Ocean Graham

Abstract

This portfolio presents problem solving techniques, methods and theories relating to programming. It displays and explains 2 computational tasks that have undergone processes explained within the portfolio.

403IT – Problem solving and programming

CW2 Portfolio



Table of Contents

[Table of Figures 2](#_Toc190942524)

[Introduction 3](#_Toc190942525)

[Problem-solving techniques 3](#_Toc190942526)

[Decomposition 3](#_Toc190942527)

[Abstraction 3](#_Toc190942528)

[The software development lifecycle 4](#_Toc190942529)

[Waterfall and agile methodologies 4](#_Toc190942530)

[Algorithm Design and generalised Problem solutions 6](#_Toc190942531)

[What is psuedocode? 6](#_Toc190942532)

[Pseudocode For palindrome checker 7](#_Toc190942533)

[pseudocode for Tip calculator 7](#_Toc190942534)

[flowchart for palindrome checker 9](#_Toc190942535)

[Flowchart for tip calculator 9](#_Toc190942536)

[programming language concepts 11](#_Toc190942537)

[Palindrome Checker 12](#_Toc190942538)

[Tip Calculator 14](#_Toc190942539)

[Testing 20](#_Toc190942540)

[palindrome Checker 20](#_Toc190942541)

[Tip Calculator 21](#_Toc190942542)

[conclusion 23](#_Toc190942543)

[Appendix 1 24](#_Toc190942544)

[Appendix 2 - Palindrome Checker Timeline 28](#_Toc190942545)

[appendix 3 – Tip calculator timeline 31](#_Toc190942546)

[References 36](#_Toc190942547)

# Table of Figures

[Figure 1 - Image of Waterfall Model 5](#_Toc190942548)

[Figure 2 - Steps in the Agile Model 6](#_Toc190942549)

[Figure 3 - Pseudocode for Palindrome Checker 7](#_Toc190942550)

[Figure 4 - Pseudocode for Tip Calculator 8](#_Toc190942551)

[Figure 5 - Flow Chart for Palindrome Checker 9](#_Toc190942552)

[Figure 6 - Flow Chart for Tip Calculator 10](#_Toc190942553)

[Figure 7 - Palindrome Checker variable 12](#_Toc190942554)

[Figure 8 - Palindrome Checker lowercase 12](#_Toc190942555)

[Figure 9 - Palindrome Checker loop 13](#_Toc190942556)

[Figure 10 - Palindrome Checker reverser 13](#_Toc190942557)

[**Figure 11 - Tip Calculator dictionary** 14](#_Toc190942558)

[Figure 12 - Tip Calculator menu recursion 15](#_Toc190942559)

[Figure 13 - Tip Calculator menu output 15](#_Toc190942560)

[Figure 14 - Tip Calculator orderlist 16](#_Toc190942561)

[Figure 15 - Tip Calculator order loop 17](#_Toc190942562)

[Figure 16 - Tip Calculator tipcalculator 18](#_Toc190942563)

[Figure 17 - Tip Calculator selection and output 18](#_Toc190942564)

[Figure 18 - Tip Calculator end print 19](#_Toc190942565)

[Figure 19 - Tip Calculator final output 19](#_Toc190942566)

[Figure 20 - Tip Calculator repeat loop 19](#_Toc190942567)

[Figure 21 - Palindrome Checker Test Table 20](#_Toc190942568)

[Figure 22 - Tip Calculator Test Table 22](#_Toc190942569)

403it – Problem solving AND PROGRAMMING

# Introduction

This portfolio displays problem-solving tasks, iterating the prompted tasks, how it was developed, resolved and completed. The primary goal of this coursework is to present a high degree of problem-solving skills, analytical skills and understanding of techniques, concepts and methodologies relating to programming. “Programming is the process of drawing up the schedule of the sequence of individual operations required to carry out the calculation” (Hartree. 1950, p. 111). The programming tasks chosen were the ‘**Palindrome Checker**’ and ‘**Tip Calculator’** applications.

# Problem-solving techniques

The first stage in resolving a problem is to identify the problem(s). This is just the same in programming, “programming requires a hierarchy of skills like abstraction, generalization, transfer and critical thinking” (Gomes & Mendes, 2007, p. 18). The nature of programming is the adaptation, development and resolution through multiple versions of code. The two tasks displayed in this portfolio present this evidently, utilising the techniques explained onwards:

## Decomposition

Decomposition is the technique of breaking down a complex problem into a litany of smaller simpler problems which are easier to solve (GeeksForGeeks, 2022, ‘What is Decomposition Computational Thinking?’). These smaller problems are solved to help find the larger original programs solution. The process typically can be broken down into a series of structured steps as suggested by (StudySmart, N.D., ‘Decomposition Computer Science’): Identification, Break down, Analysing components, Resolution, Integration.

## Abstraction

Abstraction is another important technique for computational thinking and is fundamental key aspect of object-oriented programming (Rouse, Techopedia, 2020, ‘Abstraction’). Abstraction focuses on filtering through points, ordering them on importance, then recognising similarities and ignoring differences. This allows a programmer to focus on the “elimination of the irrelevant and amplification of the essential” (Martin, 2003, Chapter 20 Section 3). This is important, developing the general idea, capturing the key functionalities and hiding irrelevant factors.

# The software development lifecycle

The Software Development Lifecycle (SDLC) has 7 main phases according to (Jackson, IBM, 2024, ‘What is the SDLC?’), Each phase has a specific purpose to maximise design: Planning & Analysis, Design, Development (Implementation), Testing, Deployment, Maintenance. The SDLC proves a valuable structure to teams, allowing them to “manage the development process systematically with clear goals and deliverables at every stage.” (Clark, The Product Manager, 2024, ‘What is the SDLC’).

## Waterfall and agile methodologies

There are a range of different SDLC models, however there are two approaches with higher prominence in relation to the others, these two approaches are:

* A diagram of a process

  AI-generated content may be incorrect.Waterfall – The waterfall model is a software development methodology, which is generally believed to have first been introduced in Winston W. Royce’s 1970s paper. This model entails a linear, sequential, document-focused model, having a clear roadmap of each stage.

Figure 1 - Image of Waterfall Model

Source: (Wieczorkowski & Polak, 2012, p. 196.)

As figure 1 shows, the waterfall model is a step by step model, requiring the completion of the current phase before being able to continue to them to the next phase (GitHub, 2024, ‘What is the SDLC’).

* Agile – The agile method is a flexible iterative approach that emphasises collaboration, adaptability and incremental deliveries, rather than waterfall’s documentation driven model. Agile can break down large projects, facilitating faster project completion and business efficiency (Xero, N.D. ‘How to implement agile methodologies in your business’)

A diagram of a process

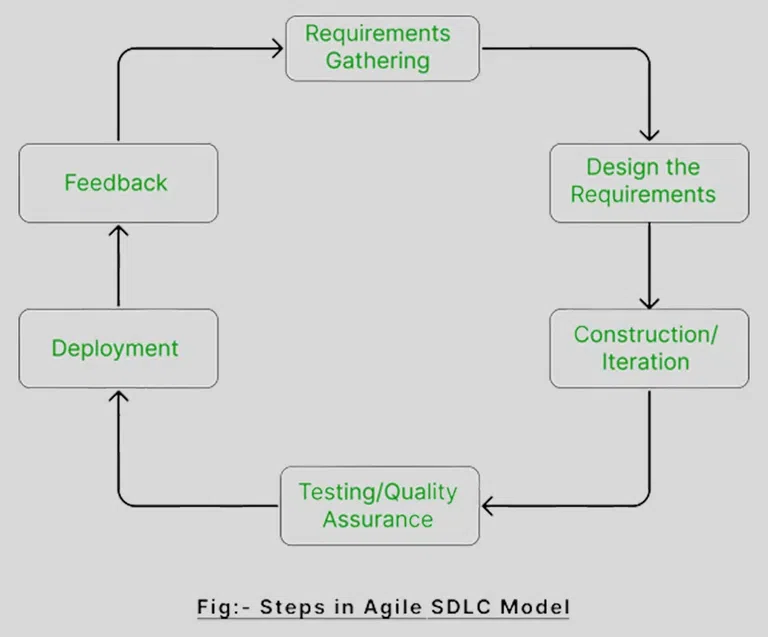
AI-generated content may be incorrect.

Figure 2 - Steps in the Agile Model

Source: Own Elaboration

The methodology used was taken into account as part of the Planning & Analysis stage. The Agile method was more utilised due to the nature of the project, having the flexibility (waterfall lacks) to alter the demands to adapt to new processes.

# Algorithm Design and generalised Problem solutions

Algorithms are a plan and set of steps for solving problems, they act as a template and foundation for programs. An algorithm is a composition of control structures; examples of such are defined by appendix 1.

When designing algorithms, preparation allows a more refined quality code in the long term, having clarified the problems and challenges during practised development. The tools such as Pseudocode and Flowcharts are essential during the design phase of the SDLC.

## What is psuedocode?

Pseudocode is “an informal contrived way of writing programs in which you represent the sequence of actions and instructions in a form humans can easily understand” said (Ubah, freeCodeCamp, 2021, ‘What is Pseudocode?’). Pseudocode helps us focus on the structure and logic to break the problem down, rather than focusing of details such as Syntax.

## Pseudocode For palindrome checker

A computer program with text

AI-generated content may be incorrect.

Figure 3 - Pseudocode for Palindrome Checker

Figure 3 shows the pseudocode for the Palindrome Checker, illustrating a template for an application that reverses words. Presenting the input a word, that word is then taken and reversed and compared to itself to see if it is a palindrome. This is designed to highlight the variability of inputs and outputs.

## pseudocode for Tip calculator

Below figure 4 shows the Pseudocode for the Tip Calculator. Presenting the framework for a program that presents an order menu to the user. This allowing them to order items and add it to a list. Next prompting a price from the accumulation of the ordered items. The opportunity to tip a selected amount finally.

A close-up of a document

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Figure 4 - Pseudocode for Tip Calculator

## A diagram of a company AI-generated content may be incorrect.flowchart for palindrome checker

Figure 5 - Flow Chart for Palindrome Checker

## Flowchart for tip calculator

A diagram of a flowchart

AI-generated content may be incorrect.

Figure 6 - Flow Chart for Tip Calculator

These maps show the flow of an algorithm using different shapes and symbols, Rectangles for processes, Diamonds for decision points, arrows indicate the direction of the path, parallelograms showing inputs and outputs.

The palindrome checker presents how the program may handle user inputs. The flowchart includes the decision point of an invalid input, this is due to the palindrome checker not allowing special characters, numbers or spaces.

The Tip calculator flow chart presents tip options, rather than a flat tip amount input like the pseudocode. This version uses percentage calculations of the ordered items list added on top of the bill if a selecting a tip option.

# programming language concepts

The language used to develop the Tip Calculator and Palindrome Checker was Python, this is due to its simplicity and ease of use, supporting multiple different types of code, including Procedural, Functional and Object-Oriented Programming (OOP)

Procedural programming involves following linear steps in an ordered path. Procedural was used in both to make the scripts follow loops that handle certain tasks. Functional programming is a when everything is tied to mathematical functions, this programming is a declarative style (GeeksForGeeks, 2024, Functional Programming Paradigm) this was not used for any of the algorithms designed. Finally, Object-Oriented Programming (OOP) is focused around objects. These objects represent real things such as in the case of the two programs, items such as an order list, menu, tip and a word to be inverted. OOP is used throughout both programs.

A vast number of practises and learning of Python coding tasks and functions have been used to develop the algorithms using Python. These include syntax elements, functions, variables, blocks, control structure and data structures. Some syntax listed is illustrated within appendix 1 and are some of the concepts used within the code.

## Palindrome Checker

A screen shot of a computer code

AI-generated content may be incorrect.The development stage of the SDLC for the palindrome checker begun with an input, this input being assigned to a variable to store the users input to be used for the process of the palindrome checker.

Figure 7 - Palindrome Checker variable

 The palindrome checker was briefed to not allow capitalisation. As a result, a ‘.lower()’ syntax was used, this returns a string to lowercase.

Figure 8 - Palindrome Checker lowercase

Next a ‘for loop’ was selected to go through the ‘word’ variable. A ‘for loop’ was utilised as the range to be repeated is known. The ‘for loop’ cycles through each character in the variable and checks if the character is within the alphabet, if not, then it defines the word as an invalid input, except if the character is a space where it removes the gap.

.replace() was also used in this algorithm, this was used to replace the space with a replacement of ‘’ which removes the gap.

A screen shot of a computer code

AI-generated content may be incorrect.All this would be done, unless the ‘word’ input was ‘q’ for quit. Which would ‘quit()’ the program, ‘quit()’ terminating the program completely.

Figure 9 - Palindrome Checker loop

A screen shot of a computer code

AI-generated content may be incorrect.The whole of the asking for an input was wrapped within a ‘while loop’ as the program would want to continue infinitely until a certain response was met (a valid input of a word without special characters, or ‘q’).

Figure 10 - Palindrome Checker reverser

After this, there is the process of reversing the word through ‘[::-1]’. This is then linked to a variable ‘reversed\_word’, this then gets compared to the original ‘word’ variable via a ‘==’ (both sides are equivalent) it prints that the word is a palindrome.  
Except if the word is defined as an invalid input where it breaks the process and starts the script from the beginning.

Finally, this is then wrapped within another while loop. Which can only be stopped by the ‘q’ input to ‘quit()’ as said before.

## Tip Calculator

The development stage for the tip calculator begun with firstly with a dictionary as they are well suited for cataloguing data, in this case a menu list. A screenshot of a menu

AI-generated content may be incorrect.

**Figure 11 - Tip Calculator dictionary**

This dictionary is reused through recursion through the program as it is a fundamental part of the designs structure.

A screenshot of a computer code

AI-generated content may be incorrect.

Figure 12 - Tip Calculator menu recursion

A screenshot of a menu

AI-generated content may be incorrect.A ‘for loop’ again is used due to desire to iterate through a sequence rather than meet a certain condition, the ‘for loop’ in the image above prints the menu into an easy to read format for the user.

Figure 13 - Tip Calculator menu output

The ordering system begins with empty list and empty/zero integer. These act as the store for the data of the ordering system.

A white background with red text

AI-generated content may be incorrect.

Figure 14 - Tip Calculator orderlist

Next the system requires a beginning input which is linked to the variable ‘ordereditem’. This variable then gets put into ‘.lower()’ to remove capitalisation and characters into lowercase. This variable is then compared to items in the list via an ‘if-else’ statement. If the ‘ordereditem’ is in the menu, it gets added to the empty list, via the ‘.append()’ function. The price of the item gets added to the empty/zero integer. However, if the input is not on the list, the invalid input does not get appended to the ‘addedorder’ list and the loop gets repeated.

A screenshot of a computer code

AI-generated content may be incorrect.Figure 15 - Tip Calculator order loop

This script is encapsulated within a ‘while loop’ allowing the user to infinitely add items to the user’s order, only ending the loop when the user enters ‘f’ for finish, breaking the loop and continues the script to its next section.

A screenshot of a computer program

AI-generated content may be incorrect.The second section of the program acts as the billing and tipping section, firstly it starts with the tip calculator function via a ‘def’ function to be recalled.

Figure 16 - Tip Calculator tipcalculator

A screenshot of a computer code

AI-generated content may be incorrect.Next the user is prompted with the items they have ordered and how much they must pay, it then asks if they would like to add a tip option. Depending on the selected valid option, a specific function within the defined ‘tipcalculator’ will be activated.

A close-up of some words

AI-generated content may be incorrect.

Figure 17 - Tip Calculator selection and output

A close up of text

AI-generated content may be incorrect.Figure 18 - Tip Calculator end print

Each function within the ‘tipcalculator’ works similarly, functioning via multiplying ‘\*’ the ‘ordertotal’ by the percentage selected. The custom amount has more complexity due to a float error occurring if attempted to be calculated with the same format as the other options. The custom option prompts an input of what percentage you would like to tip; this is then assigned to the ‘customamount’ variable which is connected to the tipcalculator.



