**Name: Jason Truong**

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…"

* 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)

|  |
| --- |
| < Write your answer here > |

**VIDEO: Basic Parameters**

* For an instance method, what do I need to have (in my program) before I have can have my program call the instance method?

|  |
| --- |
| You need have an actual object to call the method from. |

* Copy the example from the video (up to the 1:30 mark) of defining a class named Demo with a method named Triple, along with an example of calling that method from within **main**. (At this point the Triple method doesn’t actually do anything)

|  |
| --- |
| class Demo  {  public void Triple()  {  }  }  class Program  {  static void Main(String[] args)  {  Demo d = new Demo(); |

* How do you add a parameter to the Triple method’s definition?  
  How do you pass the value 7 into the method call for Triple in **main**?

|  |
| --- |
| You, on an object of your choice, on a call of a functionality with 7 passed into the function. It should look like in our case.  d.Triple(7);  With the 'Triple' definition too  public void Triple(int x) |

* Does the name of the parameter (in the method’s definition) have any connection to any variables declare in the **main** function?

|  |
| --- |
| The parameter definition does have a connection , with its only present is the values passed through to call the method. |

* Briefly, intuitively describe what the “Call stack” is, and what “stack frames” are.

|  |
| --- |
| It sets an origin point for the code when it jumps from methods to methods. Everytime its finished with the origin, it remembers the order it was in for the 'stack' and returns to its position. |

* Will changing the value of **x** in the **Triple** method ever affect the value of **num** in the **main** method?

|  |
| --- |
| Not at all. |

* If you want to list out multiple parameters, how do you separate them? How would you specify that you want an integer parameter named **x**, and a double named **y**.

|  |
| --- |
| Commas.  Like this: public void AAAAAAAAAAAAAAAAAAAAAAA(int x, double y){  } |

**VIDEO: Basic Return Values**

* Briefly, intuitively describe what “Default parameters” are used to accomplish.

|  |
| --- |
| It allows a dedicated local variable for the set methods so you aren't forced to call the method with a variable of your own and instead just have it called empty. |

* Copy down the example from the video that has the Triple method taking it’s parameter, tripling it, and then sending that new value back:

|  |
| --- |
| public int Triple(int x) {  int result = x\*3;  return result;  } |

* When Visual Studio tells you that your method has the “Not all code paths return a value” error, what problem does your code have and how do you fix it?  
  (Explain this clearly **in your own words**)

|  |
| --- |
| Two things.  Through all your ifs, whiles, in-loops, and elses, if the chance of returning a dedicated variable doesn't exist; potentially leaving the return value to unspecified or null, it will give you this error in hopes that you close up all notches with a proper return.  Second thing would just be the dumb way the code thinks and how it just wants you to add a return statement outside all of the code and logic gates just so it can feel real sure. |

* If a method produces a return value are you required to make use of that return value in, say, **main**?

|  |
| --- |
| if it isn't needed. |

**VIDEO: 02\_String\_Compare.mp4: Q + A: String Comparison with ==, String.Compare**

* During the execution of the following code snippet, how many string objects will be created? Once the snippet has finished, what object will **input** refer to? What object will **input2** refer to?  
  string input= “ok”;  
  string input2= “ok”;

|  |
| --- |
| Only one "ok" will be created |

* During the execution of the following code snippet, how many string objects will be created? Once the snippet has finished, what object will **input** refer to? What object will **input2** refer to?  
  string input= (string)“ok”.Clone();  
  string input2= (string)“ok”.Clone();

|  |
| --- |
| Two separate strings are made. both input and input2 will refer to two different ok string variables |

* In C#,what does the double-equals ( == ) operator do? Continuing the code snippet from the prior question, will the following evaluate to true or false?  
  if( input == input2 )

|  |
| --- |
| When using the '==' logic gate, it asks if the two objects compared are the. For strings input and input2, they're true because the Strings contained are the same down to the case sensitivity. |

* What are the three possible results that the String.Compare( string1, string2, true ) method might produce? What does each value mean?

|  |
| --- |
| The three results are -1, 0, and 1.0. If the strings are the same, it will return a value of 0. If the first string is larger than the second value, it reutrns a -1. If the second value is larger than the first value, it returns a 1. What determines things being larger or smaller is if they're first in a "dictionary".  No intuitive way to explain this for me, sorry I have to cough this up in verbatim. |

