Stephen Wood

Notre Dame Catholic Sixth Form College | Computer Science Dept.

A-Level Computer Science

Year 1 – Programming Challenges

Coding Challenges – **Volume 5**

Coding focus:

**File Handling &** **Subroutines**

Name:

Class:

**Coding Challenges – Volume FIVE**

Table of Contents

[Program 1 – Read in and display a file. 2](#_Toc31300087)

[Program 2 – Word Count from a file. 3](#_Toc31300088)

[Program 3 – Filtering students by subject. 4](#_Toc31300089)

[Program 4 – Writing and storing Prime Numbers. 5](#_Toc31300090)

[Program 5 – The Fibonacci Sequence. 6](#_Toc31300091)

[Program 6 – Encrypted Song. 7](#_Toc31300092)

[Program 7 – Encrypted Quote – it happened again. 8](#_Toc31300093)

[**The Hard Bit 9**](#_Toc31300094)

[Program 8 – Encrypted (again) but different. 10](#_Toc31300095)

[Program 9 – Wonky Encryption. 11](#_Toc31300096)

# Program 1 – Read in and display a file.

Write a program in C# to read in a text file (Students.txt), the program should read in the file and then successfully display each student’s name on a new line in the console window.

Test your Program using the following Text Files: **Students.txt**

Program Code:

Test Evidence: (screenshot of it working)

# Program 2 – Word Count from a file.

Write a program in C# to create a function to read in a text file, the program should then perform a word count on the contents of the Text File.

Test your Program using the following Text Files: **Gettysburg.txt**

Program Code:

Test Evidence: (screenshot of it working)

# Program 3 – Filtering students by subject.

Write a program in C# to read in a text file containing information about students and subject choices (StudentSubjects.txt). Your program should output a list of all the students that study a certain subject, this will be entered by the user.

Test your Program using the following Text Files: **StudentSubjects.txt**

Test evidence should show the students who study:

* **Maths**
* **Computing**
* **Psychology**

Program Code:

Test Evidence: (screenshot of it working)

# Program 4 – Writing and storing Prime Numbers.

Write a program in C# to make use of a function that will output all of the Prime numbers in a given range, chosen by the user. The user should be able to select both a start and end number for the range.

The program will then write these numbers sensibly into a Text File.

**Sample Output:**

Test the program using: **min - 7, max - 50**

(show the text file) **min - 14, max - 100**

Program Code:

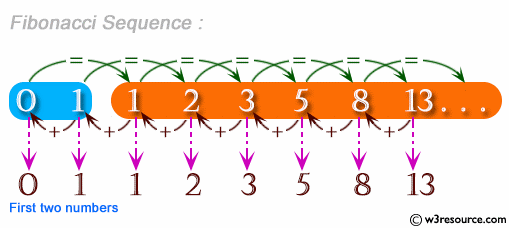
Test Evidence: (screenshot of it working)

# Program 5 – The Fibonacci Sequence.

Write a program in C# to create a program using subroutines to display the ‘**n**’ (chosen by the user) number Fibonacci sequence. The program will then be able to write these values into a text file.

**E.g.** Input a number: 10

The 10 Fibonacci numbers are: 0 1 1 2 3 5 8 13 21 34



Test the program by entering: **10, 15, 17 and 20**

Program Code:

Test Evidence: (screenshot of it working)

# Program 6 – Binary File Writer using a ‘Struct’.

Write a program in C# to allow the user to enter some details about an accident claim being made. The information they will need to store in the ***Struct*** and output to the Binary File will be:

* Claimant (person’s) Name
* Car Registration
* Date of the accident
* Location where it happened
* Number of vehicles involved

The main aim is to store all of this information in to a Binary File, so it can be quickly read by the system later.

Test your Program using the following information:

* **Name:** Dave Jones
* **Car Reg:** FF19 STG
* **Date:** 19/01/2019
* **Location:** Leeds
* **Vehicles:** 4

Program Code:

Test Evidence: (screenshot of it working)

# Program 7 – Encrypted Song.

Write a program in C# to read in a text file that has been encrypted, the key is somewhere in the range of 1-25. The programmer has forgotten though, so you will need to develop a system that is able to allow different keys to be entered or to brute force its way through each key.

The main aim is to decrypt which song has actually been encrypted.

The program will need to be make use of both upper and lowercase letters, no symbols have been encrypted though.

Test your Program using the following Text File: **Code1.txt**

Program Code:

Test Evidence: (screenshot of it working)

# Program 8 – Encrypted Quote – it happened again.

Write a program in C# to read in a text file that has been encrypted, the key is somewhere in the range of 1-25. The programmer has forgotten again!!

So you will need to develop a system that is able to allow different keys to be entered or to brute force its way through each key.

The main aim is to decrypt which movie quote has been encrypted this time.

The program will need to be make use of both upper and lowercase letters, no symbols have been encrypted though.

Test your Program using the following Text File: **Code2.txt**

Program Code:

Test Evidence: (screenshot of it working)

# 

# The Hard Bit

This is the bit where the spicy challenges live…

****

# Program 9 – Encrypted (again) but different.

Write a program in C# to read in a text file that has been encrypted, the key is somewhere in the range of 1-25. The programmer has forgotten again!!

This time looks different though, like it has been broken into a weird pattern.

The program will need to be make use of both upper and lowercase letters, no symbols have been encrypted though.

Test your Program using the following Text File: **Code3.txt**

*Rumour has it that this challenge is a bit hairy or something close to that…*

Program Code:

Test Evidence: (screenshot of it working)

# Program 10 – Wonky Encryption.

Write a program in C# to read in a text file that has been encrypted, the key is somewhere in the range of 1-25. The programmer has forgotten AGAIN!!

This time looks different though, like it’s all gone a bit wonky!

The program will need to be make use of both upper and lowercase letters, no symbols have been encrypted though.

Test your Program using the following Text File: **Code4.txt**

*This challenge seems to be difficult, don’t let it flip you out…*

Program Code:

Test Evidence: (screenshot of it working)