## Data Science**: ePortfolio**

|  |  |
| --- | --- |
| First Name | Kevin |
| Last Name | Gemson |
| Student Number | EC2111447 |
| Email Address | ec2111447@edinburghcollege.ac.uk |

## Lab 1: Title – **Data Capture & Modelling**

## Date Completed: **31/03**/2022

**Weekly summary, complete the following:**

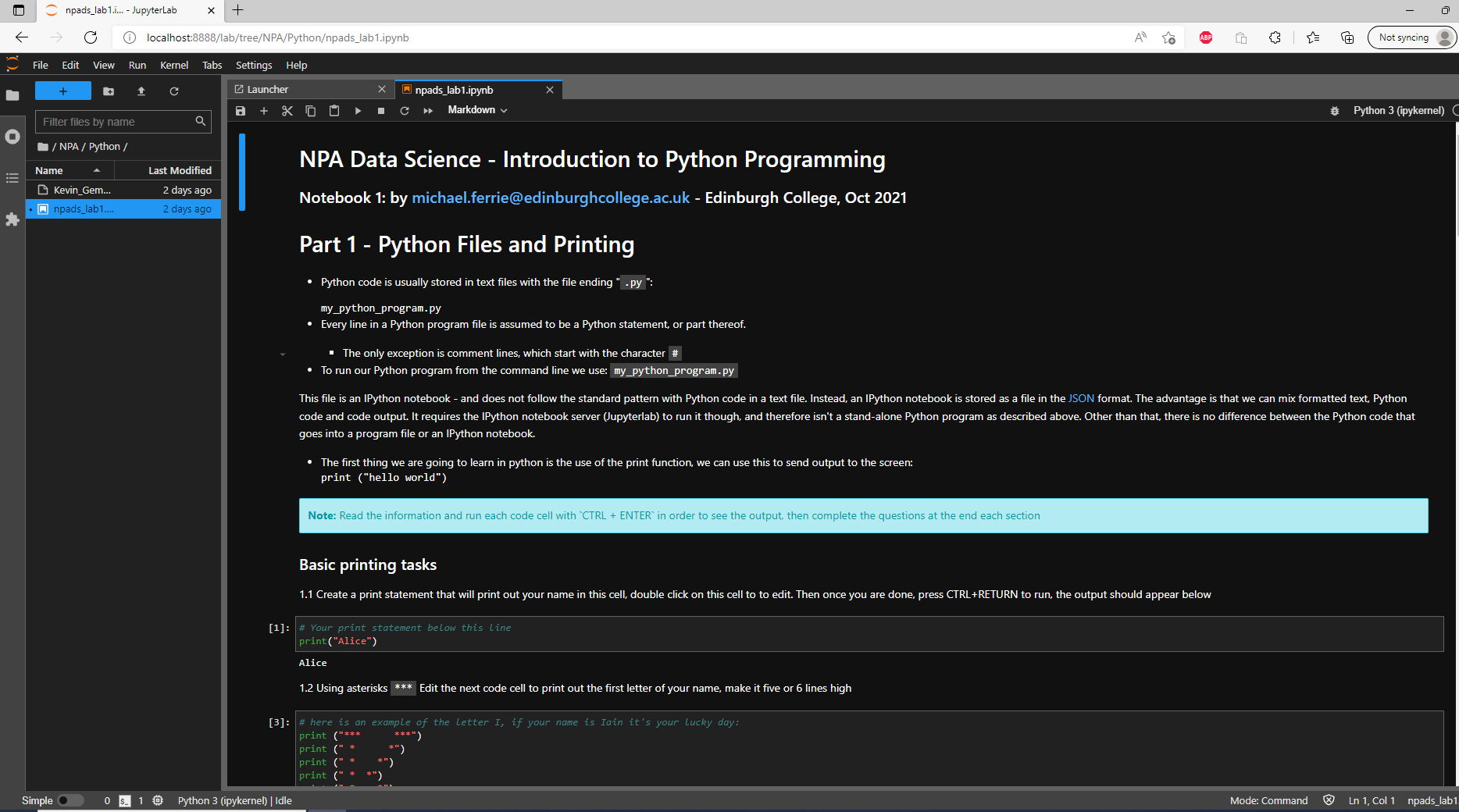
This week I learned … how to install Jupyterlab and get Python scripts running on it. From the PDF, I learned about different data capture techniques, as well as different ways of cleaning and validation. Also reviewed Python basics from the YouTube video

In the lab we … installed Jupyterlab and ran some basic Python commands

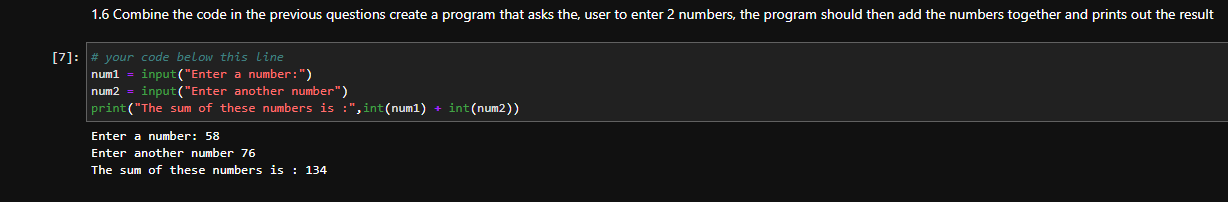
I enjoyed… learning about different data capture and cleaning techniques; having worked in a company that handles and processes large volumes of data, it is interesting to see the different techniques available, many of which I have not come across before. The company has tended to use traditional methods of handling data in the past, and it will be interesting to see which of these I may come across in the future.