Figure 19 - Tip Calculator final output

A screenshot of a computer

AI-generated content may be incorrect.Finally, the script outputs the different prices paid, in a final print, next the order list and integer are reset, then asking if you would like to repeat, either exiting or repeating the program. The order list and integer are reset in the case of a repeat.

Figure 20 - Tip Calculator repeat loop

The function ‘round()’ has been used throughout the algorithm, this rounds a decimal to the selected option, in this case, meaning the ordertotal is outputted with a decimal place of 2, this is to make it function the same as currency.

## Testing

Testing is a crucial stage in the SDLC as it helps the programmer define errors and defects in the software to be resolved and fixed. It also helps ensure that functions are working correctly, and requirements are met.

The testing in this section is the final testing of the end product, however evidence for testing and resolutions through the projects design phase can be found within the appendix 2 and 3.

## A white grid with black text AI-generated content may be incorrect.palindrome Checker

Figure 21 - Palindrome Checker Test Table

## A white grid with black text AI-generated content may be incorrect.Tip Calculator

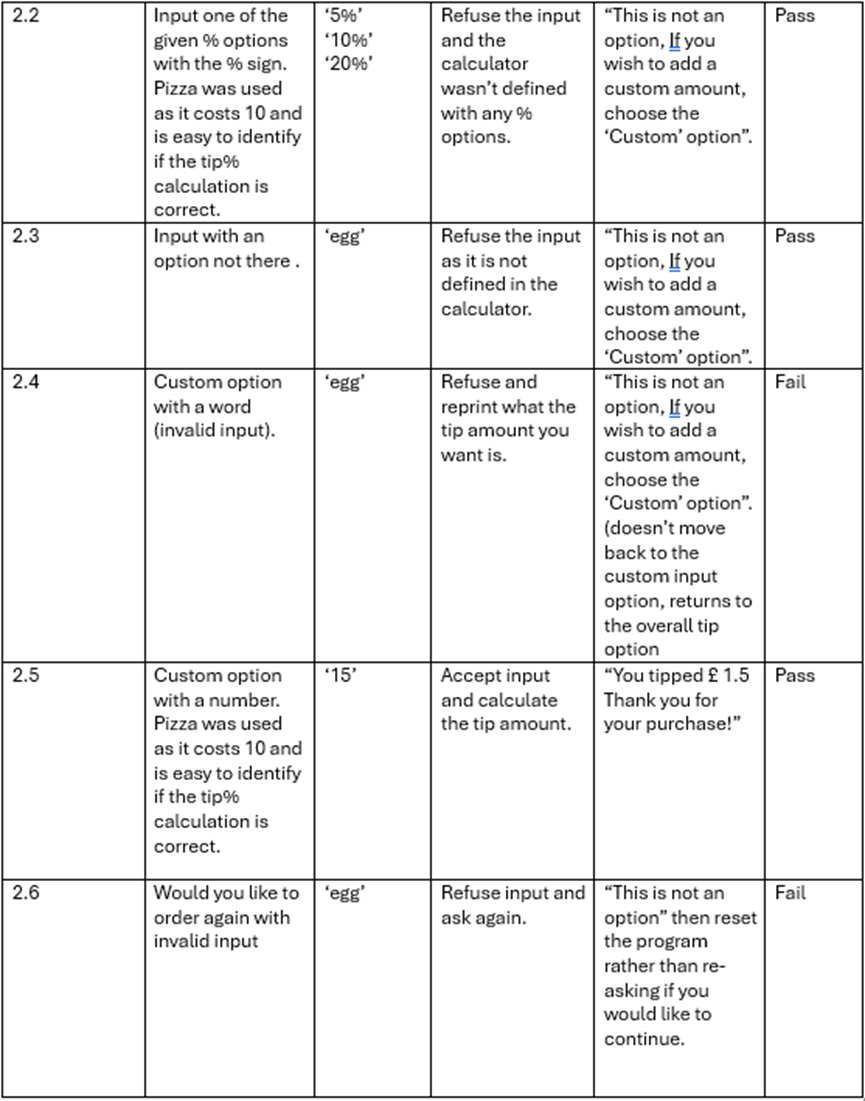


Figure 22 - Tip Calculator Test Table

# conclusion

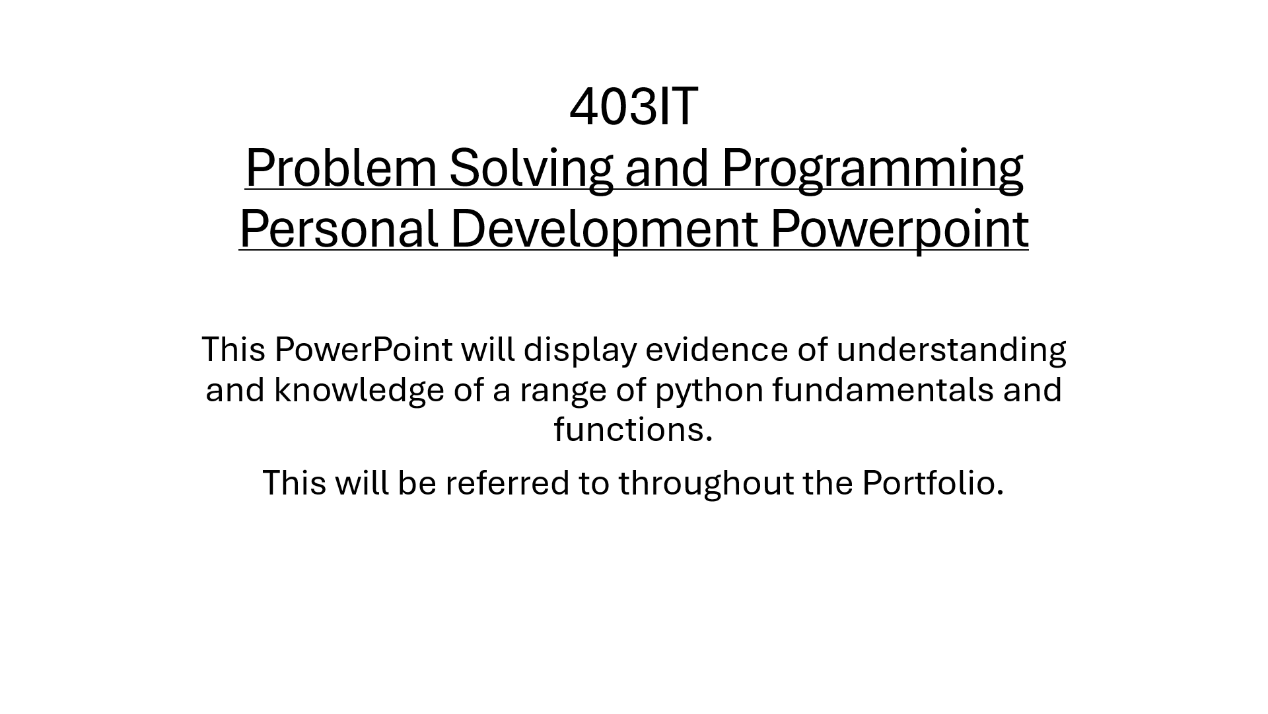
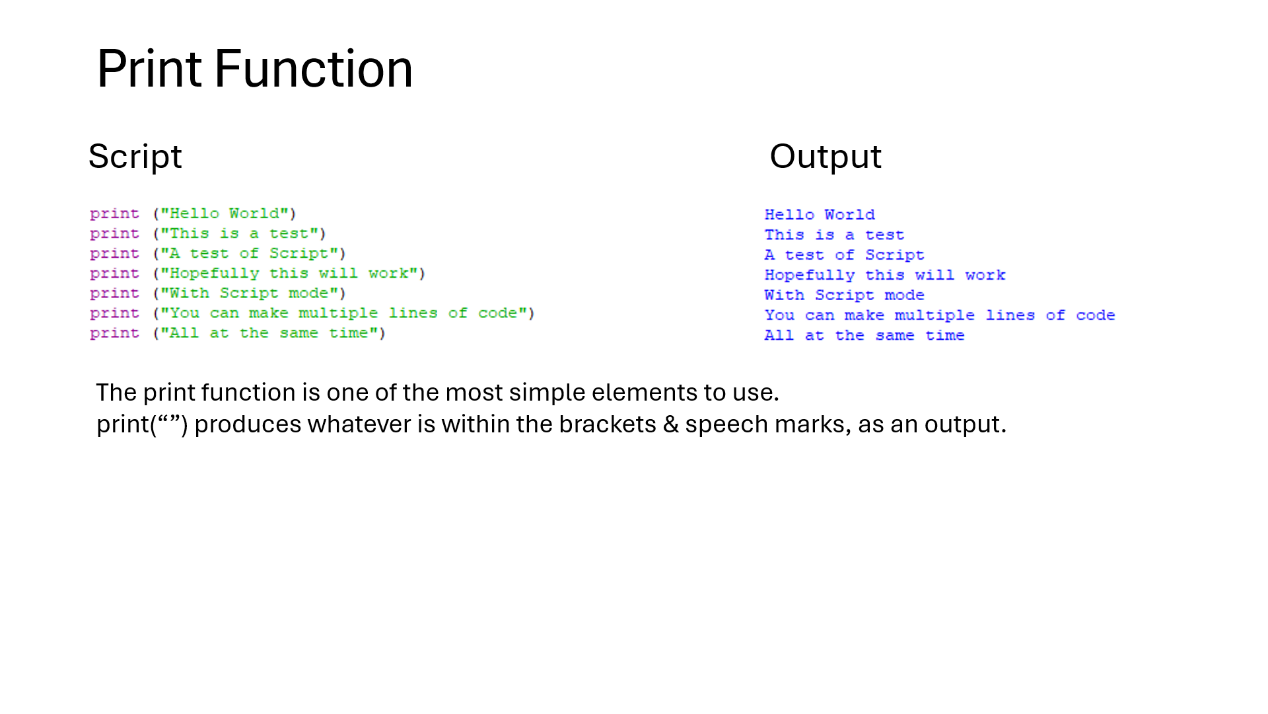
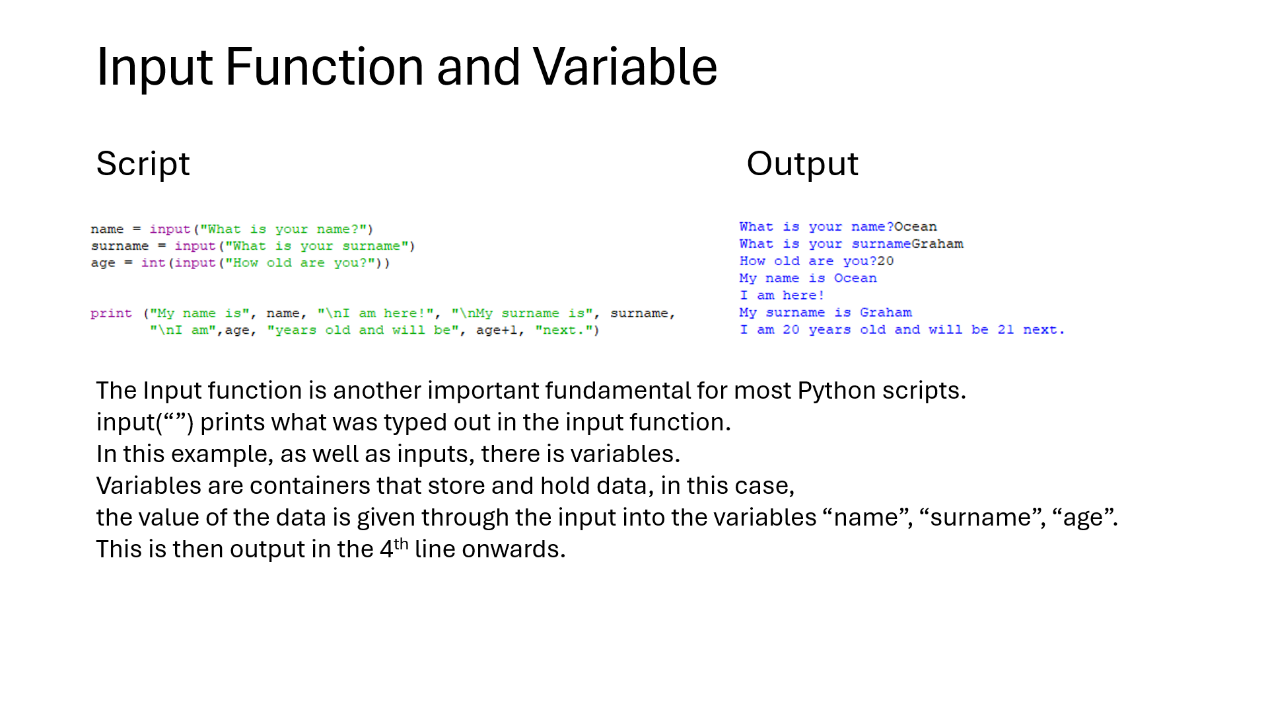
The conclusion of this project and portfolio resulted in two task programs being created, undergoing problem solving techniques listed within the Problem Solving Techniques section.

