that does the work as much as possible. |

1. Why do you have to copy-and-paste the parameters for the derived class’s constructor(s)?

|  |
| --- |
| Because the derived class is a completely new constructor method. All the variables must be redeclared. Copy the ones from the normal car, but not necessarily requiring them. C# is leaving open the option to modify these values. |

**VIDEO: Polymorphism: Overriding Methods**

1. When you declare a method to be **virtual**, what does that mean?

|  |
| --- |
| The virtual keyword is used to modify a method, property, indexer, or event declaration and allow for it to be overridden in a derived class. |

1. In order to override a method in a subclass, what do you need to copy from the superclass?

|  |
| --- |
| You need to copy the signature and the return value, the public, the void, and the name exactly, any parameters.  You intend to override the basic implementation of the base class with a more specialized version. |

1. In addition to what was listed in the prior question, what other keyword do you need to add to the subclass’s method?

|  |
| --- |
| Override  Ex: override public void Print() |

1. When **playerCar** is set to refer to the **c1** object, and you call **Print**, what happens? Why?

|  |
| --- |
| It prints the basic print method 🡪  Car tank size:10  It refers to a Car or a subclass of car. If assigned to c1, it tells c1 to print itself, checks to see if there is a better version of print (overridden), if not, use already defined and print self. |

1. When **playerCar** is set to refer to the **sp** object, and you call **Print**, what happens? Why?

|  |
| --- |
| If referring to SpyCar  Checks to see if better version of print defined in SpyCar.  Uses the new version of Print() in SpyCar to print  This is a SPYCAR, the tank size is 40. |

1. What is the name for this technique?

|  |
| --- |
| This is called polymorphism.  The playerCar variable can refer to multiple forms of the object. PlayerCar, ClownCar, etc. |

1. In C#, if you wanted the **SpyCar** to invoke (to call) the base-class version of **Print** (i.e., **Car.Print**), what snippet of C# code would you use?

|  |
| --- |
| Base.Print(); in C# in java 🡪 super.Print(); |

1. What is the name of the technology that allows (parts of) your C# program to run in parallel?

|  |
| --- |
| Threads 🡪 concurrency (how to get multiple things running at the same time) |

**VIDEO: Polymorphism: An Array Of Objects With An Overridden Method**

1. What output will be produced by the loop that iterates through the **garage** array and calls the Print method on all the non-null slots? Please summarize briefly in English what will be printed, and then make sure to explain WHY this output is produced.

|  |
| --- |
| At slot 0, there is no object  At slot 1, there is no object  At slot 2, there is no object  Car tank size:200  At slot 4, there is no object  Car tank size:30  This is a SPYCAR, the tank size is 400  Car tank size: 400  At slot 7, there is no object  At slot 8, there is no object  At slot 9, there is no object  It creates a new array called garage with 10 slots. Then slot 3,5, and 6 and filled with objects [3]=Car(200,6) [5] = c2 [6] = SpyCar(400, 4)  Then it checks the array and prints the contents if they are not null, and if they are, it prints that they are null at that slot number in the array.  It’s calling the classes and creating new objects using the base class and the derived classes. |

1. When adding objects to an array, are you required to create a new object for the array? If not, how would you add, say, the **c2** car object into the array

|  |
| --- |
| You are not required to create a new object for the array.  You could just use garage[5] = c2; |

**VIDEO: Named Constants**

1. What is a “magic number”?

|  |
| --- |
| A number selected in the video, in this case 3, that is printed in a string. |

1. In terms of printing out a magic number, what is a better alternative to simply printing the number directly?

|  |
| --- |
| A “magic” number is a (“number {0}”, numTries). What we’ve been doing with changing values in text. |

1. What modifier can be used to mark an integer (or double, or short, or char, etc) as a named constant?

|  |
| --- |
| Const |

1. What is the “readonly” modifier used for?

|  |
| --- |
| Readonly is a constant modifier that applies to reference variables. |

1. What happens if I try to change the value of a named constant (after it’s been created)?

|  |
| --- |
| It won’t let you, left hand side of assignment must be a variable, property, or indexer. Something that can change its value. A constant is not a variable, as it cannot change its value. |

1. How do people (traditionally) capitalize named constants?

|  |
| --- |
| Named constants are traditional written in all caps. All caps is intended to be a const and should never be changed. |

1. Where do you normally put named constant declarations?

|  |
| --- |
| You normally attach them to the entire class (declared outside a method). They are constants used throughout the entire program. |

1. Copy over the C# source code that declares a new class named ComputerScreen, including it’s two named constants:

|  |
| --- |
| Class ComputerScreen  {  Public const double WIDTH = 10.5;  Public const double HEIGHT = 14.0;  } |

1. Copy over a snippet of C# source code that will make use of the two constants in a class that’s separate from the ComputerScreen class?

|  |
| --- |
| Static void Main(string[] args)  {  Console.WriteLine(“The width of the screen is {0}”, ComputerScreen.WIDTH);  Console.WriteLine(“The width of the screen is {0}”, ComputerScreen.HEIGHT);  } |