I found it challenging when… I had some issues getting Jupyterlab to work; I had used Python before, running scripts through Visual Studio Code, but Jupyterlab works in a different way and took a bit of getting used to

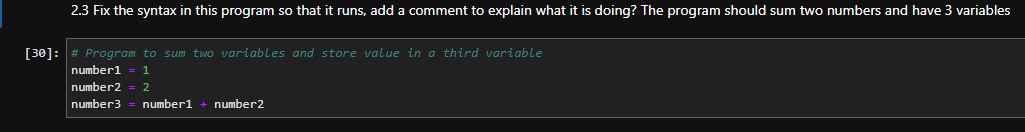
* A screenshot of Jupyterlab running on your computer



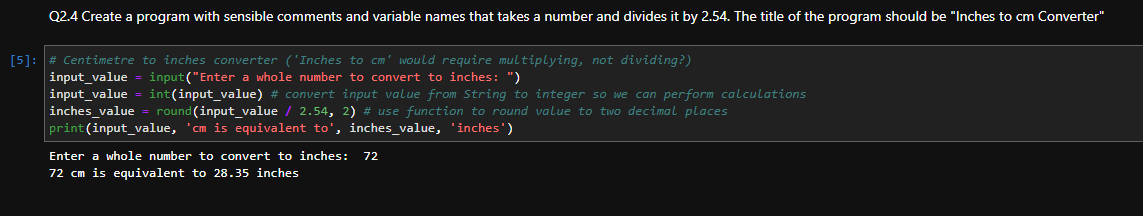
* A screenshot of your answer for question 1.6



* A screenshot of your answer for question 2.3



* A screenshot of your answer for question 2.4



## Lab 2: Title – **Data Analysis & Statistics**

## Date Completed: 03/04/2022

**Weekly summary, complete the following:**

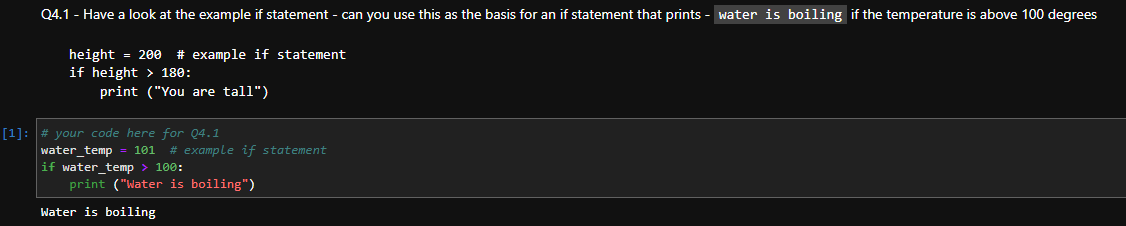
This week I learned … about the different steps in Data Analysis and how these can lead to allowing us to making predictions about future trends.

In the lab we … revised some basic operators and ran some scripts to demonstrate handling of different data types and conditionals

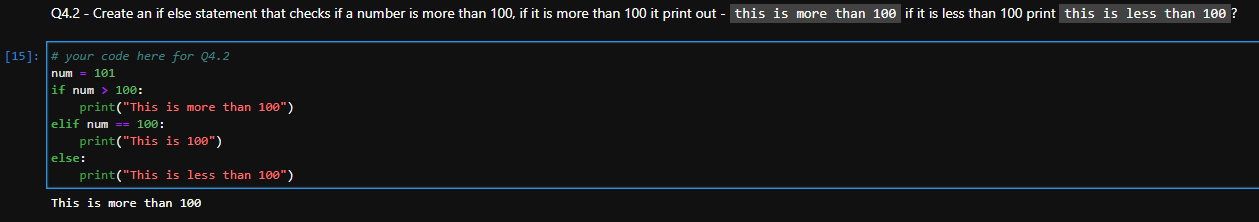
I enjoyed… revising some of the basic commands and taking on some of the programming challenges.

I found it challenging when … trying to get the appropriate data type for many of the scripts (e.g. converting string to int and performing division to get a float). Also considering the different options when using conditionals to ensure all outcomes are catered for.

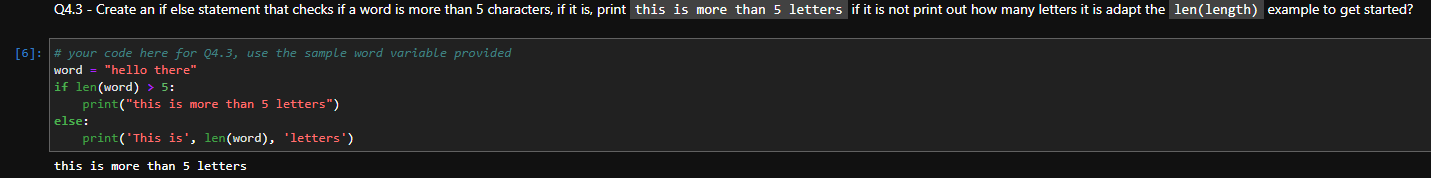
* A screenshot of the code in the code cell and the output for question 4.1



