**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)

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

# **VIDEO: File I/O (Input) In C#**

## File I/O In C#

1. What does “I/O” mean?

|  |
| --- |
| Input, output |

1. What is **input**? Specifically, where is the data starting out, and where does the data move to?

|  |
| --- |
| Input is reading data from a file and using it in the program. The file being read is where it starts out, it moves into the program to be read and stored. |

1. What is **output**? Specifically, where is the data starting out, and where does the data move to?

|  |
| --- |
| Output is when the data starts out inside the program, and it sent to a file independent of the program, typically as a means of permanent storage. |

1. What are the three steps that you need to do in order to interact with the data contained within a file?

|  |
| --- |
| 1. You need to ask the operating system to open the file. 2. Your program can then read data from the file or write data to the file. 3. Close the file |

1. If we want to move, or delete, or rename (etc.) an entire file, do we need to do the above three steps?

|  |
| --- |
| No, there’s single commands where you can ask the operating system to do those things. |

1. Give an example of how the operating system can optimize I/O when **reading data out of a file**.

|  |
| --- |
| Reading ahead to make future reads faster |

1. Give an example of how the operating system can optimize I/O when **writing data to a file**.

|  |
| --- |
| Program can “write” to a file & keep going, the OS will write the data to a hard drive later. |

1. What does the word ‘directory’ mean?

|  |
| --- |
| A directory is a file folder on a storage device defined by the operating system. |

1. What is a path?

|  |
| --- |
| A path is the route that leads through a directory structure/tree. It tells the operating system where files are stored. |

1. How do absolute paths differ from relative paths?

|  |
| --- |
| A relative path: A path that starts from the working directory, if the file being read from a working directory where the program is running, this will suffice.  Absolute paths: Includes the full path to the file which can be accessed from anywhere. Includes the root directory. |

1. What is the program’s “current working directory”?

|  |
| --- |
| The current working directory is where the program is being executed. It is the file path to where the program is being saved/ and or run from. |

1. When writing a relative path, what is different about the beginning of the path?

|  |
| --- |
| The beginning of the path would not contain a root directory. |

1. Why is it important to tell Visual Studio what the current working directory is?

|  |
| --- |
| The current working directory is where your program is being saved. Without knowing the current working directory of the program it would be impossible to run the program stored there. |

1. When you run the program is run (WITHOUT setting the current working directory) what happens?  
   What does this tell you?

|  |
| --- |
| Without a current working directory the program will display an error. Unhandled exception, System.Io.Directory not found |

1. How do you open up the data file in Visual Studio?

|  |
| --- |
| There are several ways to open a data file in visual studio. If you want to see the raw data directly in visual studio, you can view the current working directory in the solution explorer and open it there to view the data in visual studio. |

## Manipulating Files

1. For all things that we’re going to talk about, what do you need to put at the top of your C# source code file?  
   **NOTE: This is also needed for the code that reads information out of the file (not just for deleting/renaming/etc)**

|  |
| --- |
| Using System.IO;  String path = “Files\\example.txt”;  If (File.Exists(path) )  {  File.Delete(path);  } |

1. What does the backslash character ( **\** )do, in general?  
   What does **\\** mean?

|  |
| --- |
| \ is an escape character  The double \\ translates into a singe \ since the first one is an escape character. The \ demonstrates a change in directory. In the case of the question it means that it is saying to look inside the working directory Files for the file example.txt within that directory. |

1. What is a static method?

|  |
| --- |
| A static method you can call on the class directly, you don’t need to create an object of it. |

1. Why is it useful for File.Exists to be a static method?

|  |
| --- |
| When asking when File exists, you don’t want to create an object of it. |

## Reading Data From A File

1. Give a brief, intuitive description of what are binary files?

|  |
| --- |
| Binary files are files that are filled with machine code (1s and 0s, however some parts of the file can be interpreted as text). Binary files are often used as a general term for files that are not a text file. |

1. When you open up a text file in Visual Studio (or Notepad, etc), what do you see?

|  |
| --- |
| You will see a hex dump of the encoded file. The actual contents of the file and what they mean may be indistinguishable. |

1. Give some examples of text files

|  |
| --- |
| A text file is any file that doesn’t use encoding (plain text) for data storage. Any file that ends with .txt or a sourcecode file can be considered to be a text file. (.cs, .java, .cpp, etc) |

