**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)

|  |
| --- |
| I still believe it would be highly beneficial to move all the videos to a universal platform such as youtube. Otherwise it’s annoyingly tedious to get it to work correctly on a different device such as a phone. |

**VIDEO: Passing an object to a method (objects as parameters)**

(It may be best to watch this video in the ‘fullscreen’ mode)

The video starts with an overview of the program that we’ll be using in this video; it’s great review up until about the 2:30 mark.

1. What is the overall goal of passing an object to a method?

|  |
| --- |
| Being able to manipulate the data passed into the method by the object. It’s in this way that methods can be called to transform or manipulate data to perform calculations in an efficient manner for the programmer. |

1. Within this particular program, what is meant by the “leftmost” point?

|  |
| --- |
| On a cartesian plane, to the left of the x axis. So whichever point has the smallest value of x. |

1. What pattern do you follow to pass a pair of points into a method (such as PrintLeftMost)?

|  |
| --- |
| Type and then name, a name that may or may not have any relation to any other variables.  Public void PrintLeftMost(Point p1, Point p2, int x, int y) |

1. On the call**ing** side, how do you pass (say) the **origin** and **near** points into the PrintLeftMost method?

|  |
| --- |
| You create a new object of the class to call the method.  PointComparer comparer = new PointComparer();  Comparer.PrintLeftMost(origin, near); |

1. Why do we call PrintLeftMost(near, far) then immediately call PrintLeftMost(far, near)

|  |
| --- |
| Comparer.PrintLeftMost(near, far)  Comparer.PrintLeftMost(far, near) |

**VIDEO: Objects are passed (as a parameter) by reference**

1. In the example project the program first prints out the starting value of the **near** point, then calls the MovePointLeftwards command, then prints out the ending value of the **near** point.  
   What is this code trying to test / what question will this program help us answer?

|  |
| --- |
| The code is moving the point to the left on the x axis. By calling comparer.MovePointLeftward(near) it moves the point to the left by 2, then prints the new position. |

1. What does “pass by reference” mean?

|  |
| --- |
| “Passing by reference” means that you pass the variable itself into the function, not just the value.  Comparer.MovePointLeftward(near) |

1. What does “pass by value” mean?

|  |
| --- |
| “Passing by value” means that you pass the actual value of the variable into the function. |

1. What are some of the advantages of passing something (an object, an array) by reference?  
   What are some of the disadvantages/dangers of passing something (an object, an array) by reference?

|  |
| --- |
| According to: <https://stackoverflow.com/questions/373419/whats-the-difference-between-passing-by-reference-vs-passing-by-value>  “**the "pass by value vs. pass by reference" distinction as defined in the CS theory is now obsolete** because **the technique originally defined as "pass by reference" has since fallen out of favor** and is seldom used now.1” “  “When a parameter is **passed by reference**, the caller and the callee **use the same variable** for the parameter. If the callee modifies the parameter variable, the effect is visible to the caller's variable.”  “When a parameter is **passed by value**, the caller and callee have **two independent variables** with the same value. If the callee modifies the parameter variable, the effect is not visible to the caller.”  Passing an object by reference will change the original value of the variable. While it could be easier to work with, and more memory efficient, the original value could then not be called again for reference since it has been reassigned to the new value. |

**VIDEO: Return an object from a method (objects as return values)**

(Note that the new material starts around the 2:00 mark – you can skip ahead to that point if you want)

1. Why is it bad to have your constructor ask the user for input?

|  |
| --- |
| Because the constructor will print something to the console every single time it creates a new object. Therefore it will ask for user input many more times than is necessary and will lead to general confusion. |

1. How do you declare (in C#) that your method will be producing/returning a Point object?

|  |
| --- |
| Public Point CreatePoint()  {  Point localPt = new Point(x, y)  Return localPt;  } |

1. How do you tell C# exactly which Point object you wish to return when the method is executing?

|  |
| --- |
| You could an object to hold the return value of CreatePoint()  Pt1 = ptMaker.CreatPoint();  Pt1.Print(); |

1. Why will you get an error when attempting to call pt2.Print() prior to the pt2 = ptMaker.CreatePoint(); line?

|  |
| --- |
| Compile time error, because pt1 is the line that creates the new point. Because the point has not yet been created, you cannot print it. |

1. What is one very common error that you absolutely do not want to do?  
   Also make sure to explain briefly and intuitively why this is bad.

|  |
| --- |
| You do not want to do Point pt2 new Point(0,0)  Pt1= null  Pt2 🡪 creates a new point object (0,0) 🡪 allocate the new object 🡪 assigns it to pt2 at address 3000🡪 allocates new point then with Point localPt = new Point (x,y) 🡪 then localPt = 4000 🡪 returns 4000 memory address in pt2 🡪 then overwritten with 4000 🡪 3000 address is now unused garbage memory. |

1. When the CreatePoint method returns **localPoint**, what is it actually returning?

|  |
| --- |
| It is returning the new memory address at point 4000. |

1. What is the term for a block of memory that has no references pointing to it?

|  |
| --- |
| Garbage memory. |

**VIDEO: Class with an array of Double values inside it (GradeTracker)**

1. What is the major difference between the array that we previously looked at, and what we’re going to look at today?

|  |
| --- |
| An array that referred to objects. Each spot in the array referred to a separate object.  In the case of the array that we are looking at today, we are going to have an object (GradeTracker). That refers to an array, so it’s the other way around. |