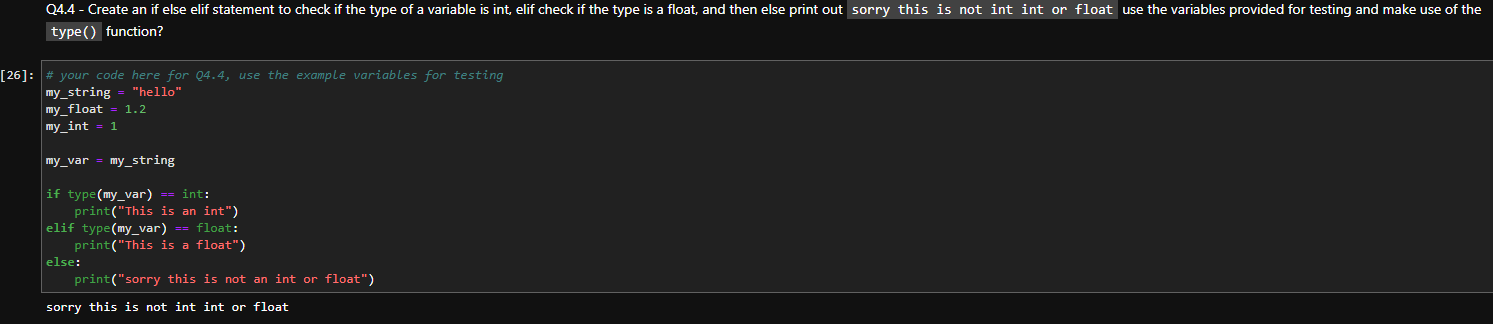
* A screenshot of the code in the code cell and the output for question 4.2



* A screenshot of the code in the code cell and the output for question 4.3



* A screenshot of the code in the code cell and the output for question 4.4



## Lab 3: Title – Data **Visualisation and Storytelling**

## Date Completed: **07/04/2022**

**Weekly summary, complete the following:**

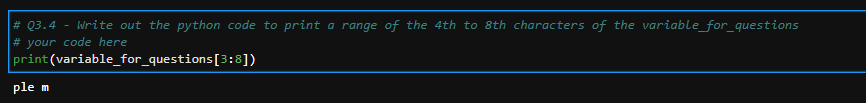
This week I learned… about data visualisation and the different options for presenting data based on what the data is trying to present. In Python, we learned about extracting individual elements from lists and also program control – for loops in conjunction with if/else statements.

In the lab we… ran a number of scripts to extract individual characters of ranges of characters from a string variable; also ran some scripts to take input data and process differently based on the values provided. Final challenge was to randomly generate a data value and handle the output based on the value.

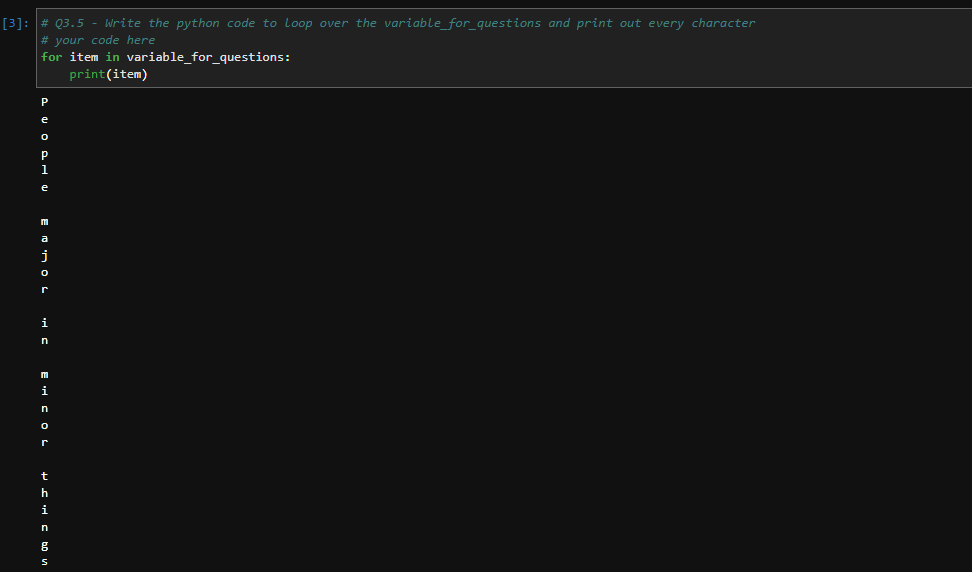
I enjoyed… the challenge of working through the different possible outcomes and writing code to handle these.

I found it challenging when… trying to work out the indexing values - I have used other programming languages that index starting at 1 and where ranges are inclusive, so ensuring I am extracting the correct values can take some thought.

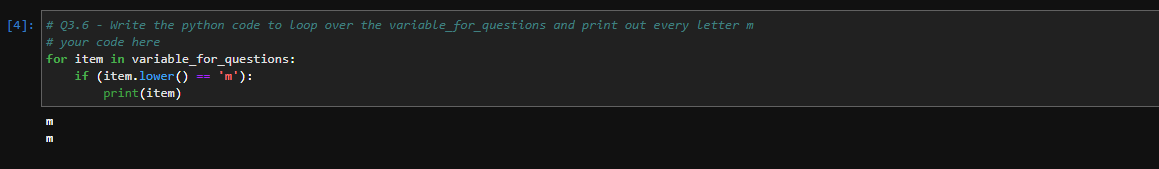
* A screenshot of the code in the code cell and the output for question 3.4