* What does the ASCII standard define? How does Unicode differ from ASCII?

|  |
| --- |
| ASCII standard is what defines letters or 'chars' in strings to an equivalent numeric value equivalent in the ASCII standard. Unicode is inclusionary for languages outside the English standard; like the Japanese lexicon and their egregiously large three alphabet system. |

* How can you control whether String.Compare is comparing the two strings in a case-sensitive manner or a case IN-sensitive manner?

|  |
| --- |
| Changing the (x, x, true) to (x, x, false) when in the String.Compare. |

**VIDEO: 04\_Basic\_Array\_Stuff.mp4: Basic Array of Primitive Types**

* Give some examples (2-3) of C#’s “primitive data types” or “simple data types”.  
  What do all primitive/simple types have in common?

|  |
| --- |
| primitive/simple data types are reserved into the space of MAIN. |

* Is an array a simple type? How do we know?

|  |
| --- |
| No, it has to request separate storage space for the array itself |

* Briefly, intuitively, what does **the heap** contain?

|  |
| --- |
| A more complex version of Stack that focuses on contigency throughout the code - often called from the keyword 'new' |

* In terms of the length of time that variables and objects exist, how are the stack and the heap different?

|  |
| --- |
| Stacks are for more often a static variable that acts its durations only as the function may call for. A heap will store a variable that will continue to exist outside of its 'active' duration. |

* When C# creates an array of integers, what value will C# fill the array in with? What value is used to fill in an array of doubles?

|  |
| --- |
| The array will be filled with 0s for integers, and 0.0 for doubles if the values aren't specified to be Null. |

* What is a good way to think about your computer’s memory (about your computer’s RAM)?

|  |
| --- |
| Interpreting your memory as an array of bytes. |

* When the instructor talks about something being located at memory address 70,000 where is that thing located in RAM (explain this using the intuitive model you described in the previous question)?

|  |
| --- |
| Building from the array of bytes, the memory adress would be the equivalent o the 70000th byte in the byte-array of your memory |

* What is the value of nums at minute 15? What does this value represent / what does it mean?

|  |
| --- |
| The nums variable is at a value of 70,000. |

* Give a short C# snippet to demonstrate how you would put the value 20 into slot 0 in the array referred to by the **nums** variable.  
  Also give a short C# snippet to demonstrate how you would print out the value stored in slot 0 in the array referred to by the **nums** variable.

|  |
| --- |
| You would inject the integer in position 0 in the integer with the nums array, nums[0] = 20; To print it you write from the 0 position, Console.WriteLine(nums[0]); |

* In C#, what is the index/address of the first element of all arrays?

|  |
| --- |
| 0 is the first element in the index/address. |

* If there are five elements in the array, what is the highest valid index?  
  If there are **N** elements in the array, what is the highest valid index?

|  |
| --- |
| There are five indexes starting at 0, so the highest valid is 4.  The index for 'N' elements would have the highest index as N-1. |

* In C#, if you create an array with 5 slots can you assign a value to slot 10?

|  |
| --- |
| You can't. |

* In C#, can you change the size of an array? If not, then what can you do that’s equivalent to changing the size of the array?

|  |
| --- |
| No, the array size is set upon its initialization. You can take all the indexes of the first array and duplicate it into a second array. |

* Typically, when you resize an array, is it a good idea to increase the size of the array by only 1 new element? Why or why not?

|  |
| --- |
| If you like creating a recursive set of arrays that are only one sized bigger, be my guest. Its a difficult and meticulous waste of space. |

* Memory that you cannot use because you have no variables that refer to it (directly or indirectly) is called what?

|  |
| --- |
| Memory that cannot be used or accessed is called "garbage memory" |

* If C# (or Java) runs out of memory, what does it do? What is this process called?

|  |
| --- |
| The process is called 'garbage collection', and it is where it takes all the unaccessable 'garbage memory' in a program and allocate it into an array for more space. |

* Why is it safe for the computer to jump directly to element 10 of an array (rather than searching through memory, slot by slot, for element 10)?

|  |
| --- |
| Because all the indexes elements are in-sequenced with no breaks. |

* If you wanted to print how many elements are in an array, how could you do that? (Give a brief snippet of C# code)

|  |
| --- |
| Console.WriteLine("Array nums is {0} elements long", nums.Length); |

* Why is it useful to choose the size of an array (when we first create/allocate the array) using a variable?

