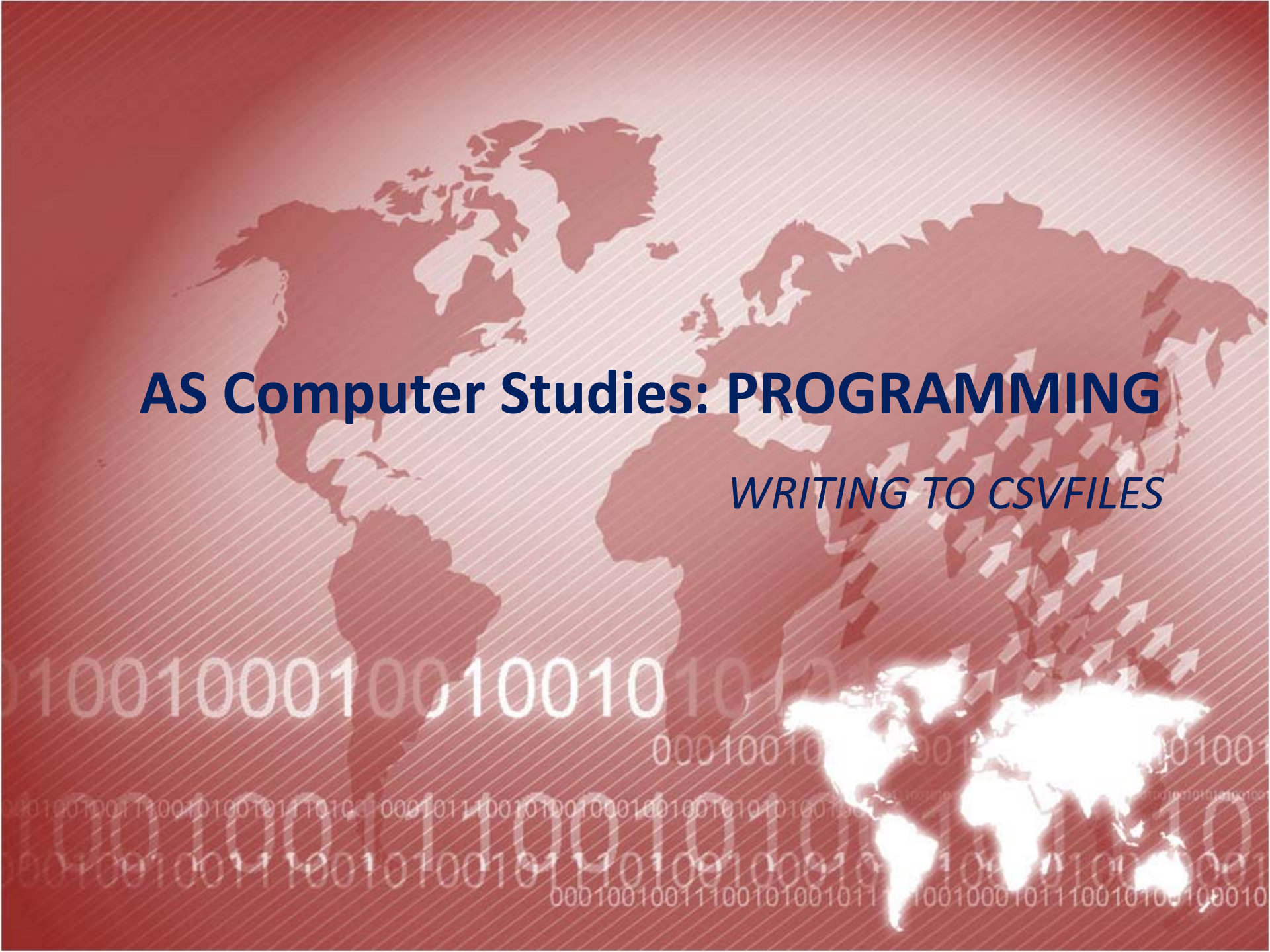


# AS Computer Studies: PROGRAMMING

*WRITING TO CSVFILES*

# Starter

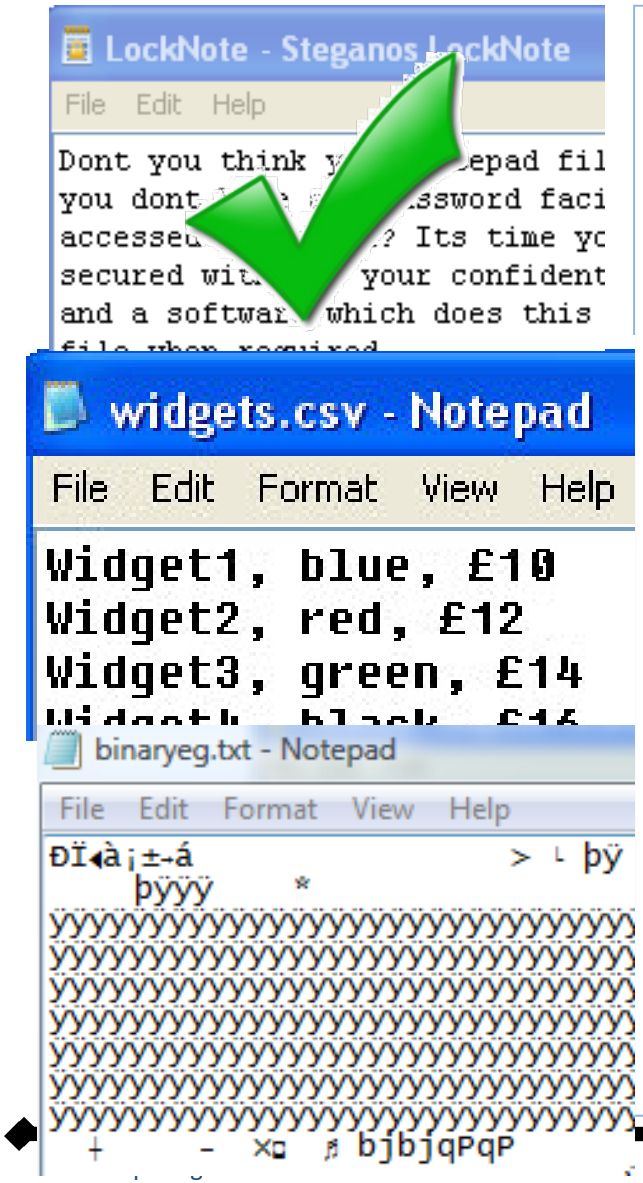
- Write a program which asks for the name and age of 3 students and writes these details to a text file.

- After you have done this, look at (from the C drive or the path you've defined in the program) your text file. I will ask you what the problem is with the file structure.

# Objectives

- Understand the different ways in which a computer can **read** and **write** data **to file**.
- Become familiar with the coding constructs of **saving and reading data**.
- Use saving and reading within your program.

# Recap: File Types




- **Text File (extension .txt).** Basically a set of character values which are in a document line by line. You could open it in Notepad or similar application and read what was inside it.
- **CSV File (extension: .csv).** A comma-separated values file. Like a text file, with all related information on one line (e.g. all information about a customer: name, age, debt etc), with **the fields separated by commas**.
- **Binary file – (extension varies).** A application, which is opened in say notepad would not be comprehensible to the human eye.

# What is the correct order?

A. Add "**Imports System.IO**" to your General section.

B. Bind the channel to the file (i.e. tell the computer what file you are going to write to, using your writing channel. **fileWriter = New streamwriter(fileName)**

C. Prompt the user to enter something in the console using the writeline command
- Store what the user types in a variable. E.g. **strEntry = console.readline()**

D. Declare a channel for writing (add writing functionality using StreamWriter
**Dim fileWriter as StreamWriter**

E. Write what was written in the console to file **fileWriter.writeline(strEntry)**

F. Close the file. **fileWriter.close**

G. Specify the file you want to write to (in a variable or constant – useful to put at top of sub).
**Dim filename as string** **fileName = "c:\writerExample.txt**

# Recap: Writing text files

1. • Add "**Imports System.IO**" to your General section.
2. • Declare a channel for writing (add writing functionality using StreamWriter
   **Dim fileWriter as StreamWriter**
3. • Specify the file you want to write to (in a variable or constant – useful to put at top of sub).
   **Dim filename as string** **fileName = "c:\writerExample.txt**
4. • Bind the channel to the file (i.e. tell the computer what file you are going to write to, using your writing channel. **fileWriter = New streamwriter(fileName)**
5. • Store values needed to be written to disk into variables
   **(strName = console.readline() or strName = "Chris")**
6. • Write what was written in the console to file **fileWriter.writeline(strEntry)**
7. • Close the file. **fileWriter.close**