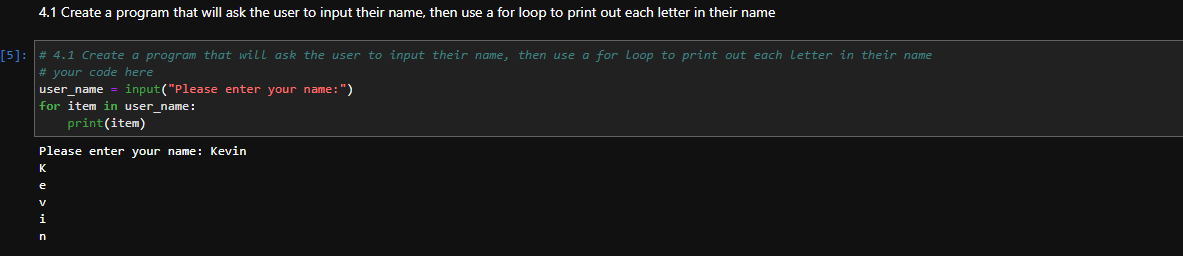
* A screenshot of the code in the code cell and the output for question 3.5



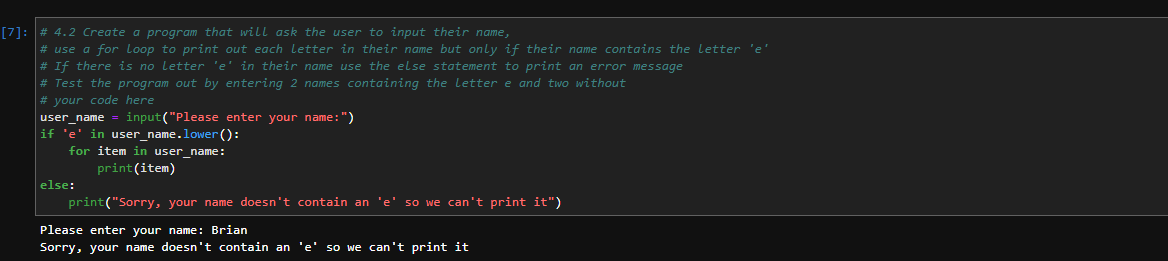
* A screenshot of the code in the code cell and the output for question 3.6



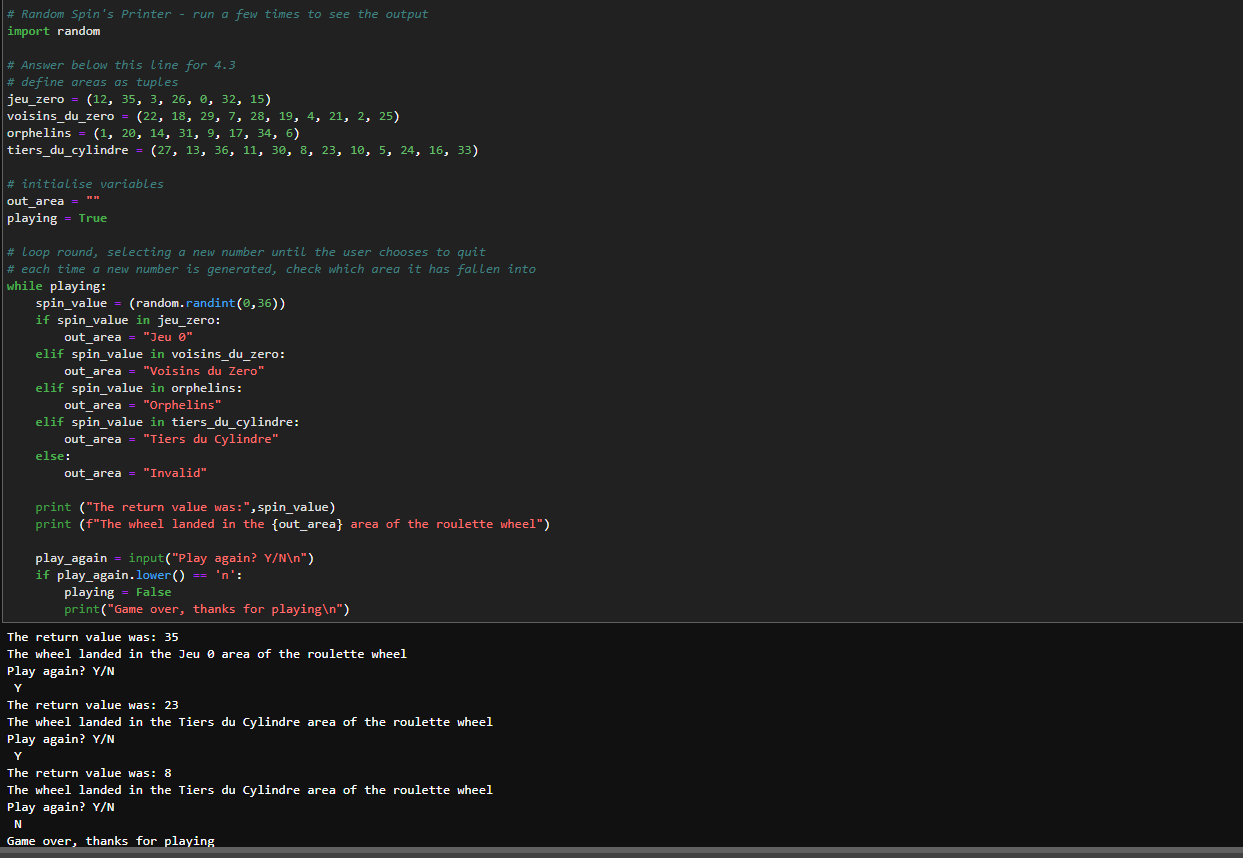
* A screenshot of the code in the code cell and the output for question 4.1



* A screenshot of the code in the code cell and the output for question 4.2



* A screenshot of the code in the code cell and the output for question 4.3



## Lab 1: Title – Computer Programming 1

## Date Completed: 12**/05/2022**

**Weekly summary, complete the following:**

This week I learned … about the ways of presenting an algorithm, both using decomposition notation and flowcharts.