1. When creating a new GradeTracker object, what will it do (array-wise)?

|  |
| --- |
| Create GradeTRacker object – creates an array that is 10 elements wide – call printgrades method on it and will print everything that is in the array.  When you call addgrade the first time, it will add the contents to the first space in the array. When you call it a second time, it will add elements to the second space in the array.  Every other space that is not called by addgrade method will be initialized to 0. |

1. What will the PrintGrades method do?

|  |
| --- |
| The PrintGrades method will print all the contents of the array to the console. |

1. How do you declare an instance variable that is an array (of, say, doubles)? Where do you put that declaration?

|  |
| --- |
| If you want to declare an instance variable with the array double, you would declare it inside a class but outside any method.  You would declare it with: double [] grades; |

1. Why do you NOT need to do the numGrades = 0; line within the constructor?

|  |
| --- |
| It’s technically not needed because numGrades is always given the default value for its type which is already 0 (so it is already initialized).  However it makes it clear to the programmer that filling the array with 0 is intentional. It is also good practice for languages that do not automatically declare variables to a default type, such as c++. |

1. Does the declaration (from the previous question) actually create the array of **double**s?   
   If not, what line of code actually creates the array (since line numbers aren’t being displayed in the video you can just copy the line of code into this quiz):

|  |
| --- |
| Assuming you are referring to question 20, which is not the previous question.  No it does not create the array itself, just the space in memory for the array. The actual array length definition is defined in the GradeTracker() constructor.  Grades = new double[10]; |

1. Briefly, intuitively summarize how the AddGrade method will work.

|  |
| --- |
| numGrades starts at 0, each time the AddGrade method is called, it will check to see if the value of numGrades is less than the total length of the grades array.  If it is less than the total length of the grades array, it will assign the value of g to the slot in the array that is the current value of numGrades.  Then the value of numGrades will increase by one. In this way each time the method is called, the next position in the array will be filled by the value of G. |

1. In the PrintGrades method, why does the for loop iterate until i reaches numGrades instead of until I reaches grades.Length? What will this print out to the screen?

|  |
| --- |
| It will print out every defined value in the array. It uses numGrades instead of grades.Length because it does not print out the spaces that have not been filled by numGrades. |

1. In the example video, can you use the syntax **gt[0]** to access the grade in slot 0 within the GradeTracker object referred to by the **gt** variable?  
   (Why would it be bad to allow this?)

|  |
| --- |
| You could do that, but it would be extremely bad to do.  It would no longer be private. Anyone could change any element of your array at any time, or allocate another array and assign their own data entirely to the new array. |

(Again, make sure that you watch the remainder of the video so that you can develop an intuitive understanding of how the computer executes this code)

|  |
| --- |
| **Class with an array of Double values inside it (GradeTracker): Additional Demo Video**  There are no viewing quiz questions for this video.  This video explains the Histogram class which should save you time when working on the exercise that uses this class. |

**VIDEO: Class with an array of Object References inside it (PointTracker)**

1. In this video we’re going to create an array of references to objects that is located where?

|  |
| --- |
| An array with references to objects, and have that array be itself be inside another object. |

1. What does the PointTracker.Print method do? How many points will it print out the very first time that it’s called?

|  |
| --- |
| The PointTracker.Print method performs the same function as in the previous question. It will print every value in the array. |

1. Why is it very convenient to use the following line of code? How might it be superior to creating a local variable to hand to the Add method?  
   pt.Add( new Point( 10, 10 ) );

|  |
| --- |
| Create a new point object, pass in origin as the p object in the Add method. Finds empty space in array and assigns the value of p to the empty space in the array.  It is superior to creating a local variable because it does not require the local variable origin that will never be used again. |

1. In the following line of code, where should you NOT put a semi-colon?  
   (There are a lot of places where you shouldn’t put the semi-colon – make sure you list the one explained in the video because it’s a very common mistake that people make)  
   pt.Add( new Point( 10, 10 ) );

|  |
| --- |
| pt.Add( new Point( 10, 10 ); ); do not put semicolon here, only at the end of the statement. |

(Again, make sure that you watch the remainder of the video (after the 8 minute mark) so that you can develop an intuitive understanding of how the computer executes this code)

**VIDEO: Class with an array of Object References inside it WITH NULL VALUES (PointTracker)**

1. What will this video deal with?

|  |
| --- |
| That is located within an object of itself.  An object 🡪 that contains a reference 🡪 to an array 🡪 of references to more objects |

1. What is different about this video?

|  |
| --- |
| How to deal with null values that are within the array. |

1. How does the Print method differ from the one shown in the prior video?   
   What effect will this have on the running program?

|  |
| --- |
| 1. It will check every slot in the array. However it specifies that if the value is not equal to null, then it will print that value. 2. It will print out nothing the very first time it is called because every value in the array is equal to null. |

1. Briefly, intuitively summarize (in your own words) how the new AddGrade method will work when you add the object referred to by the **origin** variable here:

|  |
| --- |
| Origin is created, and a new object is created.  Object is located at mem address 350, so origin has an address at 350.  Then it calls Add(Point p, int where)  P is assigned the origin address of 350.  Where is given the default value of 0.  AddGrade will check to see if where is larger than 0, and then checks to see if it is less than the total length of the array. If it is, then it will add the value of p to that current empty space in the array. |