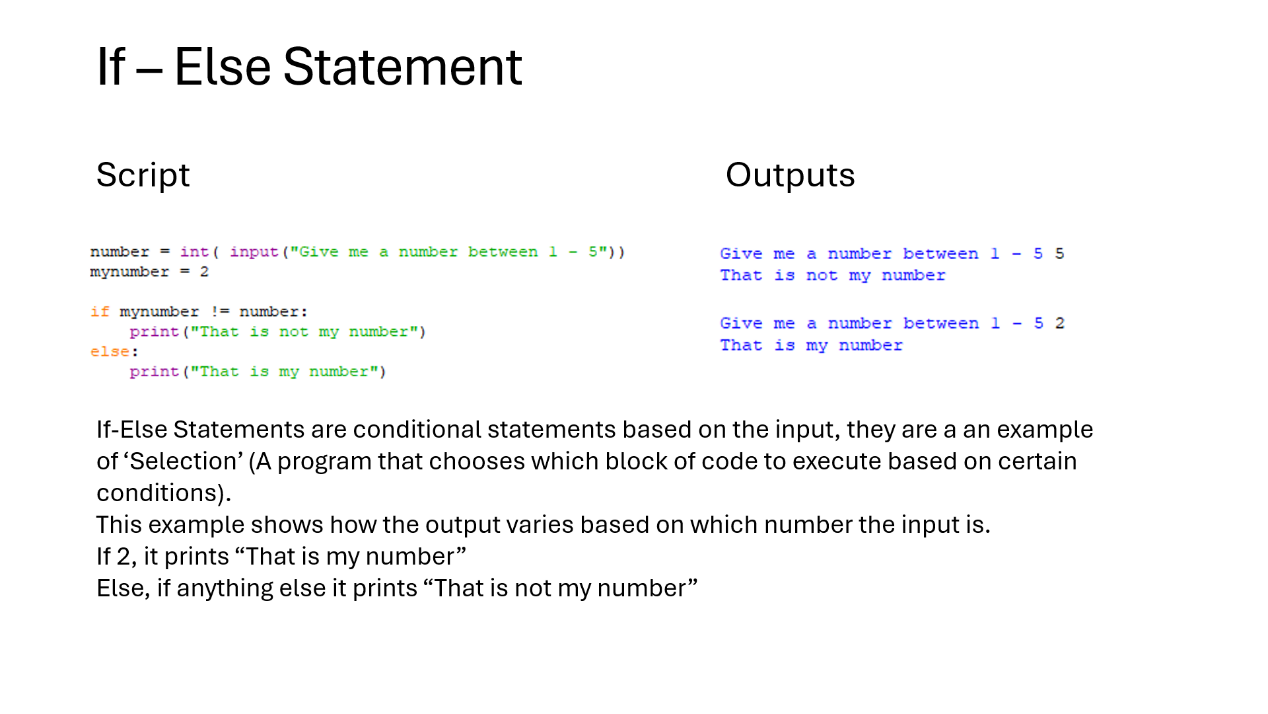
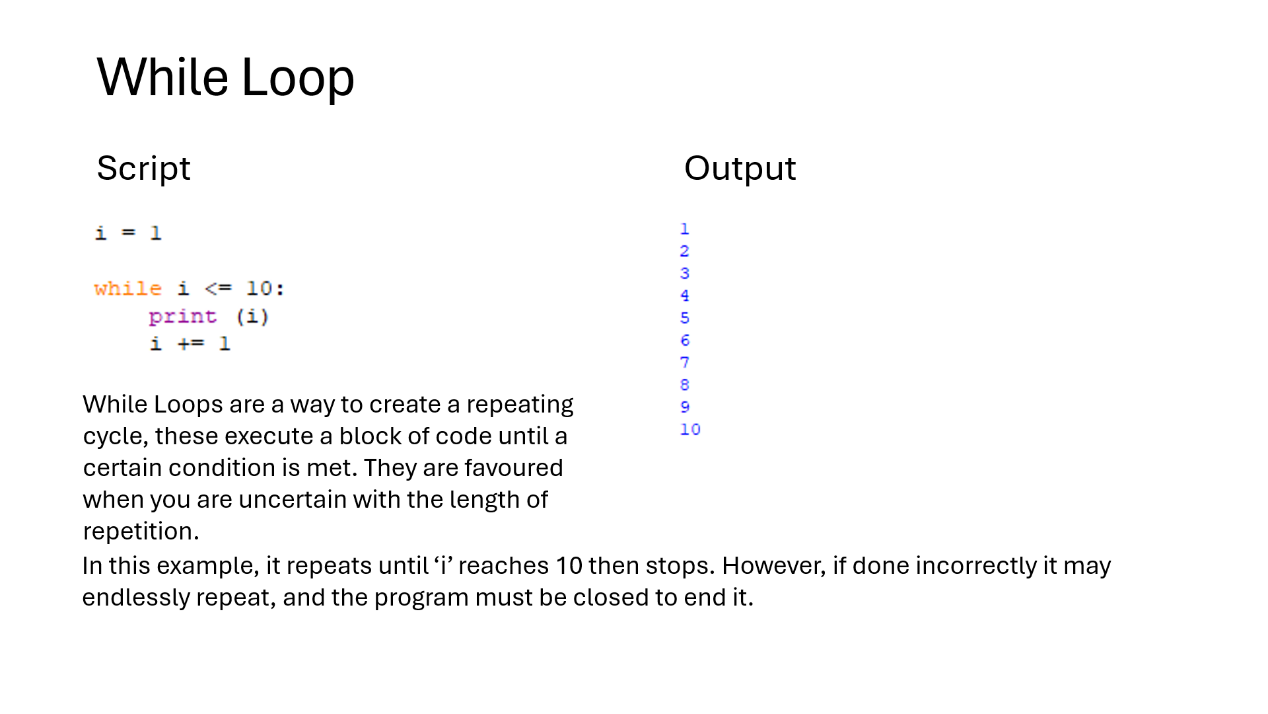
The use of the SDLC and Agile method gave valuable structure to the development, allowing a more efficient, streamlined development and overall better quality final product.

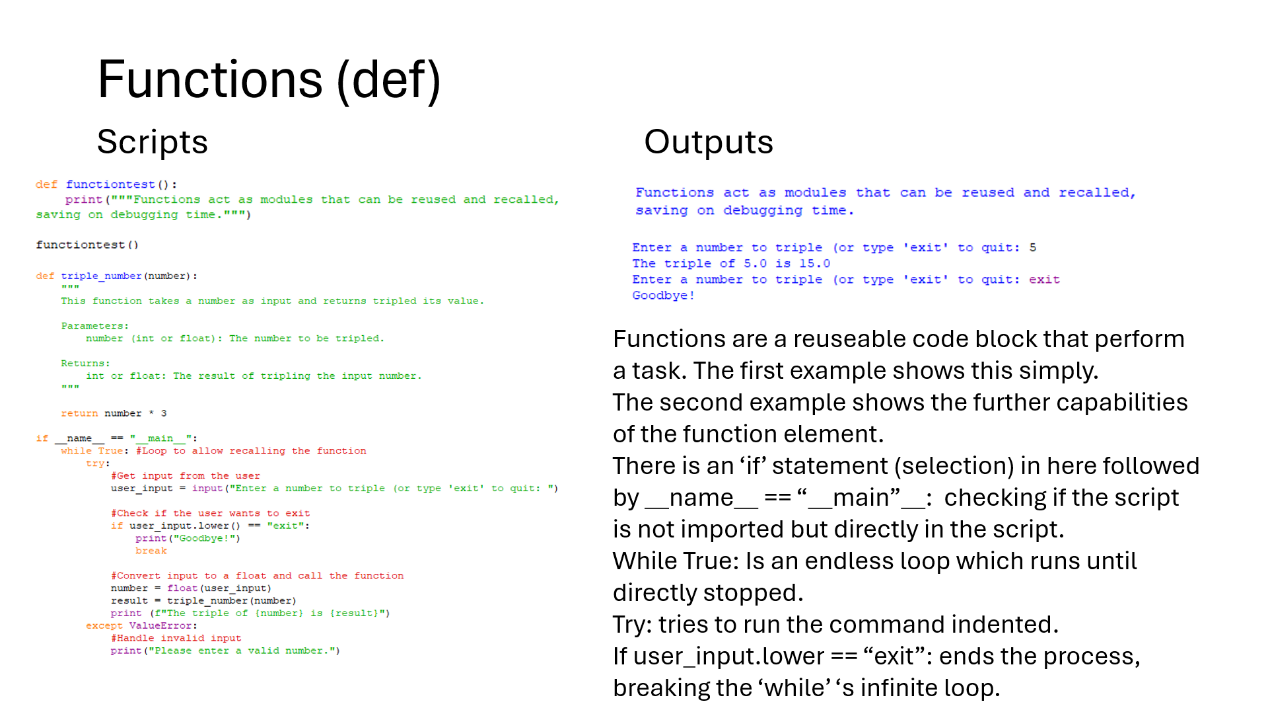
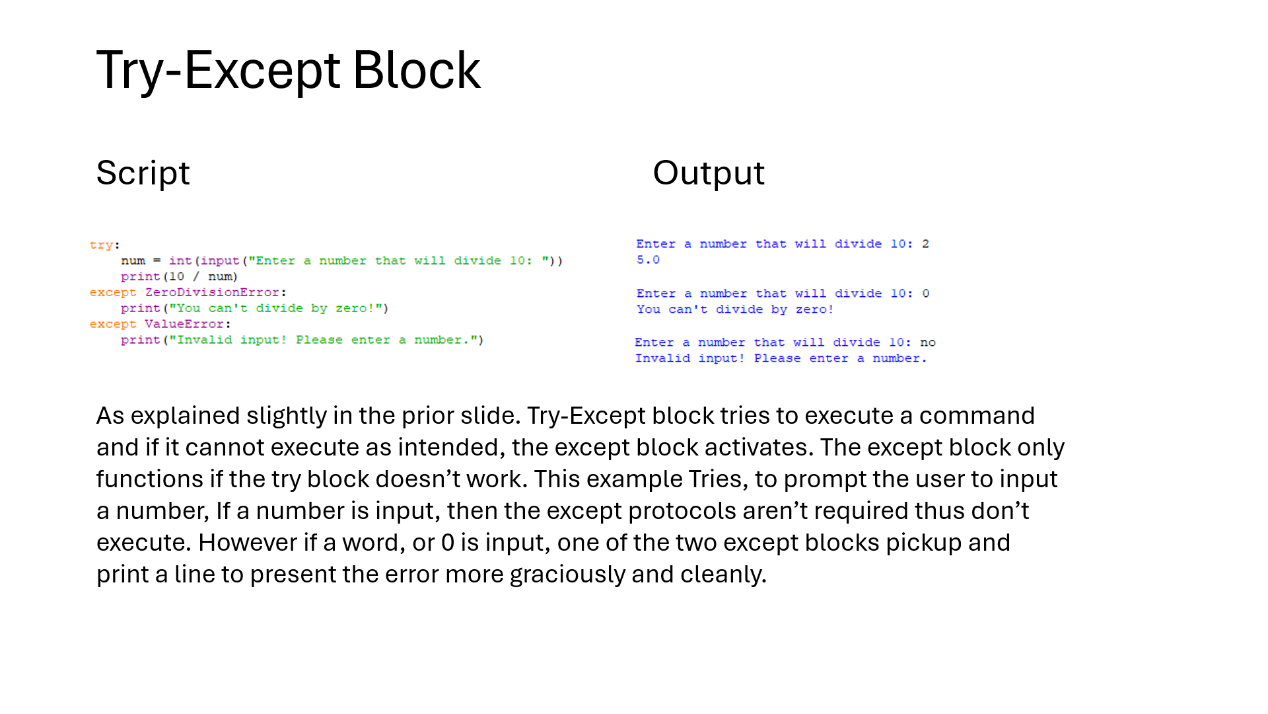
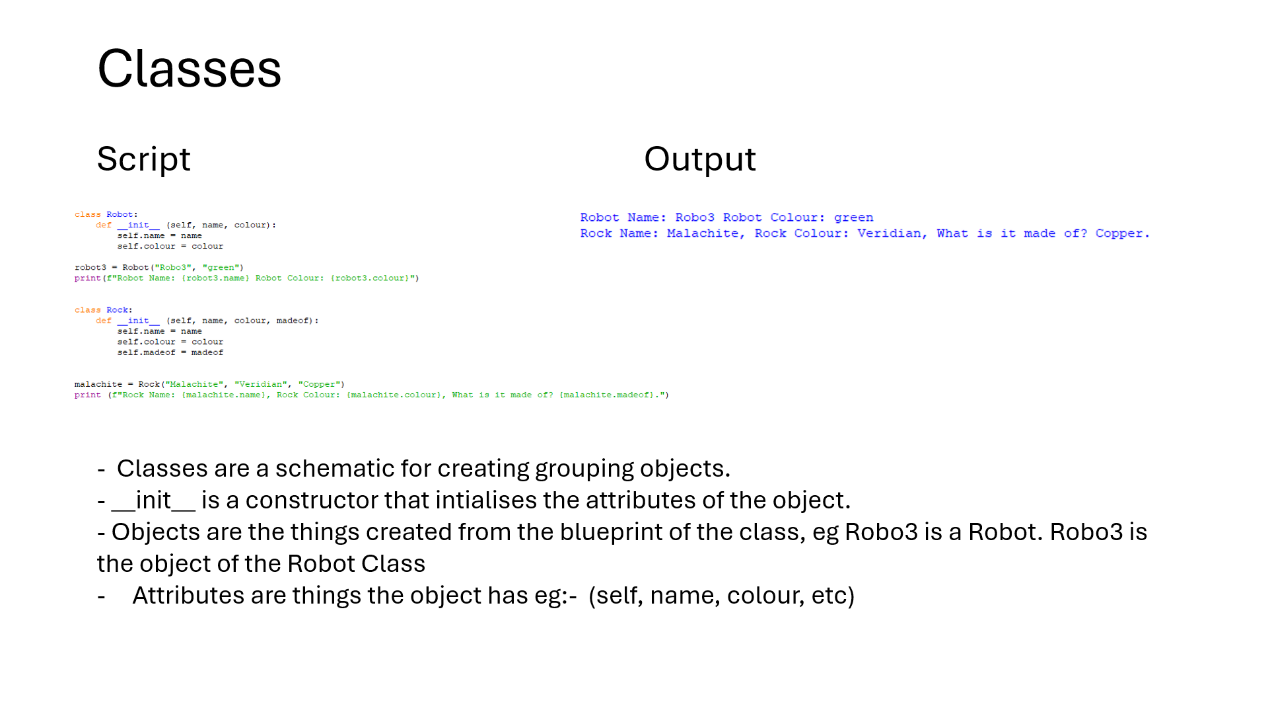
Both selected tasks succeeded in what was expected from the tasks, with some minor flaws residing within the Tip Calculator that would require further time to resolve.