In the lab we … took a number of programming challenges and presented these as algorithms in both formats

I enjoyed… the process of taking lines of code and working out the steps involved in producing an algorithm. Normally I would try writing lines of code first rather than working things out in a document, so it is a different way of programming.

I found it challenging when… trying to follow the rules for producing a flowchart in particular, i.e. where to put a decision, whether something is an input/output or a step, how to get two outputs from a decision etc. Found it much easier to write the decomposition notation.

An explanation of two different techniques for representing algorithms, how can we express an algorithm, why would we use these techniques?

- The two different techniques are decomposition notation and flowcharts. Decomposition uses text to declare the individual steps in producing the end result, and breaks these down into substeps using indentation to indicate a substep. Flowchart represents the steps in the algorithm as a chart using standard notation for presenting input/output, decisions, processes and start/stop. The first technique is more informal and probably used at a very early stage when trying to work out how to present all the steps involved in completing a task, and is easier to alter, e.g. to add or remove steps. The second type is a more formal document using standard notation and is probably used at a later stage in the design process, i.e. after the more informal earler stage of working out what steps are required. It is more difficult to alter and is used to present a plan for completing a task to others such as coders who may not be involved in the planning stage.

An explanation of the following three concepts:

- algorithmic efficiency: Algorithmic efficiency is a calculation of the amount of programming resources required by an algorithm to complete all the required steps. It uses mathematical formulae to estimate the amount of resources such as processing power, memory and time to establish whether an algorithm is likely to perform within a required set of constraints when run as code. It may help determine which of a number of possible algorithms are preferred, depending on which resource is considered more valuable.

- syntax: Syntax is the set of rules which determine how words and symbols are interpreted by a computer language and combined into a program. Each language has its own set of rules unique to it which determine how code is interpreted and expression or line of code must be presented in accordance with these rules, otherwise the program will not be understood by the language software and will present an error when a user tries to run or compile it.

- semantics: In programming, semantics refers to the meaning of the code within a program. While the code may be syntactically correct (i.e. it follows the required syntax rules for the programming language, and is accepted by the software as executable code), it may still be semantically incorrect if it does not consider whether each element of the code makes sense and carries out all the required steps in the correct order. When a semantic error occurs, normally the code will execute successfully but will not give the expected result to the user.

The flow chart and decomposition for 2.2, 2.3 and 2.4

2.2 The next algorithm to create is one to check if a word contains a vowel ‘aeiou’:

* Write out your algorithm in decomposition notation.

> Read input word, set ‘match’ flag to ‘False’

> Check each letter to see if it is in ‘aeiou’

>> read next letter of word

>> if no more letters, exit loop

>> else, check to see if letter is in ‘aeiou’

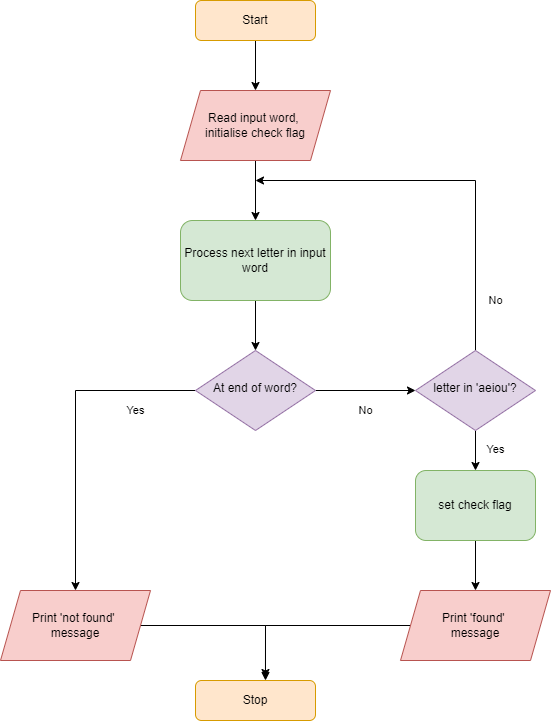
>>> if yes, set ‘match’ flag to ‘True’ and exit loop

> Print output message

>> if ‘match’ flag is ‘True’, print ‘match’ message

>> else print ‘no match’ message

* Draw your algorithm as a flowchart.



2.3 This algorithm should be an enhancement of the previous, in 2.2 you created an algorithm to determine if a word contained a vowel, can you improve this and say how many vowels - if any - a word contains?

* Write an algorithm that is an improvement of the algorithm in 2.2 in decomposition notation; this should now calculate the number of vowels in a word.

