**Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

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)

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

**VIDEO: Debugging, Part 1**

1. Other than staring at the source code a lot, what other way can people debug their code (without using the debugger feature built into something like Visual Studio)?

|  |
| --- |
|  |

1. What is good about the approach you listed in the prior question? What is bad about it?

|  |
| --- |
|  |

1. What does a “breakpoint” do?

|  |
| --- |
|  |

1. Describe (at least) 2 different ways that you can toggle a breakpoint for a given line:

|  |
| --- |
|  |

1. In Visual Studio, how do you start running the program so that the debugger can be used to debug your program?

|  |
| --- |
|  |

1. When the running program hits a breakpoint, does the debugger stop before or after the execution of the line that the breakpoint is on?

|  |
| --- |
|  |

1. How can you examine the value of a variable using your mouse?

|  |
| --- |
|  |

1. What are the two windows that you can use to examine variables? What does each one do (and how are they different)?

|  |
| --- |
|  |

1. What information does the “Call Stack” window display? How is this useful?

|  |
| --- |
|  |

1. What can you do with the “Immediate Window”?

|  |
| --- |
|  |

1. What does the “Step Into” menu option do? What does the “Step Over” menu option do? What does the “Step Out” menu option do?

|  |
| --- |
|  |

1. If the program is currently executing a method (like getX() ) and you want to examine the state of the **main** function, how can you use the Call Stack to do that?

|  |
| --- |
|  |

(The video ends abruptly because the instructor pressed the F10 key, which tells Visual Studio to Step Over and also (accidentally) tells the recording software to stop recording – D’oh!!! ☺ )

**VIDEO: Debugging, Part 2**

1. What is another name for the printing-out-information approach?

|  |
| --- |
|  |

1. What is ‘logging’, and why is it useful? Make sure to give a specific type of programming that commonly uses logging.

|  |
| --- |
|  |

**VIDEO: Enums**

1. What are enumerations generally useful for? What will we typically be using them for in this class?

|  |
| --- |
|  |

1. Copy the ErrorCode example from the video here, and make sure that you’re clear on the syntax (how you need to type this stuff). Make sure to fix any Word auto-corrections that happen.

|  |
| --- |
|  |

1. What is the goal of the ErrorCode enum, specifically using the example of the SmartArray.Add() method?

|  |
| --- |
|  |

1. Given a variable named **ec** (of type ErrorCode), how could you check if it has the OK value? Give a short snippet of C# code in order to illustrate how to do this:

|  |
| --- |
|  |

1. Given a variable named ec (of type ErrorCode), how could you check if it does **NOT** have the OK value? Give a short snippet of C# code in order to illustrate how to do this:

|  |
| --- |
|  |

1. How will we be using ErrorCode enums in this class?

|  |
| --- |
|  |

1. When you declare something to be an enum, what data type does the compiler translate that into?

|  |
| --- |
|  |

1. How can you specify the value you want an element of your enumeration to have (for example, how could you specify that OK should have the value 0)?

|  |
| --- |
|  |

1. For all the elements of the enum that you do NOT explicitly assign a value to, how does C# figure out what value to use?

|  |
| --- |
|  |

**VIDEO: Exception Handling: Overview**

1. What is major objective of exception handling?

|  |
| --- |
|  |

1. What is the major downside of using the older model of error handling?

|  |
| --- |
|  |

1. What is the major advantage of exception handling (error-handling-wise)?

|  |
| --- |
|  |

1. Should you use Exception Handling to handle normal, expected situations in your code?   
   (Give one example of a normal situation that’s not an exceptional situation, and therefore one that you should NOT use exception handling for)

|  |
| --- |
|  |

1. Write out a quick example of C# source code that demonstrates the try, catch, and finally blocks.  
   (You can leave empty the stuff between the { and } – I want to mostly make sure that you saw the red text that highlights the syntax of exception handling)

|  |
| --- |
|  |

1. What happens if an exception happens inside a try block?

|  |
| --- |
|  |

1. What is the finally block useful for?

|  |
| --- |
|  |

**VIDEO: Exception Handling: File I/O Overview**

1. Are you expected to be able to write File I/O code based on the material that’s being covered in this video?  
   (Note that you may cover File I/O elsewhere in this course, in which case you would then be expected to know this material well enough to write code using it)

|  |
| --- |
|  |

1. What two ‘flavors’ do files come in? Briefly describe each one, and mention why we’re using a text file for this program.

|  |
| --- |
|  |

1. Where will the file appear when the WriteFile method creates a new file?

|  |
| --- |
|  |

1. Give a quick overview of what the ReadFile method does, focusing mostly on what the method does to read the information out of the file.