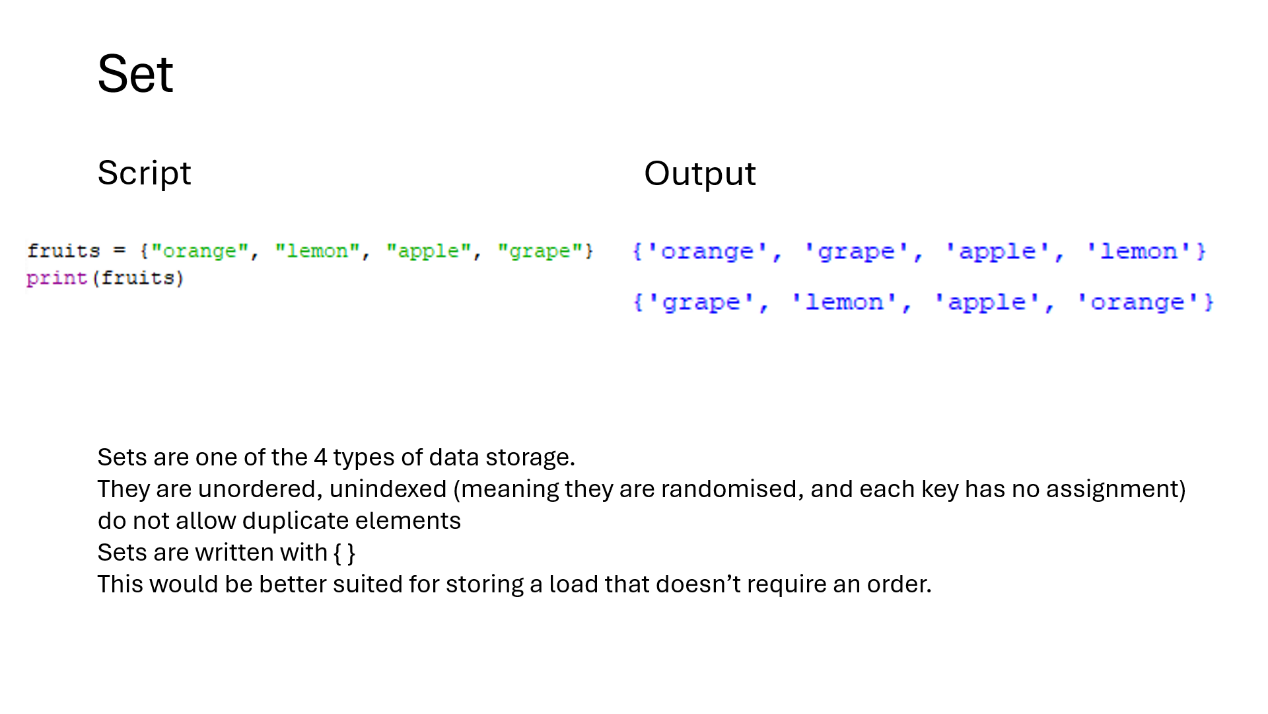
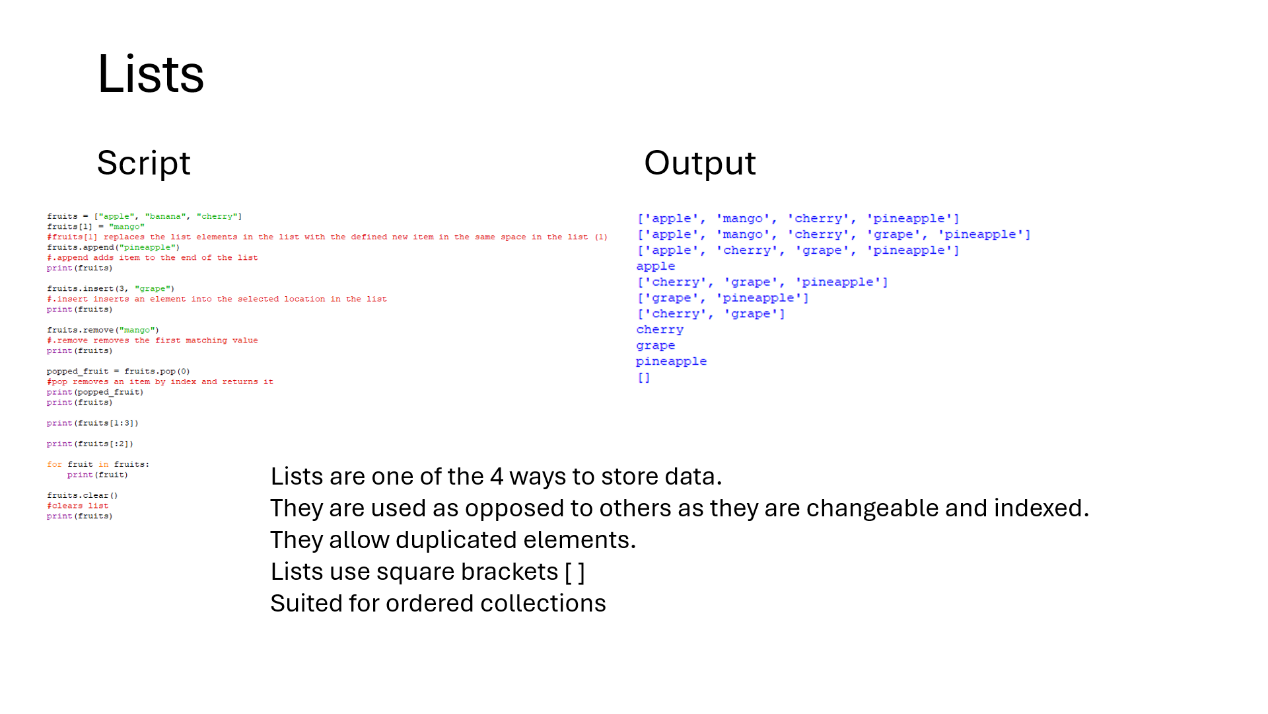
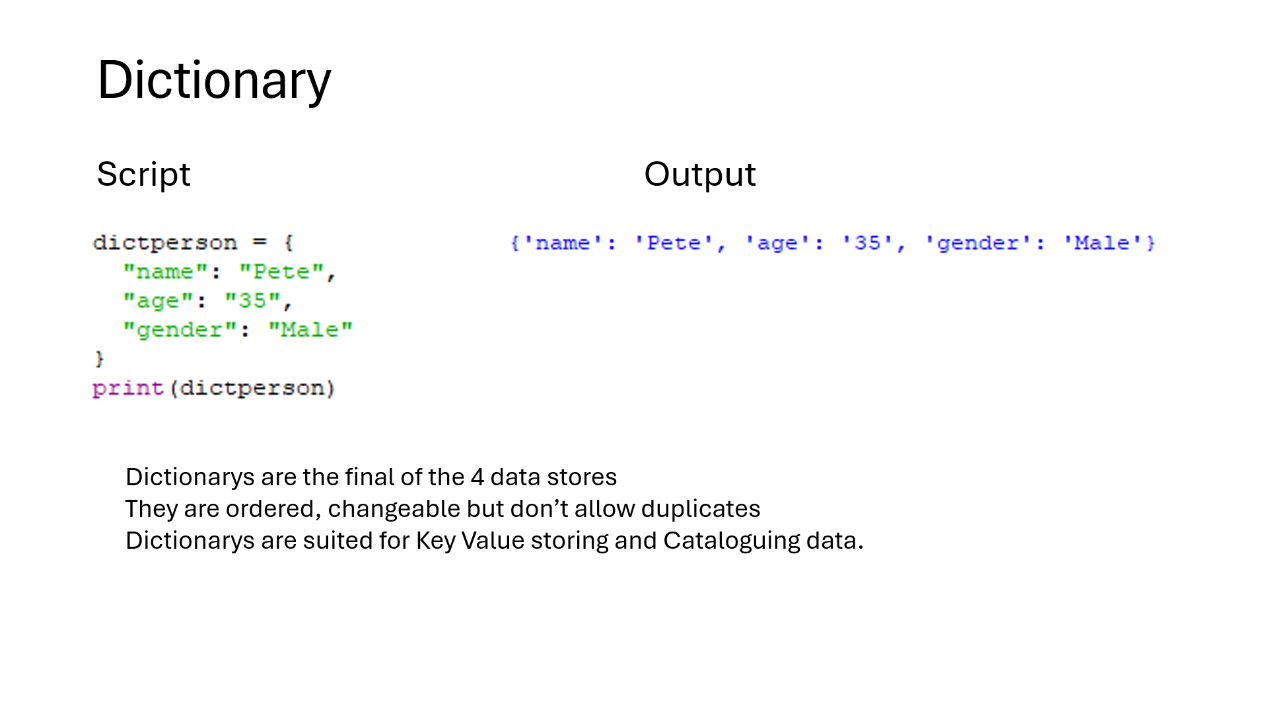
Overall, the project has high degrees of problem solving and programming skills, and the displayed the need to follow a structured approach to development.

# Appendix 1

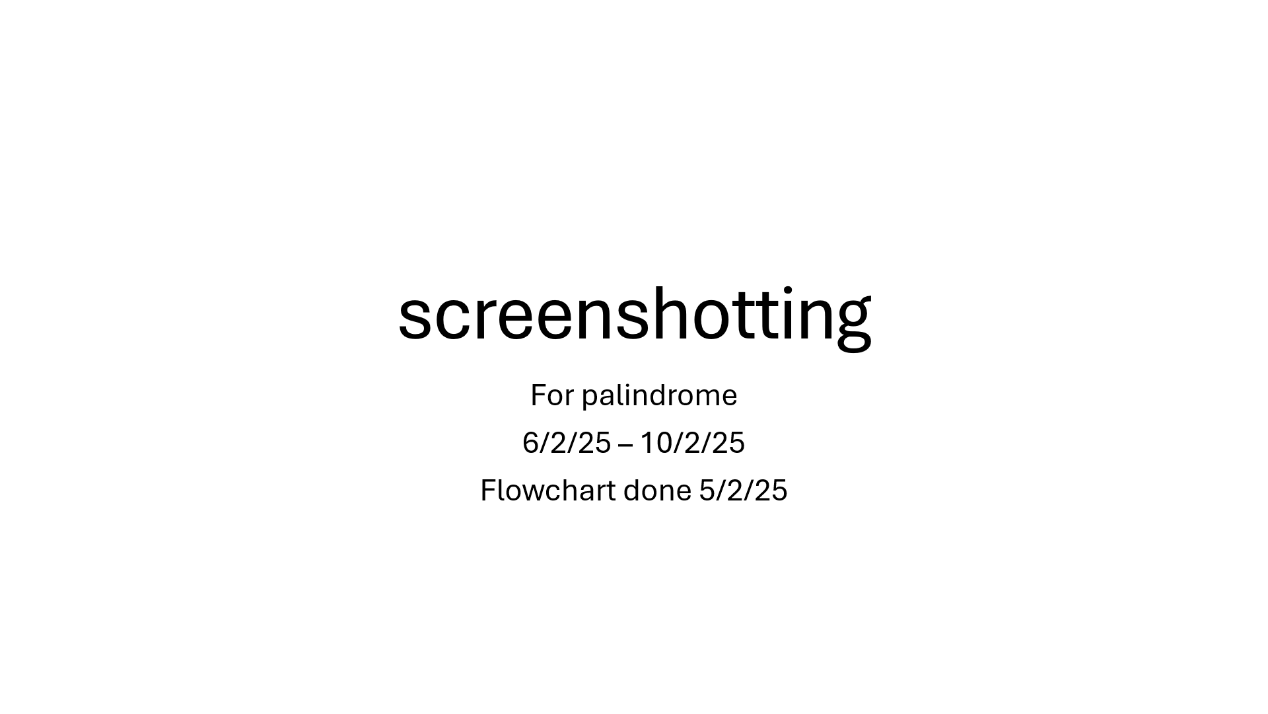


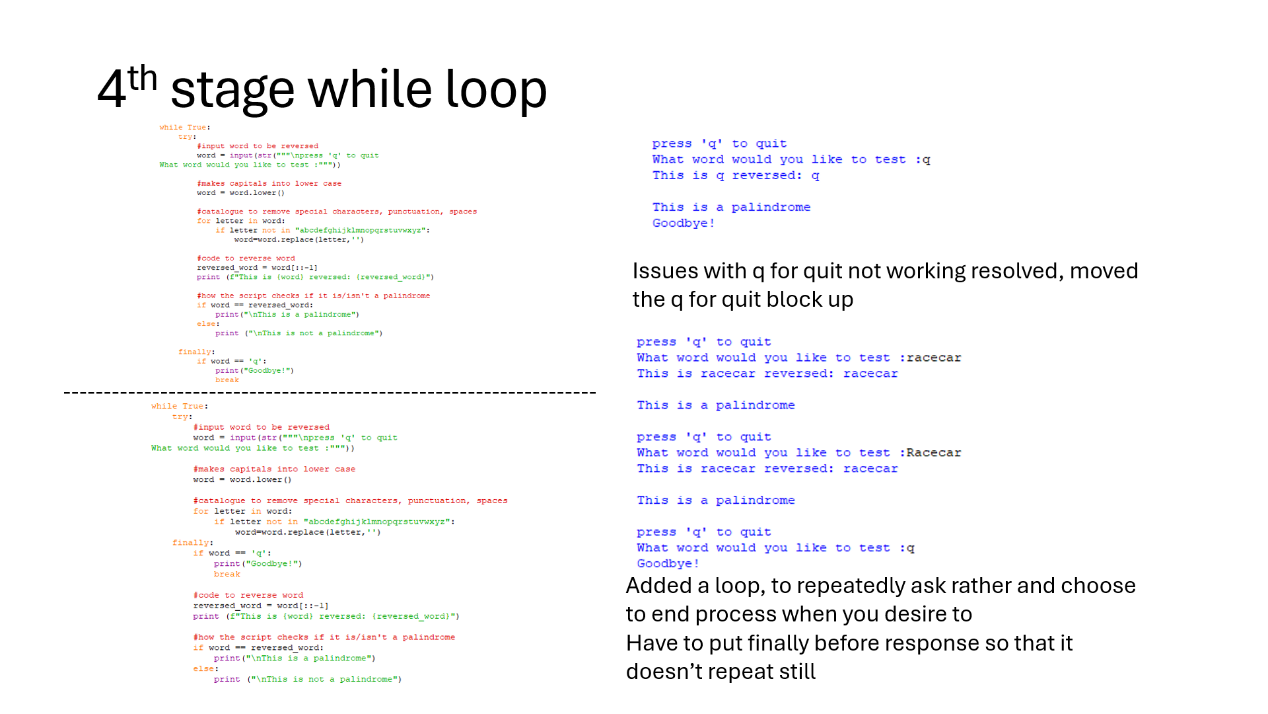
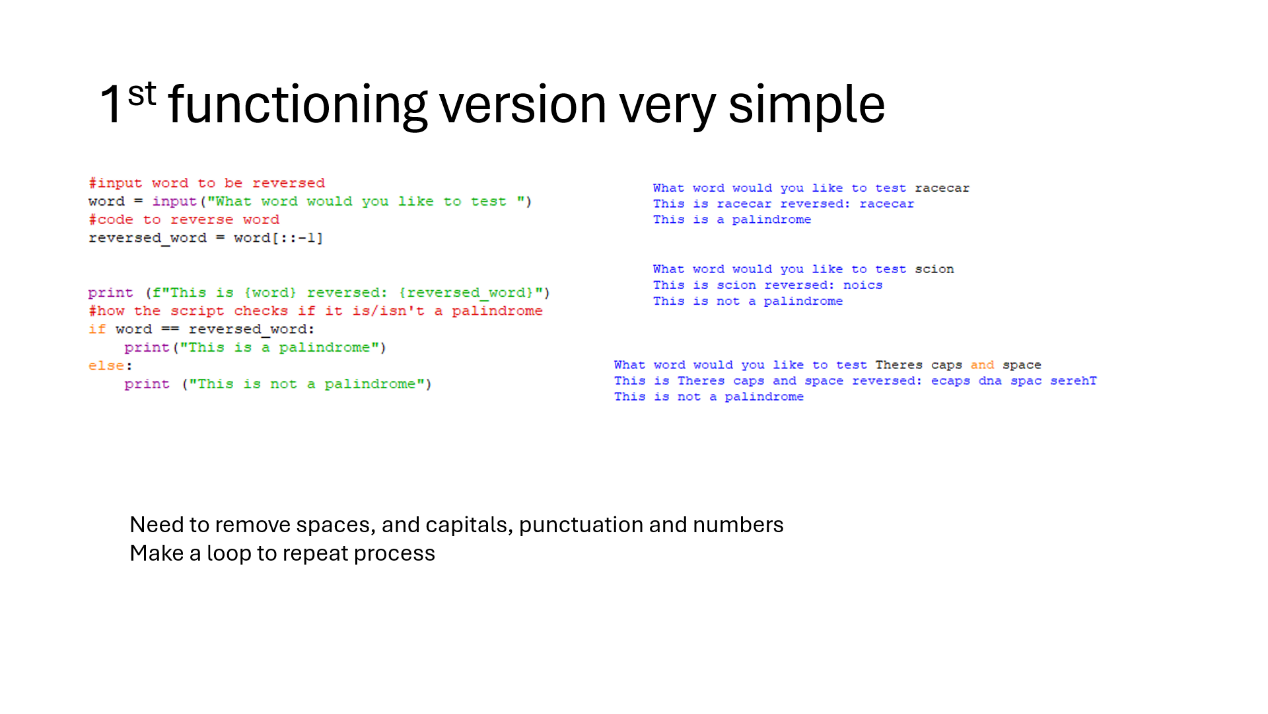
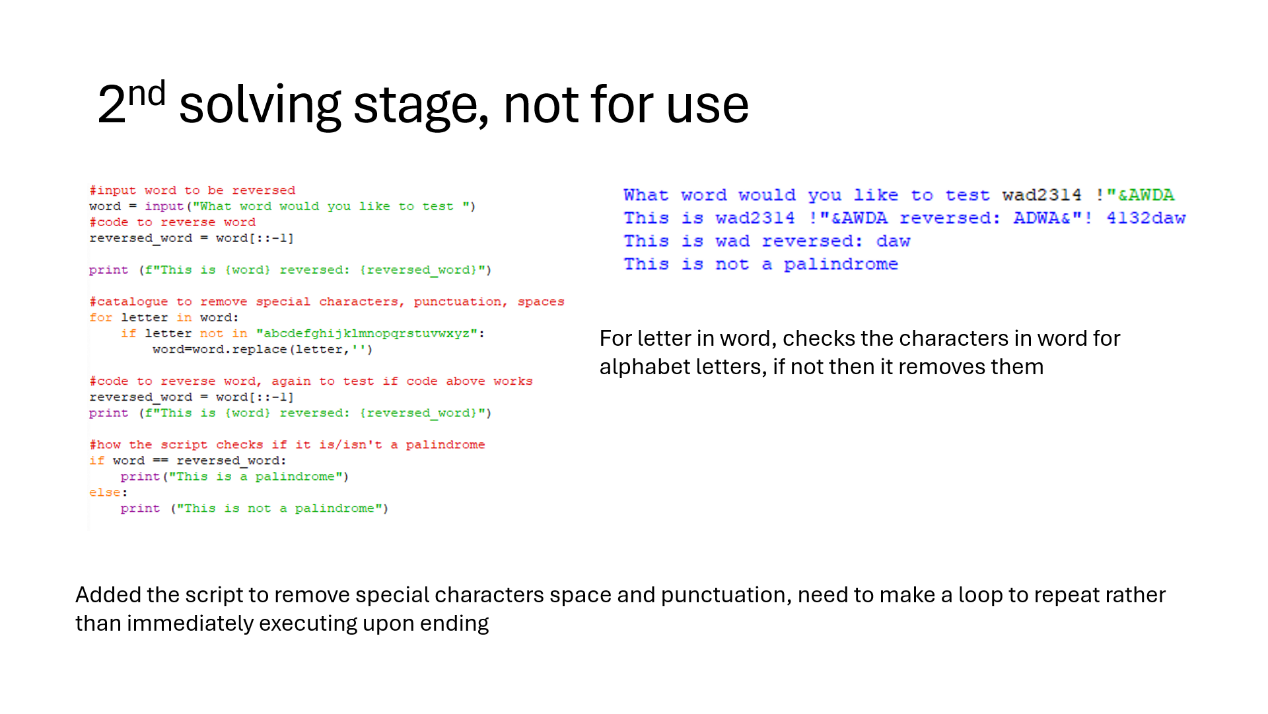
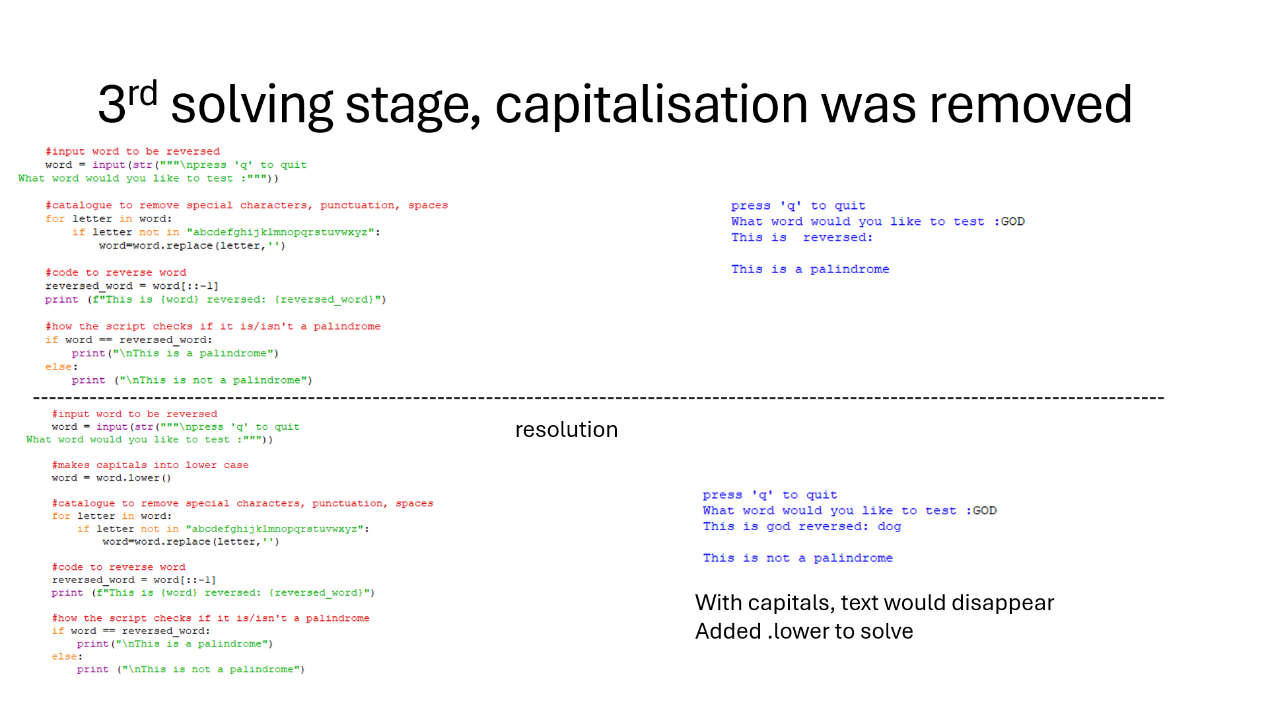