> Read input word, set counter to zero

> Check each letter to see if it is in ‘aeiou’

>> read next letter of word

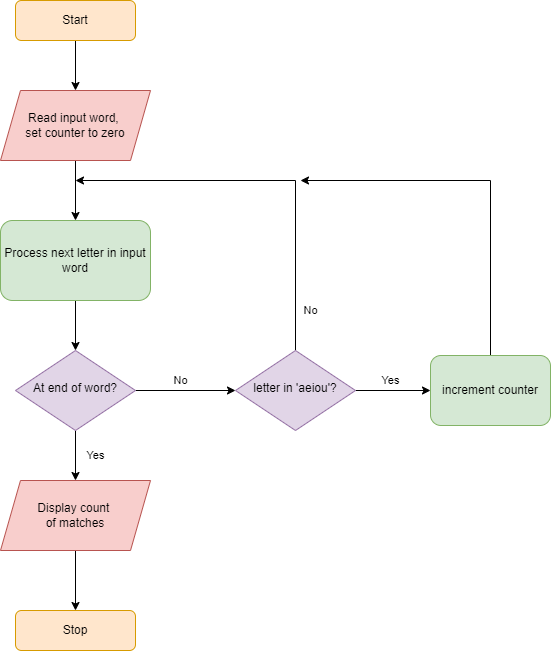
>> if no more letters, exit loop

>> else, check to see if letter is in ‘aeiou’

>>> if yes, increment counter

> Print output message containing count of matches

* Write a flowchart to represent your algorithm.



2.4 The next algorithm we want to create is an algorithm to check if a number can be divided by 3 without a remainder. The algorithm should then check if the number can be divided evenly by 4. Then the algorithm should check if the number can be divided evenly by both 3 and 4.

* Write out your algorithm in decomposition notation.

> Read input number, initialise booleans to ‘False’

> Check number is divisible by 3

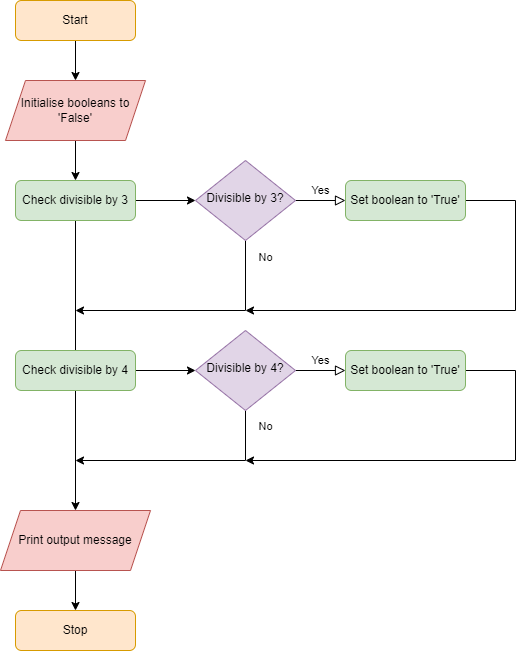
>> if modulus of input number divided by 3 is equal to zero, set ‘divisible by 3’ boolean to ‘True’

> Check number is divisible by 4

>> if modulus of input number divided by 4 is equal to zero, set ‘divisible by 4’ boolean to ‘True’

> Print output message containing ‘true’ or ‘false’ result for both booleans

* Draw your algorithm as a flowchart.



## Lab 2: Title – Computer Programming 2

## Date Completed: 19**/05/2022**

**Weekly summary, complete the following:**

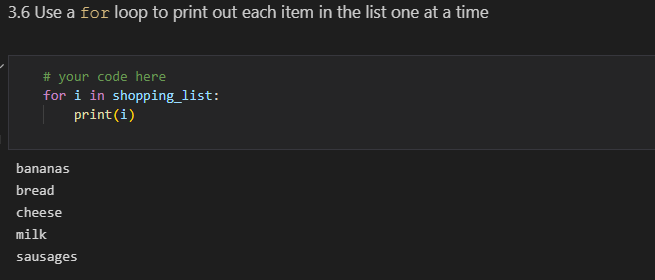
This week I learned … more basic elements of the Python language such as data types, operators and lists.

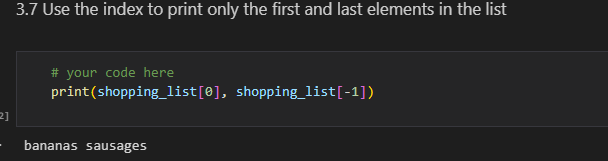
In the lab we … generated and ran code to carry out a few challenges using loops and list elements

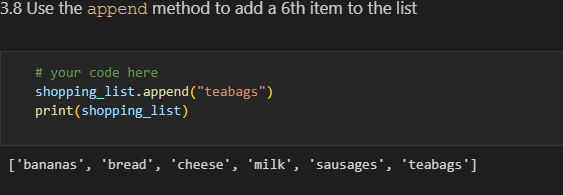
I enjoyed… the challenge of trying to work out some of the more difficult tasks, such as combining conditionals with loops to only access certain list elements

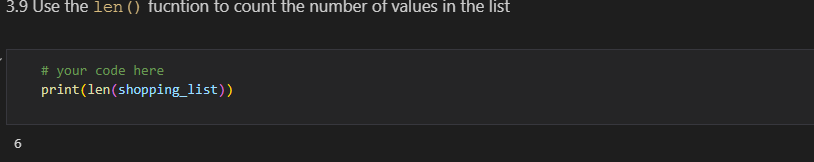
I found it challenging when… trying to work out the most efficient way of accessing some of the list elements, particularly when accessing elements at the end of a list rather than at the start.