|  |
| --- |
| When the array is created from the initialization from established input, it can't be shifted. Using a variable from the start can have a less static and more flexible array size we can use. |

* Copy the example C# code that can print out all the elements of an array into the space below

|  |
| --- |
| for (int index = 0; index < nums.Length; index++) {  Console.WriteLine("Slot {0} contains the value: {1}", index, nums[index]);  } |

* Briefly explain how your code in your answer to the previous question works.

|  |
| --- |
| The index integer represents the 0 index for the array, then it goes through the entire array until it reaches the cap, "N-1". |

**VIDEO: 05\_How\_To\_Adjust\_Console\_Buffer.mp4: How To Adjust The Console's Buffer Size**

* How can you adjust the console’s buffer size in Microsoft Windows? Assuming that the console window is open, list each of the steps that you need to go through (first click here, then click this, etc, etc)

|  |
| --- |
| Clicking on the icon at the top left, a scroll tab appears, go down to edit. A new window appears, and adjus the width and height of the buffer size specifically. |

**VIDEO: 06\_Array\_Params.mp4: Arrays as Parameters (syntax)**

* When calling a method that accepts an array as a parameter, how do you print the name of the array?  
  (I.e., in main, how do you write out the name of the array)

|  |
| --- |
| Alice a = new Alice();  hypothetically have a method named Alice.  You have an array named "variance".  a.PrintArray(variance); will print the array with the name, assuming the Alice method is built. |

* In the example, how was the array parameter declared for the PrintArray method?  
  Do you need to include square brackets in the method’s definition?

|  |
| --- |
| Yes, you have to declare the method 's definition with brackets to specify its looking for an array parameter. |

* Copy over the entire definition for the Demo class and the PrintArray method (use the version that has only the array parameter)

|  |
| --- |
| public void PrintArray(double [] nums){  } |

**VIDEO: 07\_Pass\_By\_Reference\_Value.mp4: Arrays as Parameters (pass by reference vs. pass by value)**

* In the example in the video at the 2:00 minute mark, what value do both **vals** and **nums** have? What does this mean in terms of which actual block of memory each one is referring to?

|  |
| --- |
| The vals and nums hold the same value in the memory address/array, they have a mirrored byte set. |

* Because of your answer to the prior question, executing the C# code for nums[1] = 10; in the PrintArray method has what side effect?

|  |
| --- |
| The vals array being substantiated in main will print nums[1]'s slot identitically. So nums[1] will be 10, and be printed as such. |

* What is the name for passing a variable this way?

|  |
| --- |
| Passing by reference |

* Fill in the blank: “When passing a simple type, you actually get a \_\_\_\_\_\_\_\_\_”

|  |
| --- |
| copy |

* At the 5:00 minute mark in the video, if you change the value of y what effect does that have on the variable x?

|  |
| --- |
| Because y is using a copy of variable x thats passed through the method, rather than x itself, x does not change. |

* What is the name for passing a variable this way?

|  |
| --- |
| pass by value |

**VIDEO: 08\_Return\_Values.mp4: Arrays as return types**

* Copy down the example that demonstrates how to return an array from a new MakeArray method from the video (at the 1:30 mark)

|  |
| --- |
| public int[] MakeArray(int size) {  int[] newNums = new int[size];  return newNums;  } |

* When returning an array, do you use square brackets?   
  Give a short C# snippet to demonstrate how one might return an array named newNums:

|  |
| --- |
| No. newNums is already established as an array, treat it like a single table thats boxed.  Just use, return newNums; |

* Describe (briefly and intuitively) the two different types of multidimensional arrays that are available in C#. What is the name of the second type?

|  |
| --- |
| The int[,] is a giant matrix, it uses a 'coordinate system' to find the index.  The second array is acalled a 'jagged array', it allows a dynamic size setting for each row in the array. |

* Copy down the example that demonstrates how to receive an array from the MakeArray method and assign it to a variable named differentName (around the 6:00 mark)

|  |
| --- |
| int[] differentName;  differentName = d.MakeArray(10) |

* Given the following code snippet, how many arrays are allocated? At the end of the snippet, which array does differentName refer to?  
  int[] differentName;  
  differentName = d.MakeArray(10);  
  differentName = d.MakeArray(10);

|  |
| --- |
| It creates a whole new array the second time. So assume the first array is at memory adress 80,000; the 'new' differentName array is at address 90,000. |