|  |
| --- |
|  |

1. Why is there almost no error handling in the ReadFile method?

|  |
| --- |
|  |

**VIDEO: Exception Handling: Try Catch in the same method that throws the exception**

1. Describe (briefly) what code is protected by the try block.

|  |
| --- |
|  |

1. If everything goes fine, which line will the program reach?

|  |
| --- |
|  |

1. When the third number is removed from the file, which lines causes an exception? Briefly describe the cause of the exception.

|  |
| --- |
|  |

1. When the exception happens, where does the program’s execution jump to?

|  |
| --- |
|  |

1. What does the catch block do?

|  |
| --- |
|  |

**VIDEO: Exception Handling: Try Catch in Main**

1. How do you figure out which methods throw exceptions (and which exceptions they throw)?

|  |
| --- |
|  |

1. When the program attempted to convert the string “Mike” to an Int32, what happened?

|  |
| --- |
|  |

1. How does the program figure out which catch block to use?

|  |
| --- |
|  |

**VIDEO: Exception Handling: Throwing your own exceptions; using exception handling in a constructor**

1. What is the syntax for throwing your own exception?

|  |
| --- |
|  |

1. How do you figure out which exceptions that you can throw?

|  |
| --- |
|  |

1. Typically, how do the different exception classes differ from each other?

|  |
| --- |
|  |

1. What are the catch statements based on?

|  |
| --- |
|  |

1. Explain how exception handling useful inside constructors.

|  |
| --- |
|  |

**VIDEO: Big Oh Review**

1. What does a “profiler” do? How is this useful in the software development process? In what way / at what time is this NOT useful for software development?

|  |
| --- |
|  |

1. What level / time during the development life-cycle does the Big Oh notation help you figure out which algorithm to use?

|  |
| --- |
|  |

1. What is (conceptually) important about the “blocks of constant time”?

|  |
| --- |
|  |

1. What determines how much time the linear search function will take?

|  |
| --- |
|  |

1. What is the “exact expression/equation” that’s given in the video for how much time linear search might take? What does each part of the expression mean (I.e., what does the 72 represent? The N? etc)*(Keep in mind that 72 & 113 were clearly picked arbitrarily, in order to have values – this is not necessarily an accurate expression)*

|  |
| --- |
|  |

1. Fill in the blank in the space provided below:  
   At the end of the day, we’re not really interested in the exact amount of time [that this particular implementation] takes, instead we’re interested in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

|  |
| --- |
|  |

1. O(N) is what sort of bound?

|  |
| --- |
|  |

1. Intuitively, what will the running time be for any routine where you end up running through every single element within an array?

|  |
| --- |
|  |

1. What will the running time be for something like the SetValueAt method?

|  |
| --- |
|  |

1. How do you write out “constant time” **IN BIG OH NOTATION?**

|  |
| --- |
|  |

1. According to the chart (within the Big Oh Review document on the website), will linear search or binary search be faster (assuming that you’re able to use either one)? Why (explain briefly and intuitively)?

|  |
| --- |
|  |

1. According to the table that summarizes common running times (within the Big Oh Review document on the website), what are the five most common running times?  
   (You may need to slow down / pause the video, and/or open up the document to read the five common categories)

|  |
| --- |
|  |

1. Is it normal to apply Big Oh notation to individual lines of code within a method?   
   At what “level” is Big Oh notation normally used?

|  |
| --- |
|  |

NOTE: There are no Viewing Quiz questions for the following videos. Instead, you should fill out the following video outlines.  
NOTE #2: There are [directions on the course website that explain how to outline the videos](http://faculty.cascadia.edu/mpanitz/Courses/BIT142/Homeworks/Outlining/index.html). These direction videos are pretty short, so please do watch them!

**Outline for “How To Run The Tests”**

**File:**

* First major topic (replace this with your outline)

**Outline for “How To Debug A Test”**

**File:**

* First major topic (replace this with your outline)

**Outline for “Explanation Of Files”**

**File:**

* First major topic (replace this with your outline)

**Outline for “NUnit Details”**

**File:**

* First major topic (replace this with your outline)

**Outline for “Finding Things Quickly In Visual Studio”**

**File:**

* First major topic (replace this with your outline)

**Outline for “Attributes: 'Values' and 'TestCase'”**

**File:**

* First major topic (replace this with your outline)

**Outline for “Example #1 for getting tests to pass”**

**File:**

* First major topic (replace this with your outline)