* Your answers for questions 3.6,3.7,3.8,3.9

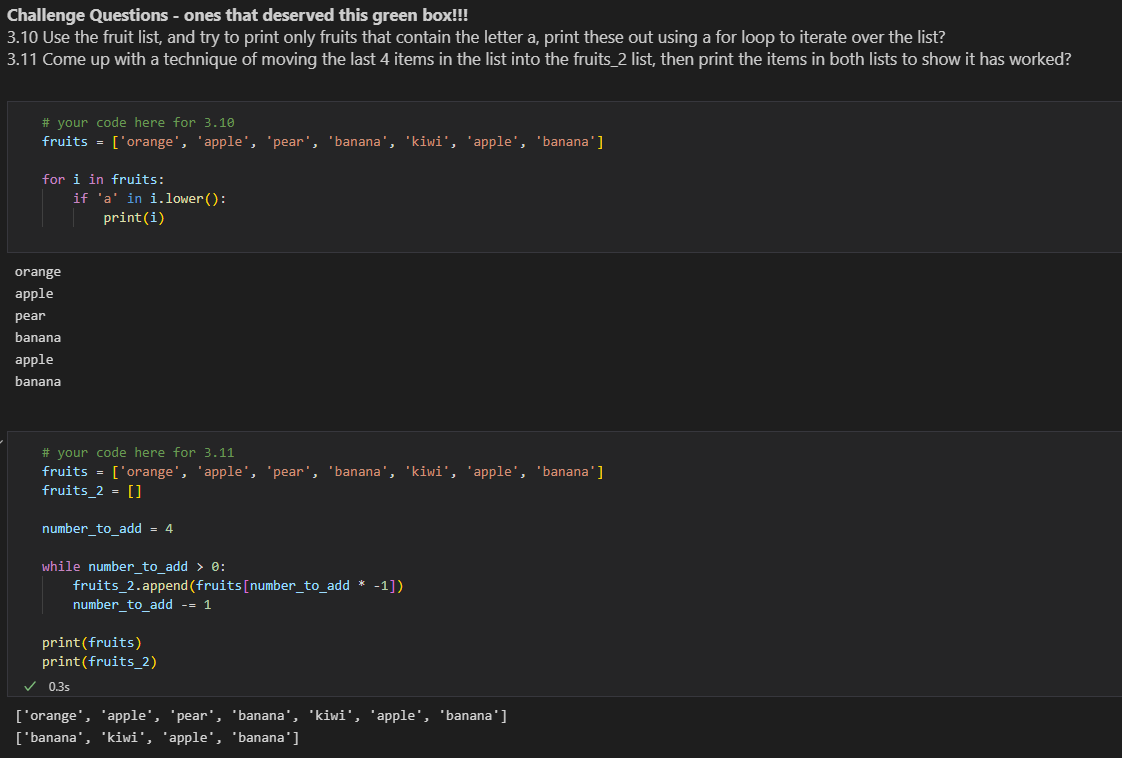








* Your answers for the challenge questions 3.10 and 3.11



* An explanation of the following programming concepts:
* **Iteration**: this refers to completing a task a number of times, for example executing a piece of code a specific number of time or executing it until a certain condition is identified.
* **data types:** these are a way of identifying the ‘type’ of an element of data. The different data types have their own characteristics and are used differently by the programming language, for example numeric data types can have mathematical calculations performed on them while character data types cannot.
* **operators:** operators are symbols used by the programming language to identify what action to take with variables and values in a program, such as assigning a value or performing a mathematical operation.
* An explanation of the order of evaluation in python, in what order are arithmetic operators evaluated

In Python, operators are evaluated using the following order:

Order of operation - PEMDAS

1. Parentheses ( )

2. Exponent \*\*

3. Multiplication \*

4. Division / // %

5. Addition +

6. Subtraction -

After the PEMDAS order goes left to right. To override an order, an expression should be enclosed in parentheses.

* An explanation of the purpose of internal documentation (comments) in a program

The purpose of comments in a program are to aid readability and understanding of the code. A comment is an explanation or annotation in the source code, readable only to the programmer. They are added to make the source code easier for humans to understand, and are generally ignored by compilers and interpreters.

## Lab 3 Title – Computer Programming 3

## Date Completed: 26**/05/2022**

**Weekly summary, complete the following:**

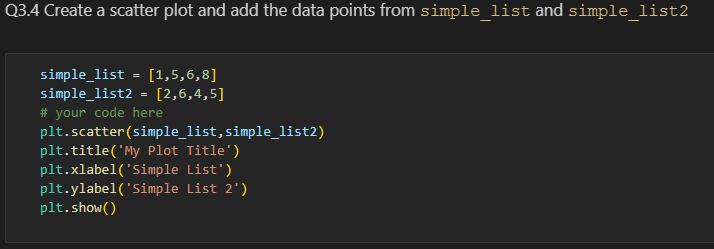
This week I learned … about the libraries available to be imported into Python to present data in a visual way. Also learned about using dataframes and the operators available such as sorting and extracting data to use as input to the visual presentation libraries.

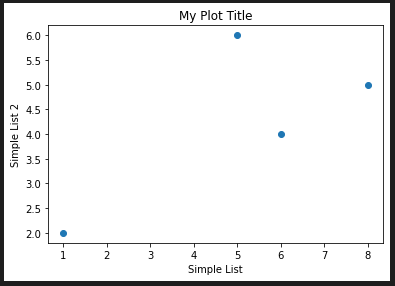
In the lab we … wrote and ran a number of pieces of code to present data in different forms, such as line graphs, bar charts, scatter graphs etc.