1. How can you check what data is in a text file?

|  |
| --- |
| There are many ways to check what data is a text file. You can use a terminal command like cat. You can open it in visual studio with the solution explorer. You can also open it directly with the OS, or call it using C# so it displays in the terminal. |

1. Both File.OpenText and new StreamReader accept a parameter. What is that parameter?

|  |
| --- |
| fileToRead |

1. Give an example of what might happen if your program doesn’t close a file that it has opened.

|  |
| --- |
| The program will continue to hold the data in memory (often until the program is closed). Garbage collection may not cleanup until a very long time later. |

1. What does the **using(...) { ... }** construct do for you?

|  |
| --- |
| It disposes unnecessary things inside of the object that using is called on. After the code within the second set of parentheses is run. |

1. What is a ‘verbatim’ string? How do these differ from normal C# strings?

|  |
| --- |
| @ symbol  Means that there are no escape characters within the contained string. |

1. What happens when your program tries to open a file that doesn’t exist?

|  |
| --- |
| System.IO.FileNotFoundException |

1. What is an exception?

|  |
| --- |
| An unhandled error that crashes the program. An object representing a runtime error. |

1. Describe a typical way for a program to react to an exception being thrown

|  |
| --- |
| It will print the error code of the exception in the terminal. It will attempt to provide as much information as possible about the exception to identify where the problem originated. |

## Parsing Data In A File

1. What does it mean to ‘parse’ a file?

|  |
| --- |
| Parsing in C# methods means that the method is taking input from a string (extracted from a text file) and returning some other data type. |

1. What is a token?  
   What are they separated by? (Give specific examples of whitespace)

|  |
| --- |
| Tokens are generally any unit that is not whitespace or a comment. They are part of the text of a program. Types include keywords( class), identifiers (program), literals (2) and operators (=).  Separated by whitespace. Hello world 🡪 the space between hello and world is whitespace. |

1. List all the tokens on the following line:  
   **1 “John Smith”**

|  |
| --- |
| 1  “John  Smith” |

1. Why are fixed-width / monospaced fonts useful when looking at text data files (and programming language files)?

|  |
| --- |
| Monospaced font or fixed width, a space that is not blank takes up the exact same amount of space as a blank space. |

1. What does the ReadLine() command do?

|  |
| --- |
| ReadLine() gets the next line in the file. Reads up through and includes the new line. |

**It looks like I lost the drawings from slide 23 (“File Input Answer”) – sorry about that!**(I think that the program is simple enough that you should be able to follow along anyways)

1. What is “leading whitespace”?  
   What is “trailing whitespace”?

|  |
| --- |
| Leading whitespace is white space at the beginning of the line.  Trailing whitespace is at the end of the line. |

1. What TWO THINGS happen when you give Int32.TryParse a string that contains a valid integer?

|  |
| --- |
| 1. Given a valid string parameter changes x parameter to an integer that represents the valid text that it was given.  2. It will also return the value true/ false depending if valid int found. |

1. How do Int32.TryParse, Double.TryParse, and Int16.TryParse differ?

|  |
| --- |
| 32 parses whole numbers  16 parses whole numbers up to 65,000 |

**Note that even though there aren’t a lot of questions about the example programs you should study these programs closely. They’re valuable because they can both explain how to use these concepts (and code) to solve problems and they provide useful models to follow when creating your own programs.**

1. If we want our program to read all the lines in the file (regardless of how many lines there are), how do we set up the program?  
   (You can answer this by briefly describing the differences between this program and the prior program)

|  |
| --- |
| Using (TextReader file = new StreamReader(“Files/numbers.txt”))  {  Double sum = 0.0;  String sLine;  sLine = file.ReadLine();  while (sLine != null)  {  Double dNum;  If (Double.TRyPrase(sLine, out dNum))  {  Sum += dNum;  ConsoleWriteLine(“number = {0}”, dNum);  sLine = file.ReadLine();  }  Cosole.Writeline(“Sum = {0}”, sum);  } |