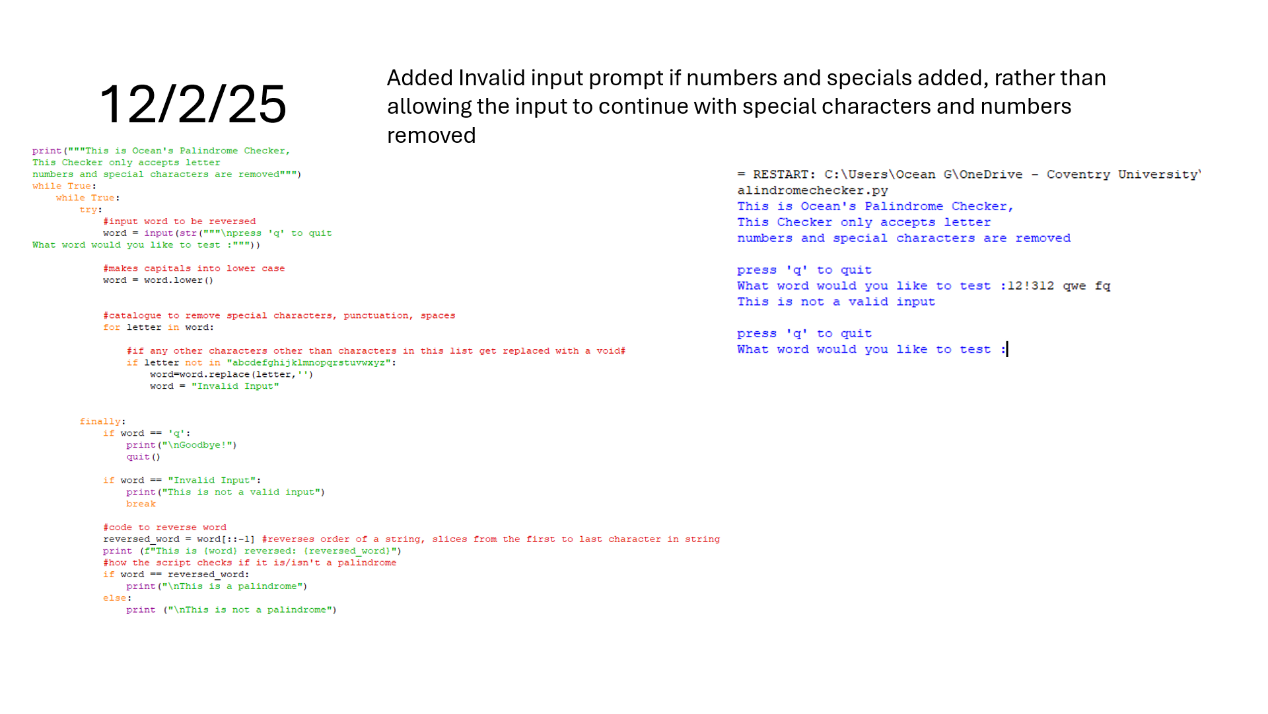


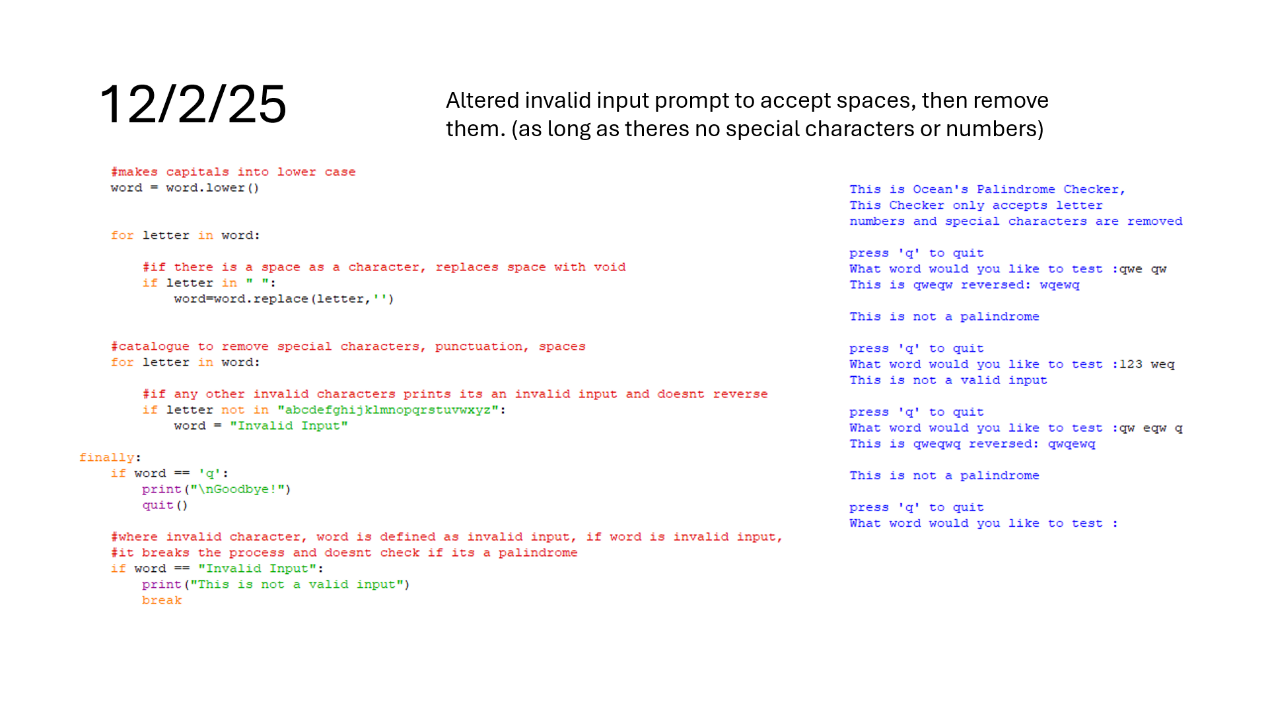


# PowerPoint Slide Show - Presentation in C:\Users\Ocean G\OneDrive - Coventry University\403IT_Programming_repos\403_Programming\assignments\15228802_403IT_CW2.docxAppendix 2 - Palindrome Checker Timeline

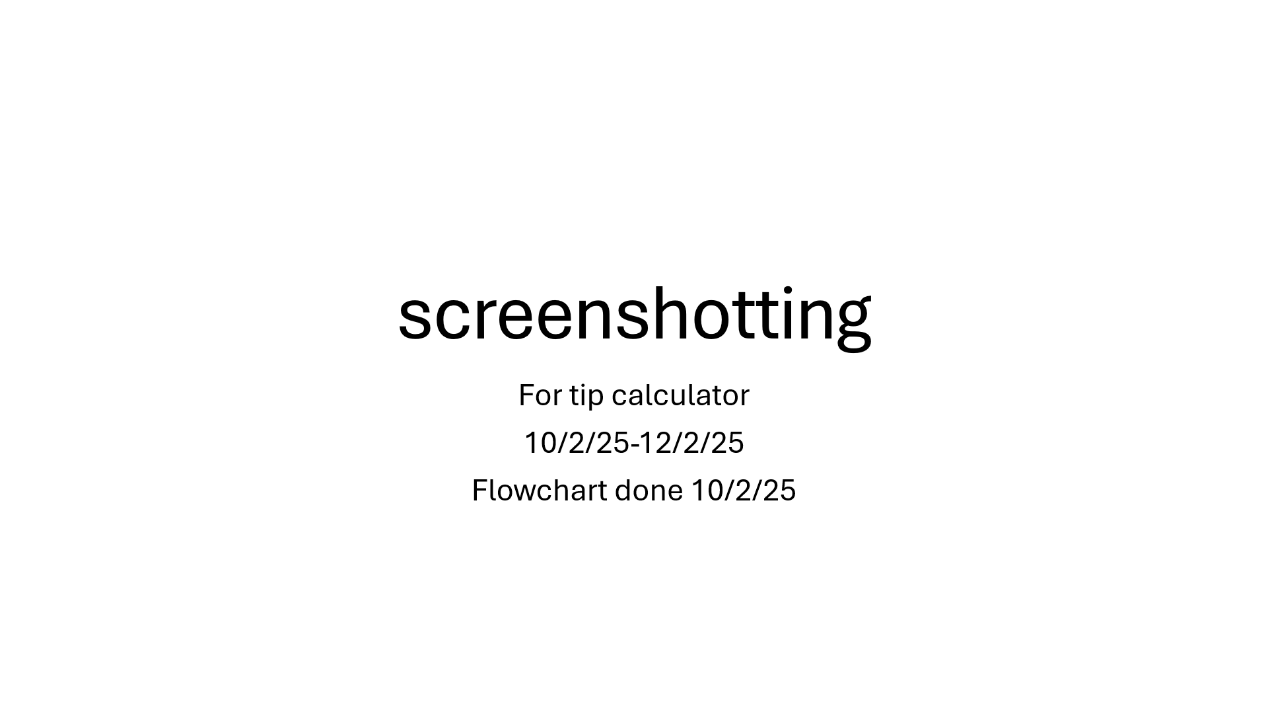
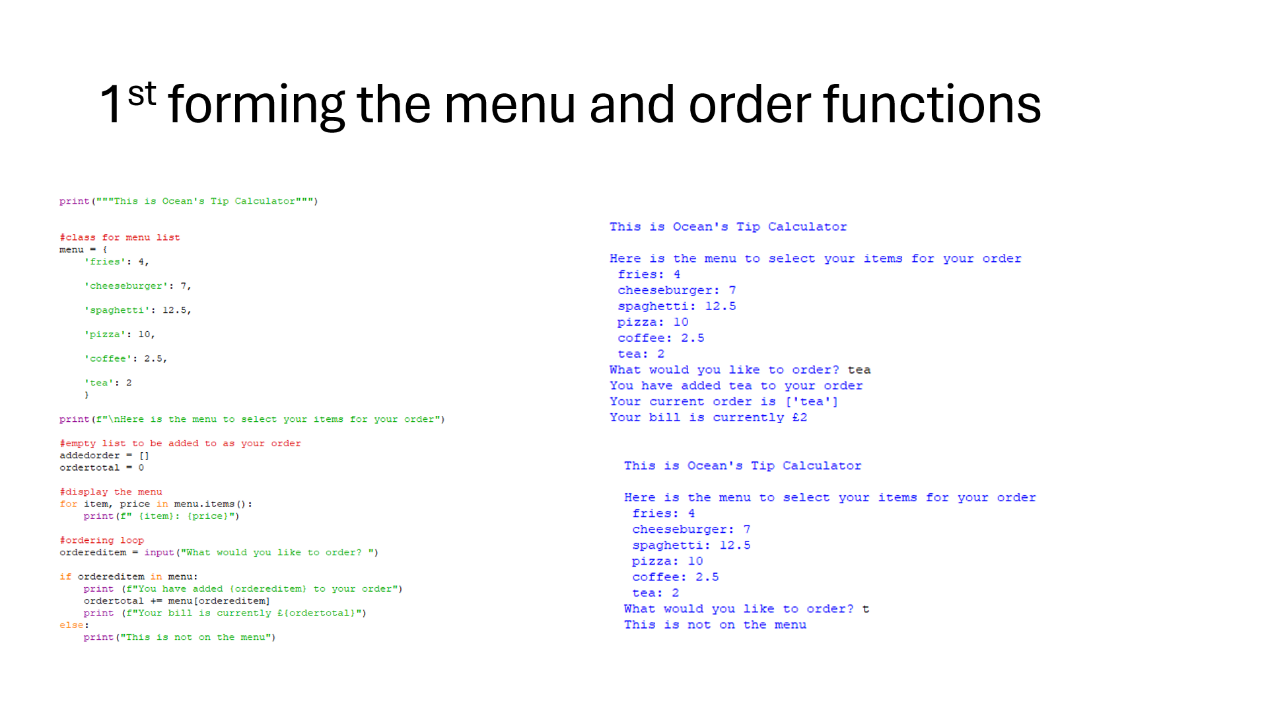