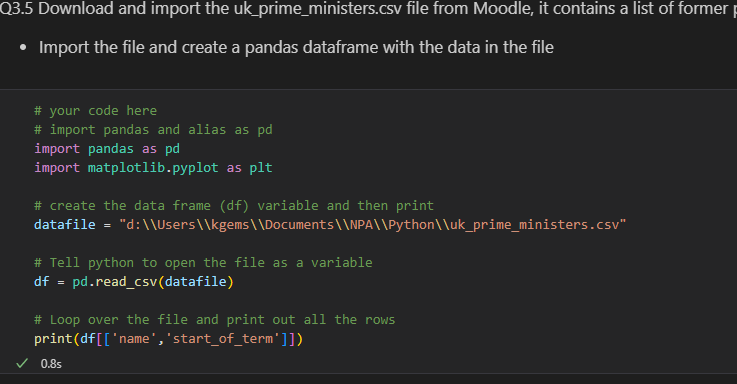
I enjoyed… seeing the complex output that can be generated from a few lines of code using imported libraries

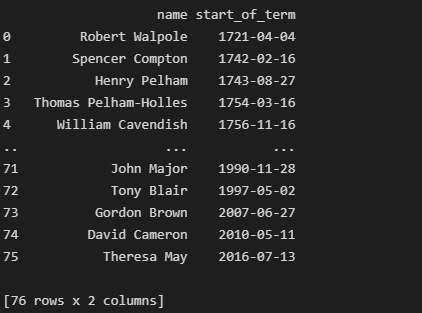
I found it challenging when… trying to work out how to use the different parameters to resolve issues with the output, for example, getting sizing right, setting colours and fonts, etc.

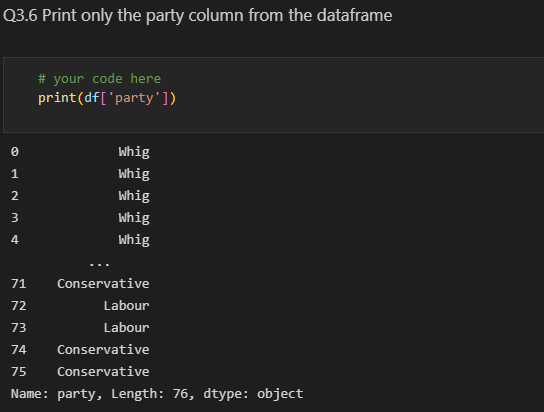
* A screenshot of the code and the output from Q3.4-3.7

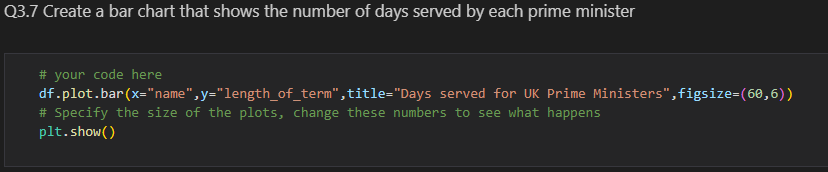


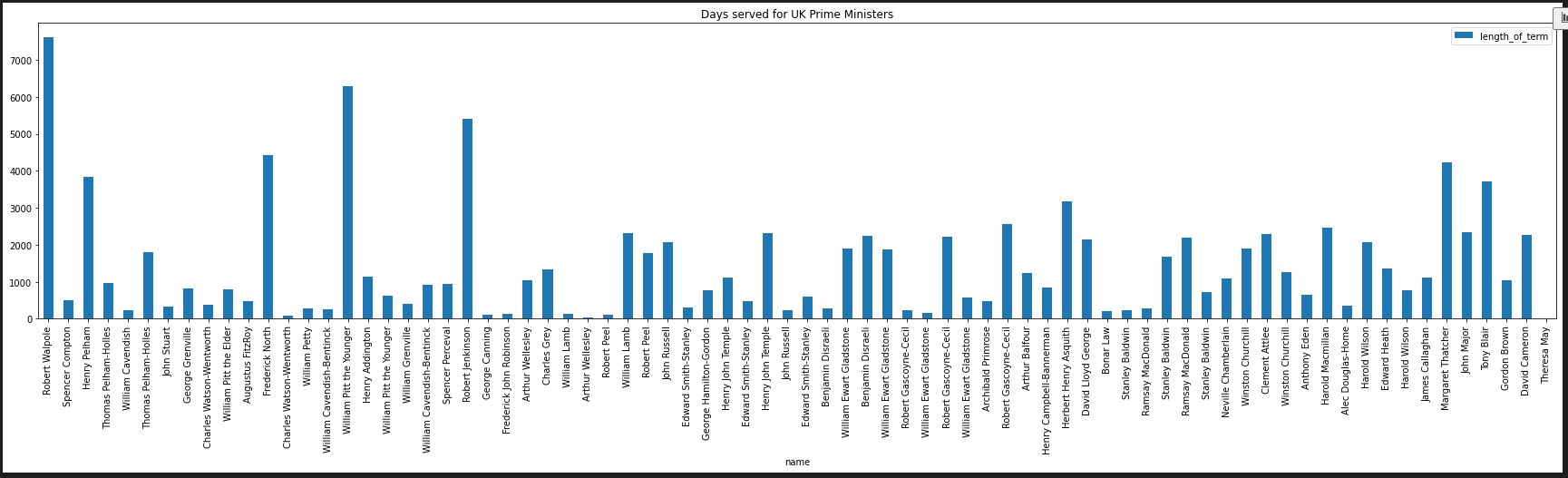












* A screenshot of the code and the output from Q3.8 challenge question