# Recap: Reading text files

1. Add "**Imports System.IO**" to your General section.
2. Declare a channel for reading(add writing functionality using StreamReader
   **Dim fileReader as StreamReader**
3. Specify the file you want to read from (in a variable or constant – useful to put at top of sub). **Dim filename as string** **fileName = "c:\writerExample.txt**
4. Bind the channel to the file (i.e. tell the computer what file you are going to write to, using your writing channel. **fileReader = New streamReader(fileName)**
5. Read each line from the file, until the file is empty. (**Do Until …. FileReader.EndofStream) … Loop**.
   Use **FileReader.Readline()** within the loop to read a line. E.g. **strEntry = filereader.readline()**
6. Display the contents of the variable on screen: **console.writeline(strEntry)**
7. Close the file. **fileReader.close**

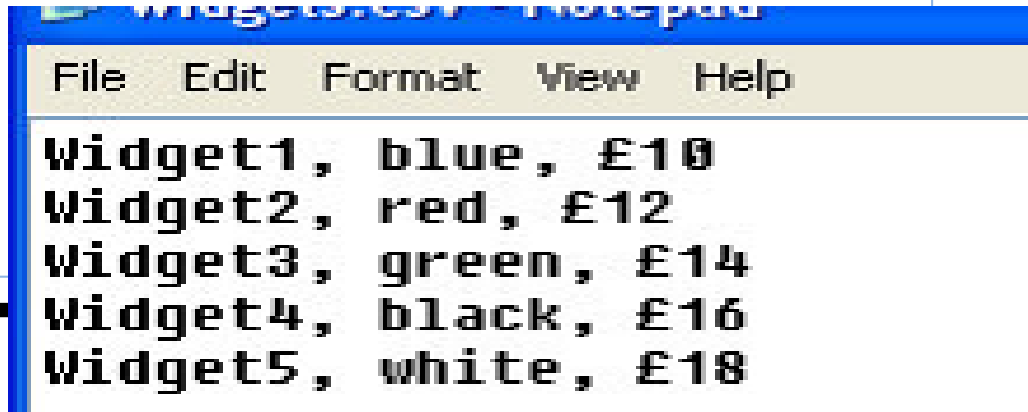
# RECAP: Preparing to Write

- Your program needs **instructions from a library**.
- It needs to know HOW to read/write to files.
- At the top of your module write:

**Imports System.IO**

# Comma Separated Values

- What if we want to keep a lot of **information about one entity**?
- What if we want to keep customer details (name, age, address, DOB, amount owed) etc all in one file?
  - there are 5 records, each containing three pieces of information (the widget name, colour ad price).
- Notice that they are **related**, as they are each on their own distinct line, separated by commas.
- Useful for Mail Merging, Importing into other programs or use in a database.
- Let's look at an example.



# How to make a CSV File

- Exactly the same as a normal text file.
- You must just prepare the item being written to file.

E.g. Say you had three variables:

| strName | strStreet | strTown |
|---|---|---|
| David Jones | Dove Street | York |

The easiest way is to **concatenate** them, **separated by commas** into one variable.

E.g. strEntry = strName & “,” & strStreet & “,” & StrTown

(strEntry would then be: “David Jones,Dove Street,York”)

You could then write the line to disk: fileWriter.WriteLine(StrEntry)

# Today's Task

- Handout first.
- Continue the questions in Task 16.

# Objectives

- Understand the different ways in which a computer can **read** and **write** data **to file**.
- Become familiar with the coding constructs of **saving and reading data**.
- Use saving and reading within your program.

# Required Reading

- Each week you will be given required reading.
- If you fail to do this, you will 100% find the lessons which follow it EXTREMELY difficult.
- Before next lesson you should have read:
- **Pre-reading: 39-45**

# Plenary

- What is the difference between a text file and a CSV file?
- Why would you use a text file?
- Why would you use a CSV file?
- What problems still exist with both files (think about security)?