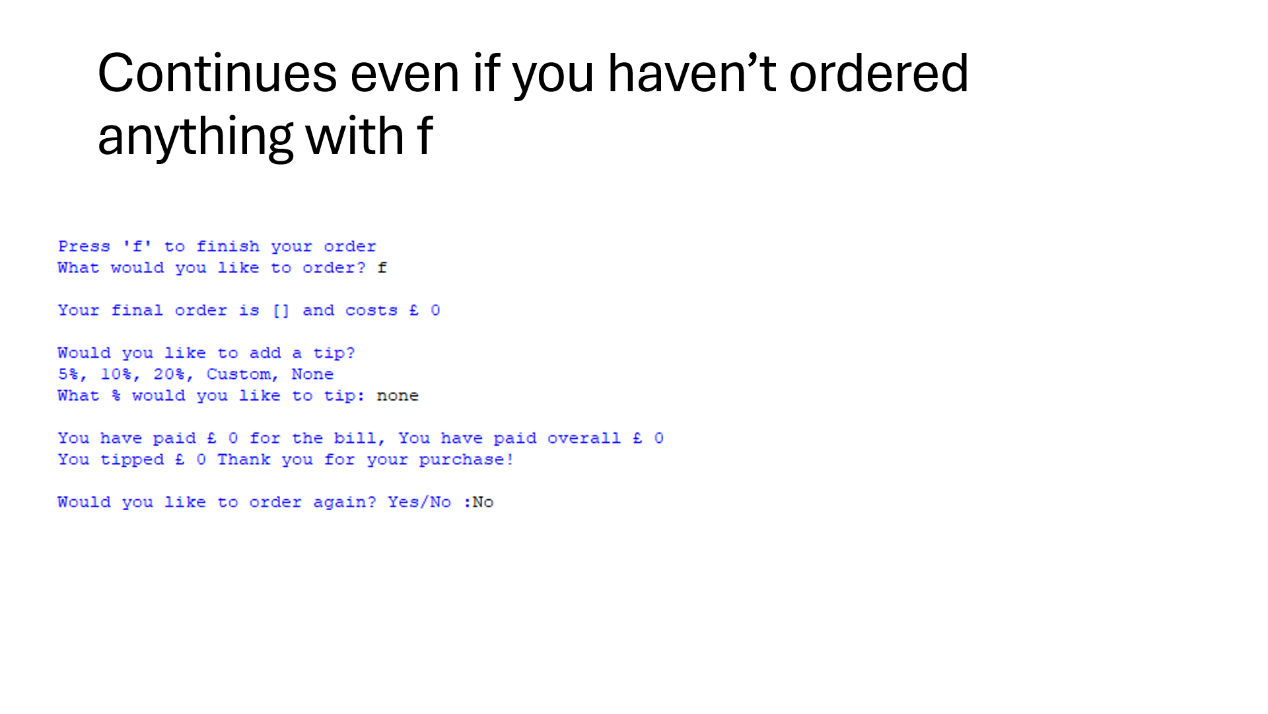
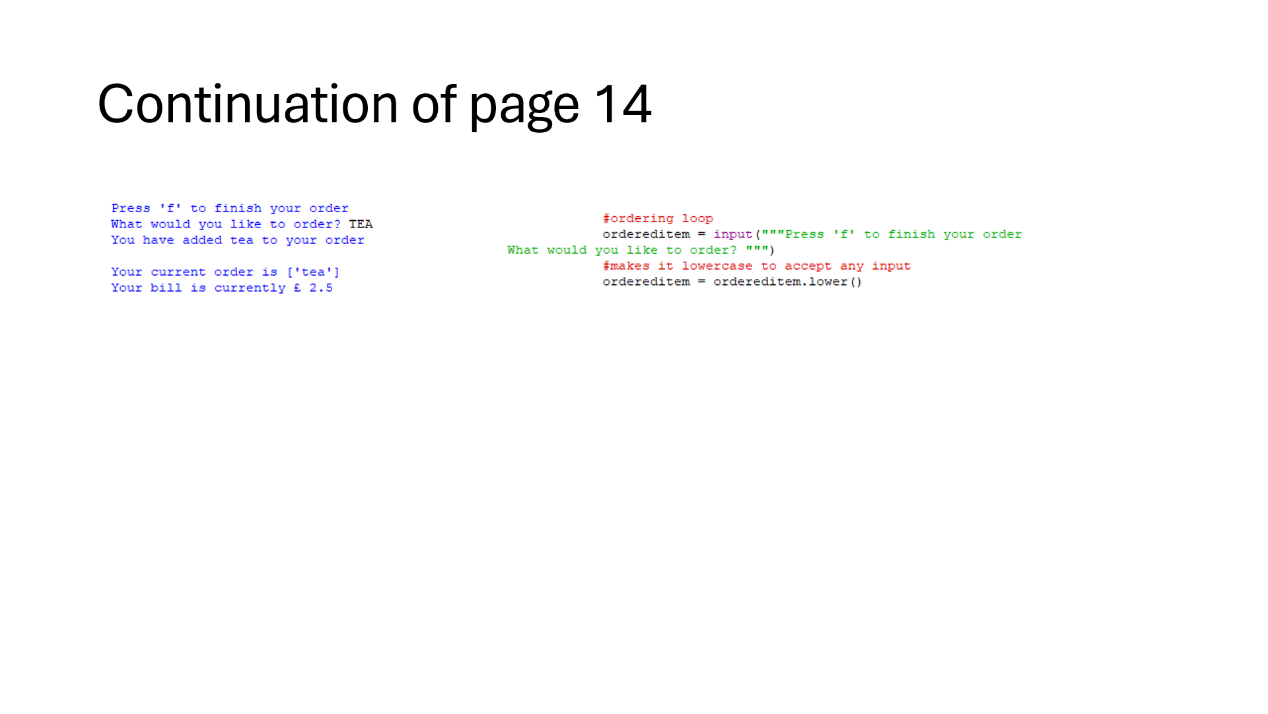
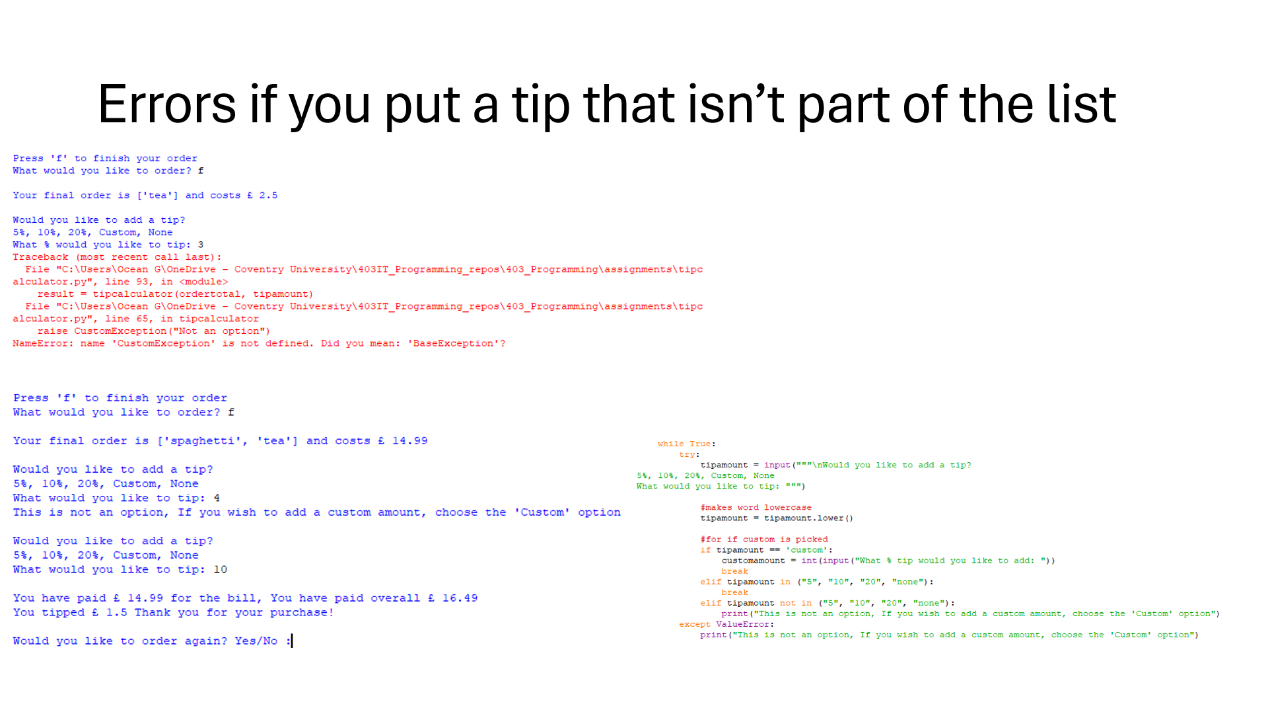
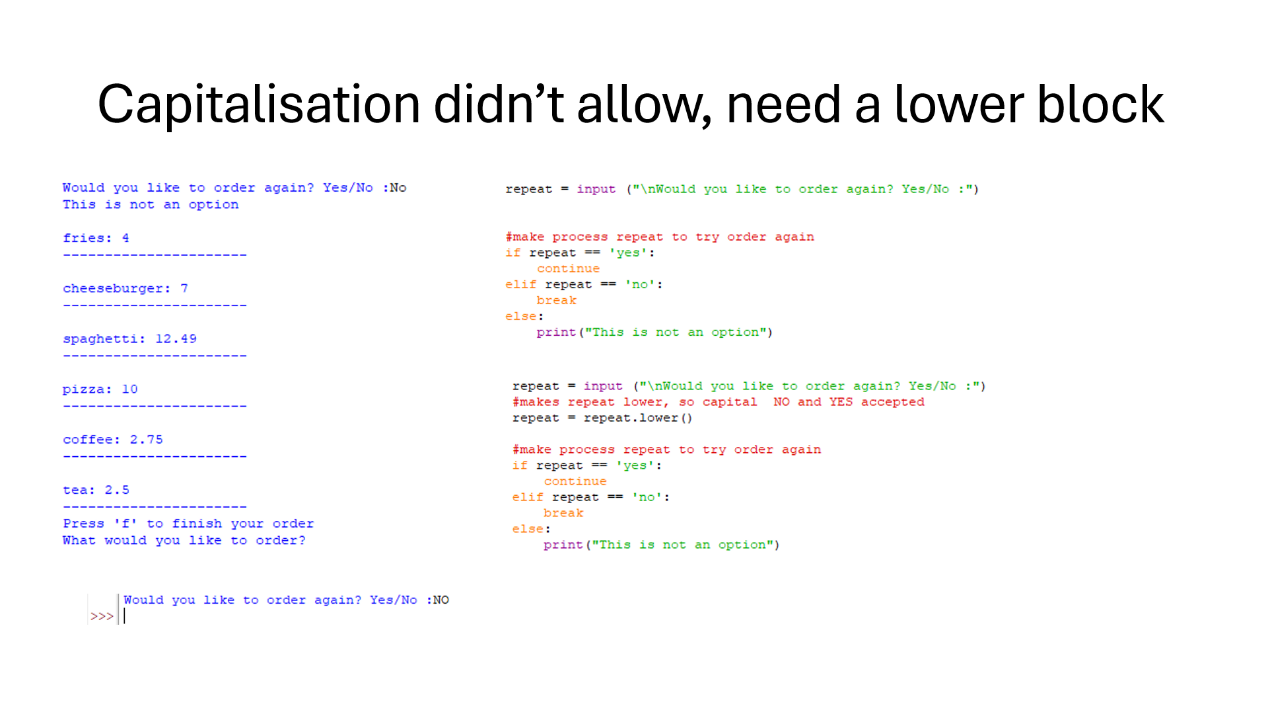
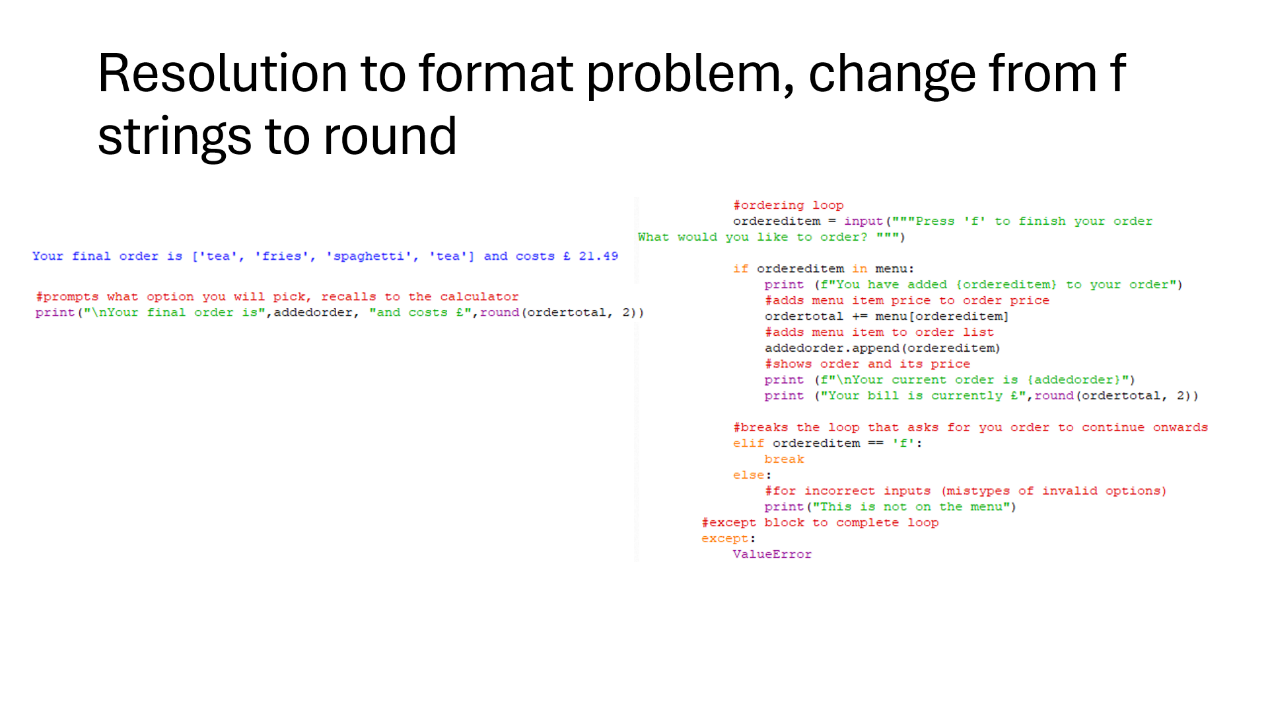
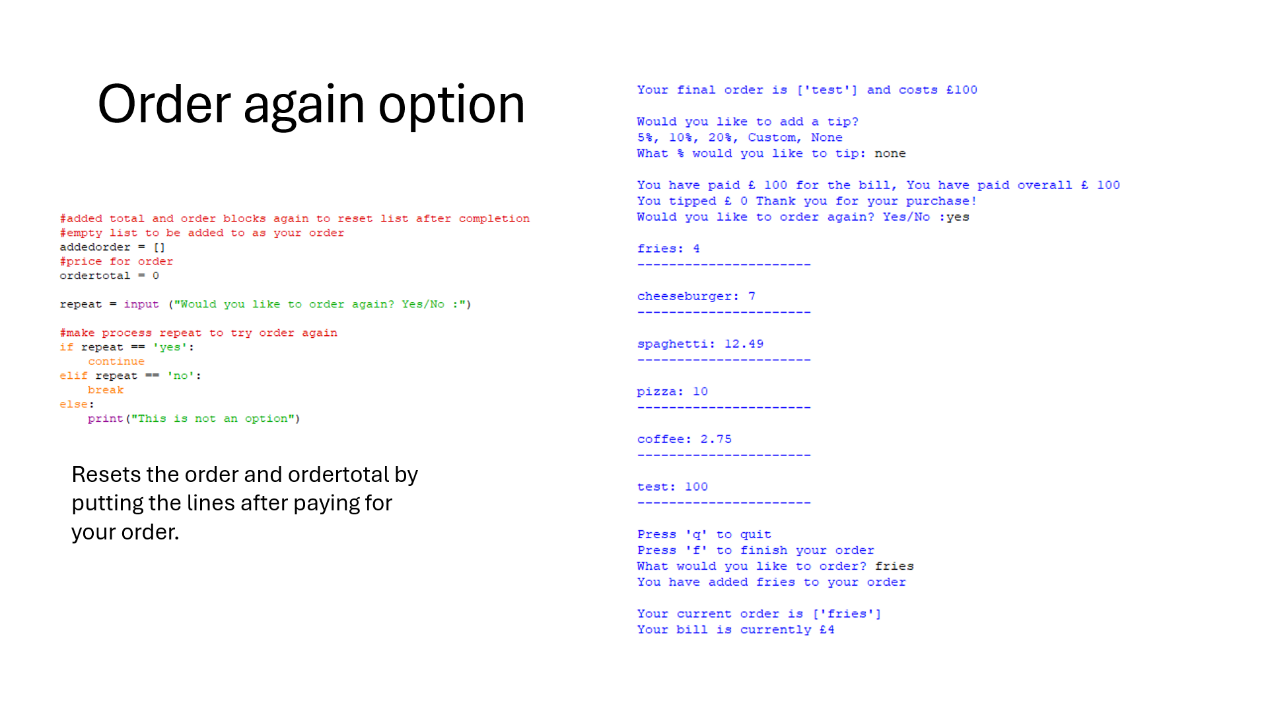
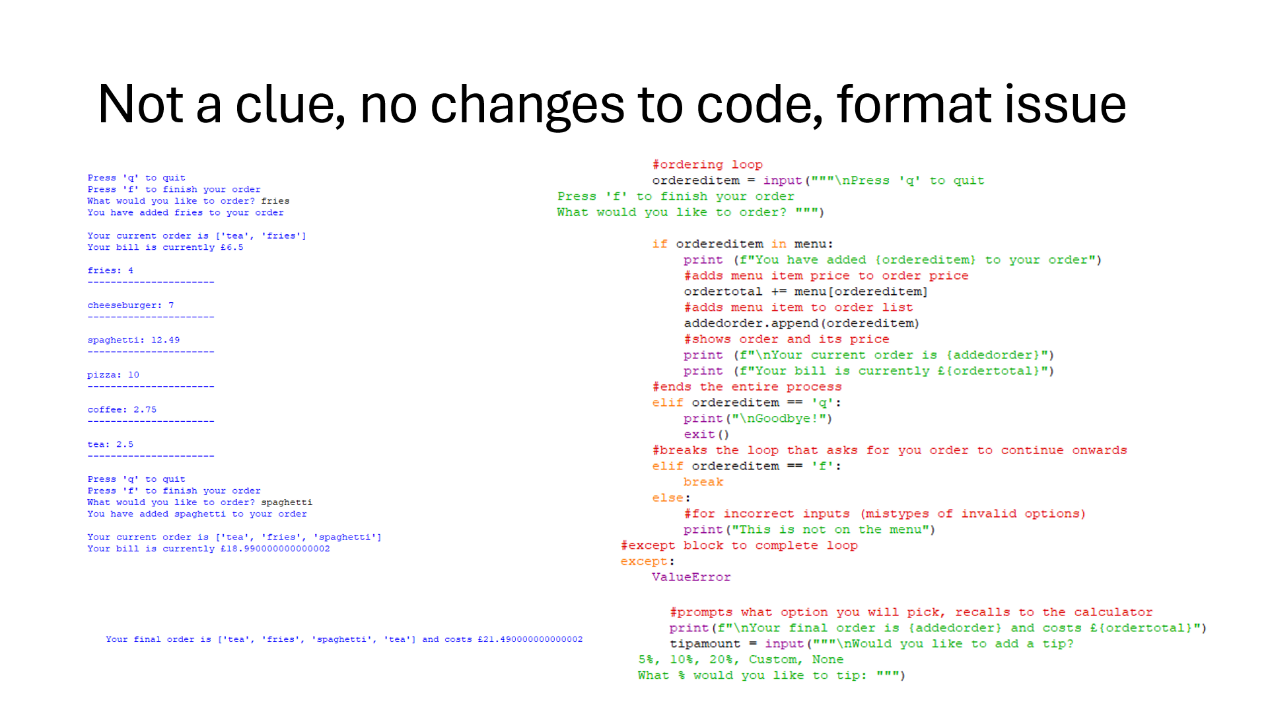
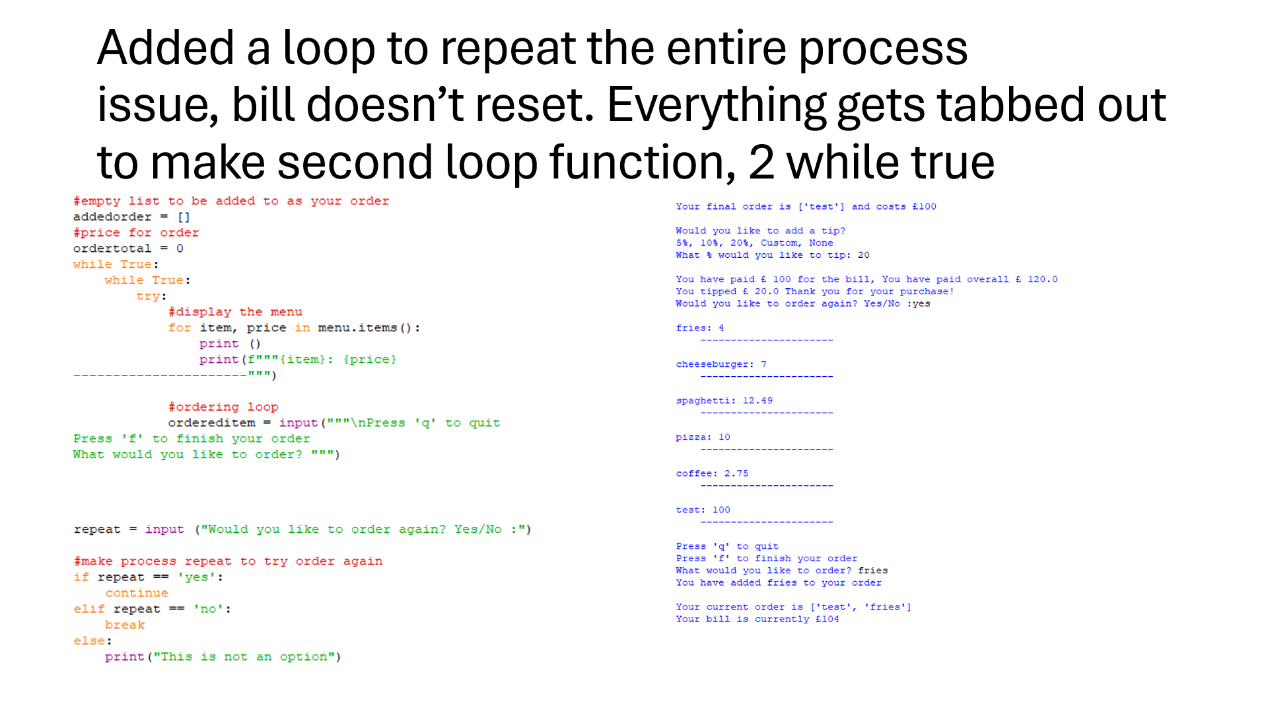
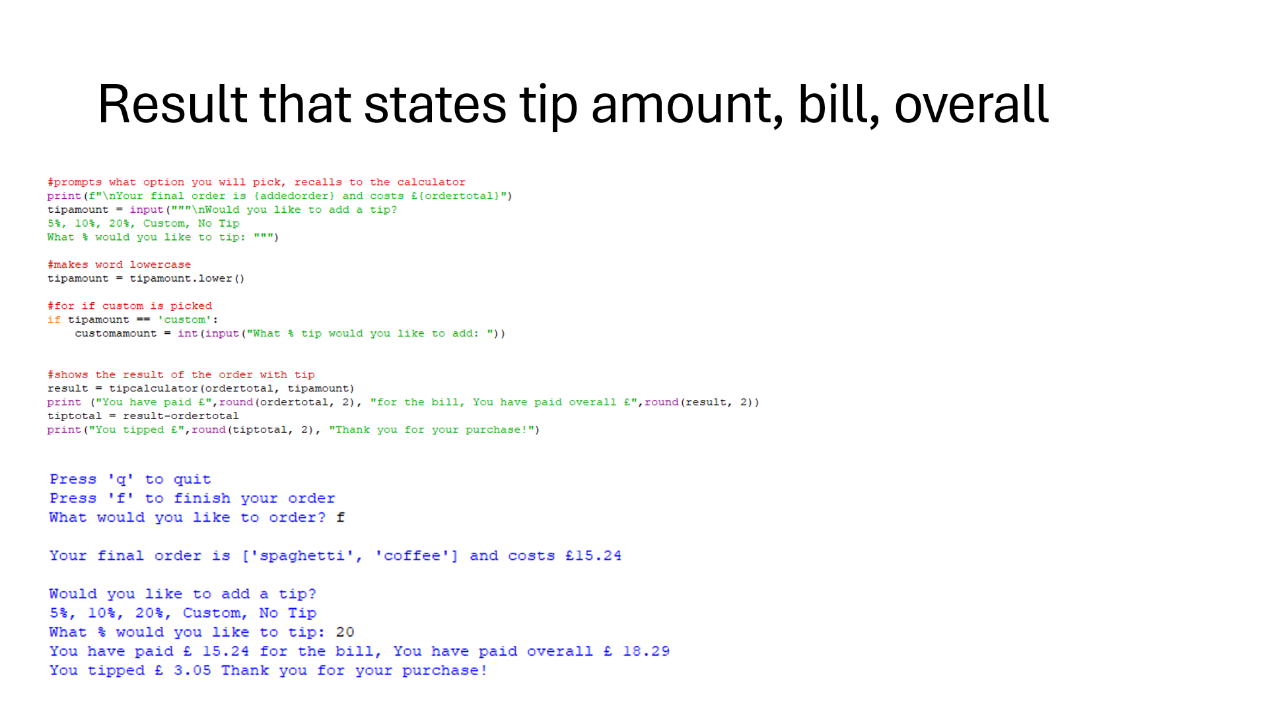
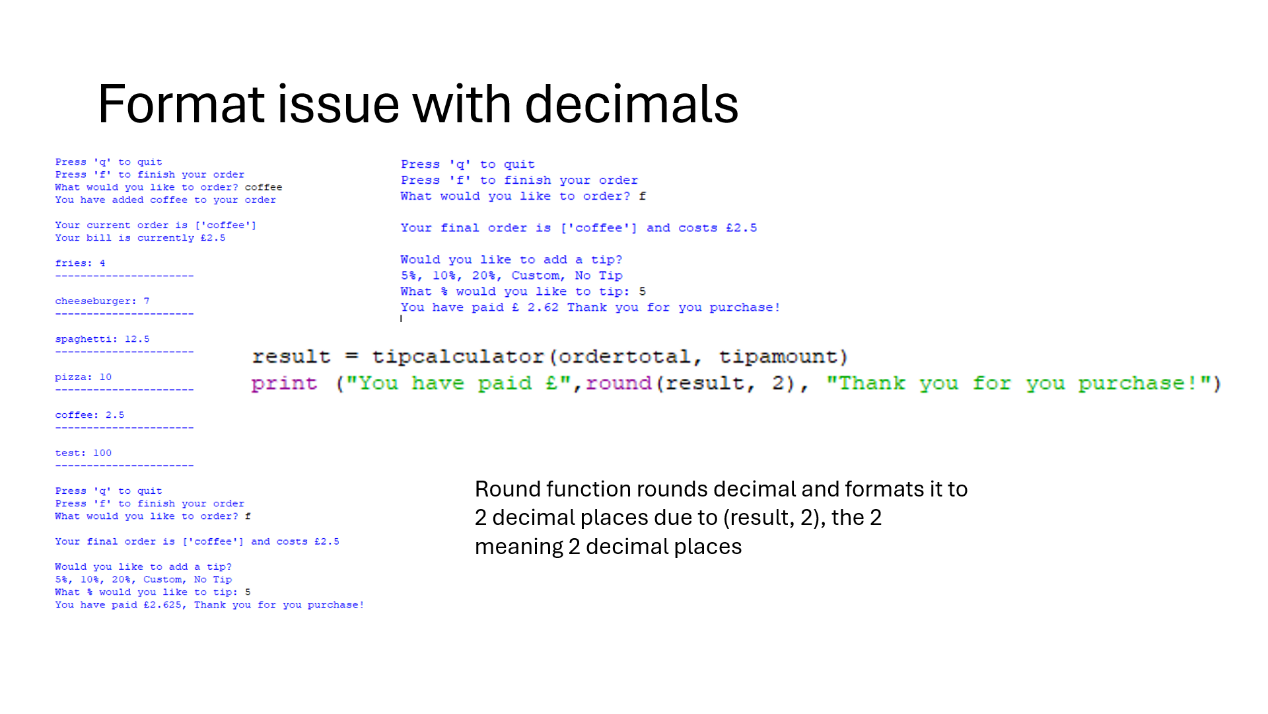
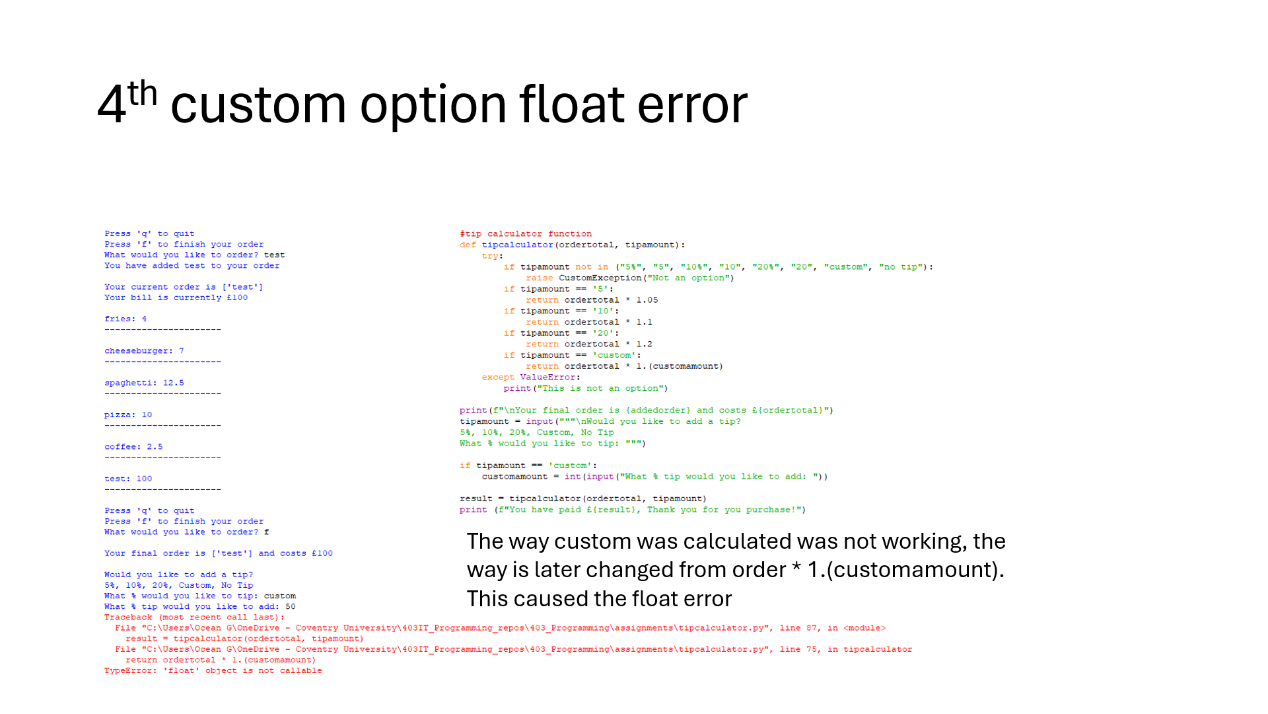
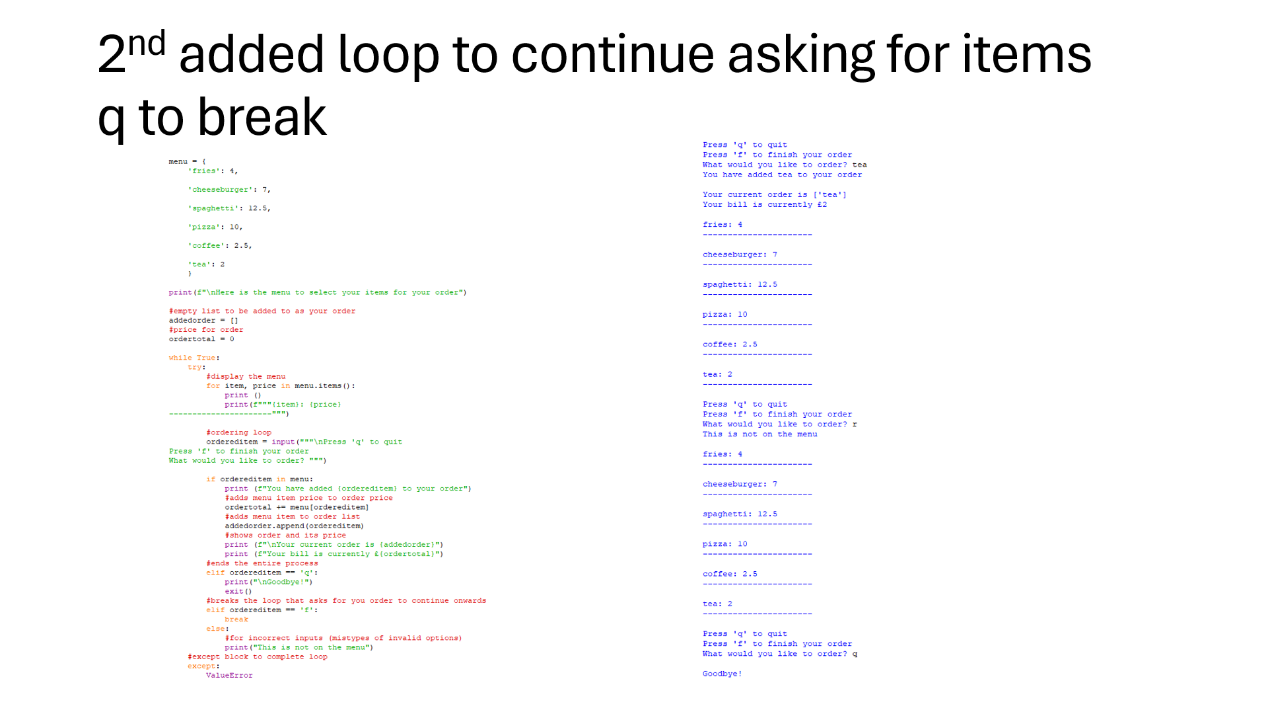
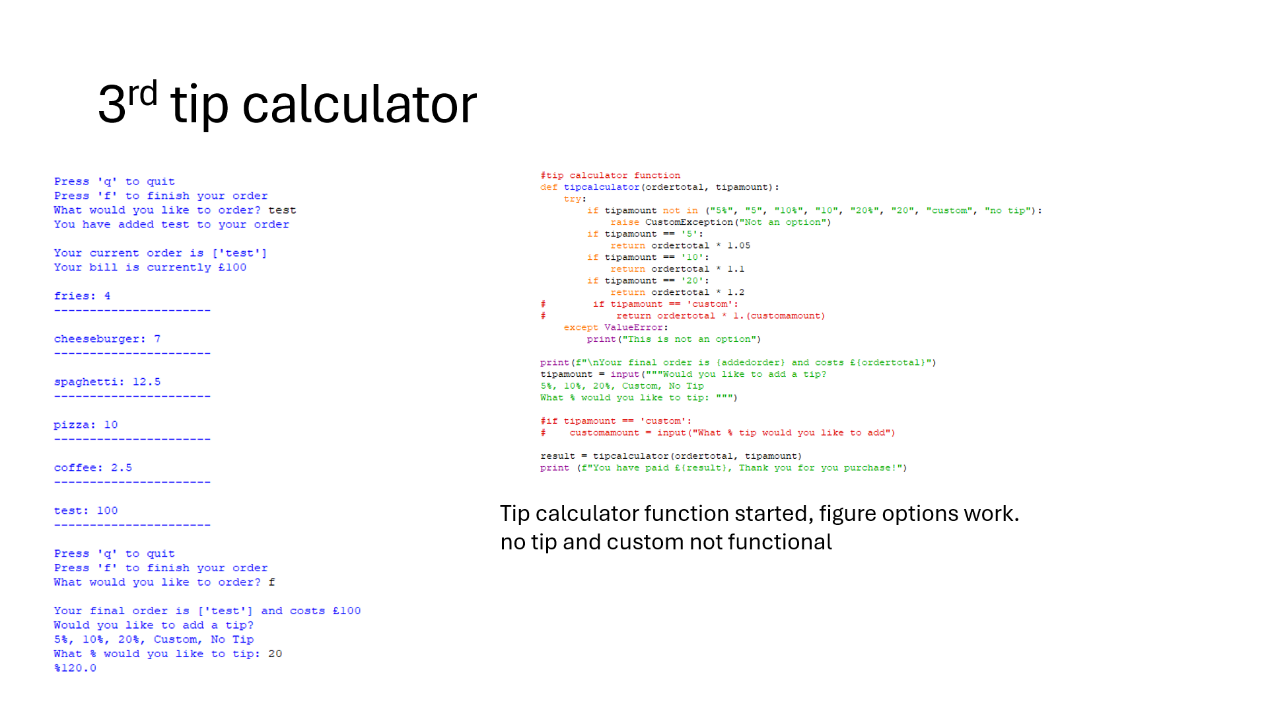