1. What does ReadLine() return when there’s nothing else in the file?

|  |
| --- |
| null |

1. If we want our program to ignore non-numeric tokens, how do we set up our program?  
   (You can answer this by briefly describing the differences between this program and the prior program)

|  |
| --- |
| Say each line only has one number.  …  Add a delimiter  Char[] delimiters = { ‘ ‘, ‘\t’ };  Using (TexReader fi….  String [] tokensFromLine = sLine.Split (delimiters, StringSplitOptions.RemoveEmptyEntries);  Foreach (string token in tokensFromline) {  Double dNum;  …  … |

1. What is a ‘delimiter’ character?

|  |
| --- |
| A delimiter is a sequence of one or more characters used to specify the boundary between separate, independent regions in plain text or other data streams. An example of a delimiter Is the comma character, which acts a field delimiter in a sequence of comma-separated values. |

1. What does the .Split() method do?

|  |
| --- |
| The split method is built into the basic string in c#  Passes in delimiter array in the parameter.  Says we want to split the string into tab character or individual spaces. |

1. What does the ReadToEnd() method do?

|  |
| --- |
| Reads every single line of the file out at once into memory as a single string. |

1. What is a downside to reading the entire file into memory all at once?

|  |
| --- |
| Poor memory performance. It also assumes that the stream knows when to reach an end, otherwise it will continue to read indefinitely. |

# VIDEO: File I/O (Output) In C#

1. When talking about file output, where is the data going to?

|  |
| --- |
| In file output it is taking the data that is accessible inside the program and writing the data to a file that is accessible outside the program. |

1. We’re going to be using StreamWriter objects to write to files, using several methods.  
   Which methods are these, and where have you seen (and used) these methods before?

|  |
| --- |
| Console.out  WriteLine  Dispose()  Yes I have both seen and used these methods before. |

1. When printing to a file, will the output immediately be written to the file on the disk?  
   If not, where is that information being stored?

|  |
| --- |
| No, it will not be immediately written to a file on a disk.  It will be stored in memory until it is time to store it on permanent storage on the storage medium. |

1. When will the information be written to the disk?

|  |
| --- |
| When you call output.Dispose(); |

1. C# provides a way to ensure that the Dispose() method is called. Copy the example code from the slides and put in a brief comment explaining what each line does.

|  |
| --- |
| StreamWriter output = new StreamWriter( “out.txt”); // creates a new streamwriter object  Output.WriteLine(“Hello, file!”); // writes a new line to memory  Output.WriteLine(“This is a second line of output.”); // writes a second line to memory  Output.Dispose(); // stores both lines stored in memory to a txt file on the disk. |

1. Why will we NOT be using the C# using construct in these slides?

|  |
| --- |
| Because we want to learn how to use the Dispose(); pattern. Construct does it automatically. |

1. If you create a new StreamWriter object (in order to write data to a file) and the file does NOT exist, what happens?

|  |
| --- |
| If it does not exist, it will attempt to create the file. It will check to see if you are allowed to check the file, whether it is allowed in that location, etc. |

1. If you create a new StreamWriter object (in order to write data to a file) and the file DOES exist, what happens?

|  |
| --- |
| If there is already a file there, it will be overwritten. |

1. How can you check that your program has successfully written the data to the file?

|  |
| --- |
| You will have to open the file and view contents. The output is not written to the console when the process is completed. |

1. For the ‘Hours’ question, put in a quick explanation of what each part means:  
   *123 Kim 12.5 8.1 7.6 3.2*  
   E.g., explain what the ‘123’ means, what the ‘Kim’ means, etc.

|  |
| --- |
| 123 = ID  Kim = String token person name  12.5 = one or more separate numbers for hours worked.  Compute the total numbers of hours worked / day |

1. In the answer to the Hours problem, explain what this line does, and how it works:  
   *while( (sLine = input.ReadLine() ) != null )*

|  |
| --- |
| Calls readline and assigns it to the sLine string variable. If ReadLine has another line to read, it will assign a non-null value to sLine. If there is no more lines left, it will assign null to sLine. Then the entire check will evaluate to null (check == false) 🡪 does the next things after the loop.  Reads all the lines in the file. |

1. Why is it necessary to check if count is zero?

|  |
| --- |
| It is necessary because it tells the program whether there were any valid hours worked that were found. If they did not parse as doubles, it will print an error message saying that there weren’t any hours worked.  Attempt to read the next one, if not, it will skip/end the loop. |