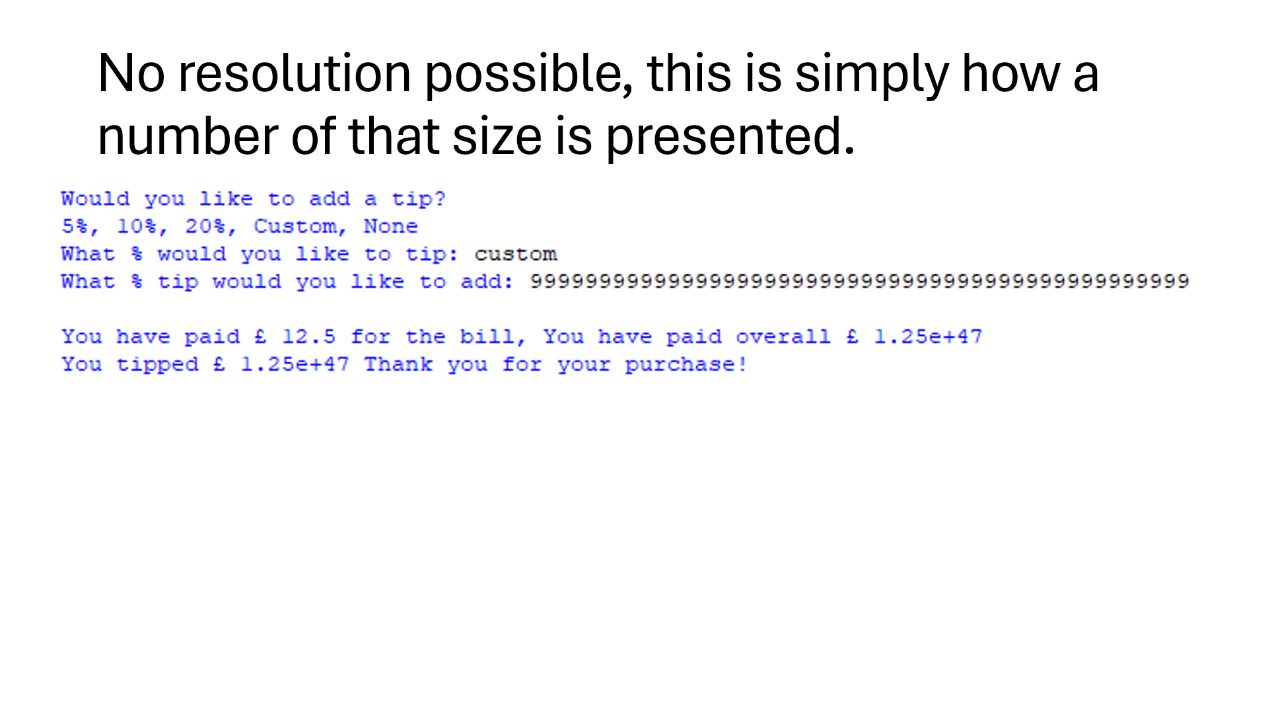
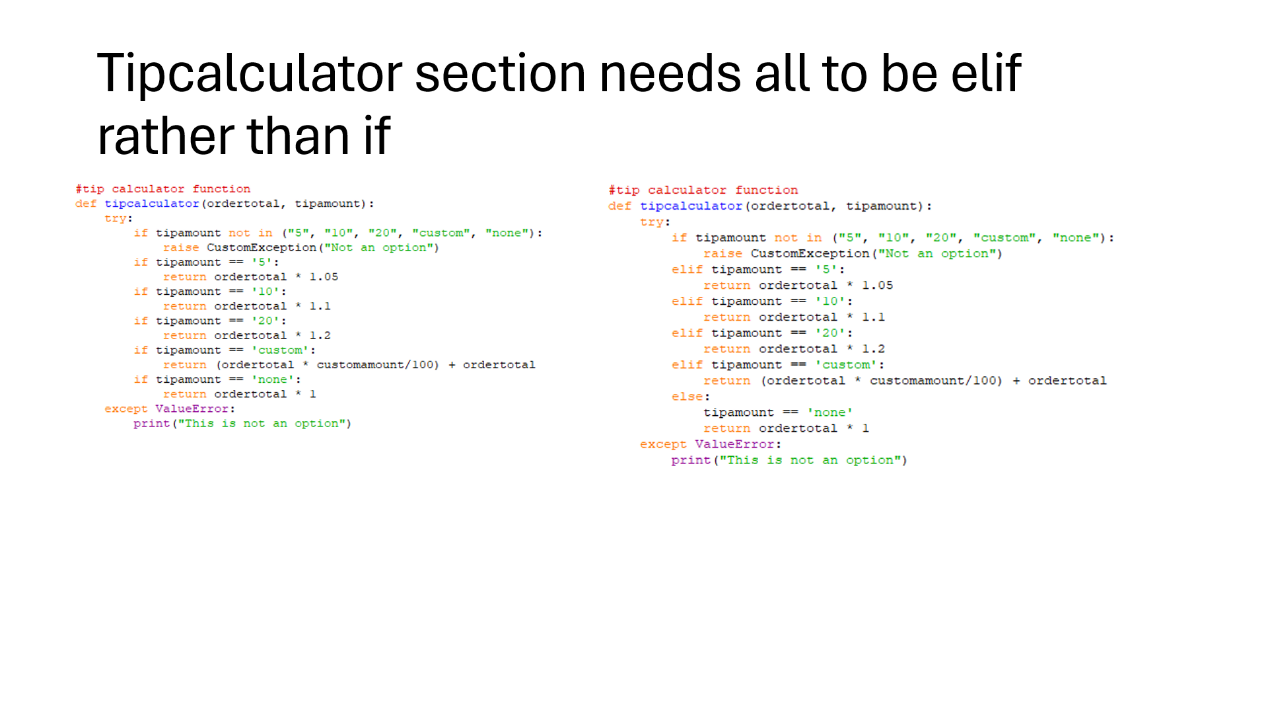
4th solving stage, invalid input

5th solving stage, accepts space

# appendix 3 – Tip calculator timeline







# References

Clark, H. (2024). *7 Stages Of The Software Development Life Cycle (SDLC)*. [online] The Product Manager. Available at: https://theproductmanager.com/topics/software-development-life-cycle/ [Accessed 17 Feb. 2025].

GeeksForGeeks (2024). *Functional Programming Paradigm - GeeksforGeeks*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/functional-programming-paradigm/ [Accessed 18 Feb. 2025].

GeeksforGeeks. (2022). *What is Decomposition Computational Thinking?* [online] Available at: https://www.geeksforgeeks.org/what-is-decomposition-computational-thinking/ [Accessed 13 Feb. 2025].

GitHub (2024). *Understanding the SDLC: Software Development Lifecycle Explained*. [online] GitHub. Available at: https://github.com/resources/articles/software-development/what-is-sdlc [Accessed 17 Feb. 2025].

Gomes, A. and Mendes, A.J. (2007). *Problem Solving in Programming*. [online] Gomes, A.J., & Mendes, A.J. (2007). Problem Solving in Programming. Annual Workshop of the Psychology of Programming Interest Group. Available at: https://www.ppig.org/files/2007-PPIG-19th-Gomes.pdf [Accessed 13 Feb. 2025].

Hartree, D.R. (1950). Automatic Calculating Machines. *The Mathematical Gazette*, 34(310), p.111. doi:https://doi.org/10.2307/3611023.

IBM and Jackson, G. (2024). *What is the software development lifecycle (SDLC)?* [online] Ibm.com. Available at: https://www.ibm.com/think/topics/sdlc.

Martin, R.C. and Martin, M. (2007). *Agile principles, patterns, and practices in C#*. Upper Saddle River, N.J.: Prentice Hall.

Rouse, M. (2020). *What is Abstraction (Computer Science)? - Definition from Techopedia*. [online] Techopedia.com. Available at: https://www.techopedia.com/definition/3736/abstraction-computer-science [Accessed 13 Feb. 2025].

StudySmarter UK. (n.d.). *Decomposition: Meaning, Examples, Pros & Cons | StudySmarter*. [online] Available at: https://www.studysmarter.co.uk/explanations/computer-science/problem-solving-techniques/decomposition-computer-science/ [Accessed 13 Feb. 2025].

Ubah, K. (2021). *What is Pseudocode? How to Use Pseudocode to Solve Coding Problems*. [online] freeCodeCamp.org. Available at: https://www.freecodecamp.org/news/what-is-pseudocode-in-programming/ [Accessed 18 Feb. 2025].

Wieczorkowski, J. and Polak, P. (2012). *INFORMATYKA EKONOMICZNA BUSINESS INFORMATICS 3(25) · 2012 AN APPROACH TO ANALYSIS AND IMPLEMENTATION. FROM THE WATERFALL MODEL TO THE TWO-SEGMENTAL MODEL OF INFORMATION SYSTEMS LIFECYCLE*. [online] Available at: https://dbc.wroc.pl/Content/22594/Wieczorkowski\_Polak\_An\_Approach\_To\_Analysis\_And\_Implementation.pdf [Accessed 17 Feb. 2025].

Xero. (n.d.). *How to Implement Agile Methodologies in Your Business*. [online] Available at: https://www.xero.com/uk/guides/agile-methodologies/?utm\_source=MICROSOFT&utm\_medium=cpc&utm\_campaign=UK+-+NB+-+MF+-+SMB+-+DSA+-+D%26I&utm\_content=Guides&utm\_term=DYNAMIC+SEARCH+ADS&ds\_kid=39700080430922630&gclid=d2d3f11b3bf0173054e196a831b16780&gclsrc=3p.ds&msclkid=d2d3f11b3bf0173054e196a831b16780 [Accessed 17 Feb. 2025].