Class Test 2 - Practice Coding Tasks

Class Test 2 will take place during labs in Week 13

Description

Class test 2 contributes **60%** of the overall module mark and the questions will be surround key concepts, particular those covered the second half of the module. This document sets out seven **practice coding tasks**, which you are strongly encouraged to attempt and revise in advance of the formal class test.

The formal 90-minute class test will take place on campus [insert details of dates and times of when QAHE students will sit the test).

The test will be conducted within Blackboard Ultra. During the test you will only be allowed access to:

- Your own personal solutions (via OneDrive) to the practice coding tasks outlined in this document
- An IDE – e.g. PyCharm, VSCode or Idle
- Any module materials available on BbL
- 12 pages (24 sides) of handwritten notes or typed notes (not screenshots) at point size 12, 1.5 spacing

For each Class Test question, you can expect to be asked to undertake some combination of the following:

- Execute your solution code for a provided input or data file
- Modify your code to perform a task related to the original task
- Identify bugs in sample code that is a faulty solution to the task
- Complete a partially implemented section of code that performs a task related to your solution code

You should expect that the formal Class Test questions will range in complexity, with some questions examining fundamental coding skills and others examining more advanced topics. The test will comprise approximately 12 questions.

Feedback on Class Test 2 will be provided within 20 working days of submission.

Practice Coding Tasks

Programming Fundamentals

Practice Task 1

Write a program that asks the user to enter an even number at the command line interface and returns the sum of all the even numbers up to that value. For example, if the user entered 12, the program returns the value 42 (=2+4+6+8+12).

Practice Task 2

The file optometry.txt (available on Blackboard Ultra) contains the eye colour (brown/ blue/ green) and age of a group of people who attended an eye clinic. Write a program that reads the file and outputs (a) the average age of the group correct to 1 decimal place (dps) and (b) the percentage of the group (correct to 1dps) with each eye colour.

```
Sample file:
# Eye Colour, Age
Brown, 25
Blue, 30
Green, 28
Hazel, 22
Brown, 35
Blue, 27
Green, 40
Hazel, 32
Brown, 29
Blue, 26
```

Practice Task 3

An algorithm, **A1**, takes a list of n integers and performs a computation.

The time taken (in seconds) to run the algorithm on a given computer is found to be of the form.

$$T_1 = 0.0045n^3$$

Field Code Changed

A different algorithm, **A2**, is devised which performs the same computation and it is found to take seconds.

$$T_2 = 0.36n^2 + 0.15n$$

Field Code Changed

Write two functions T1 and T2 which evaluate the above times for a given input n. At what value of n does algorithm A2 become more time-efficient (i.e. takes less time to perform the same computation) than A1?

Data Structures

Practice Task 4

Complete the function (filling in each [Blank x]) to find and return the maximum value in a given list of numbers.

```
def find_max([Blank 1]):
    max_value = [Blank 2]  # Initialize max_value with the first list element
for [Blank 3] in [Blank 4]:  # Iterate through the list
        if [Blank 5]:  # Check if the current number is greater than max_value
            [Blank 6]  # Update max_value to the current number
    return [Blank 7]  # Return the maximum value

result = find_max([7, 2, 9, 4, 5])
print(result)
```

Q. What is the resulting value?

Practice Task 5

You have been hired by a small local store to develop a simple inventory management system that will help track the store's stock. The store sells various products, and you are required to create a program that can add items to the inventory, sell items, and display the current stock. The store owner has requested that the program be able to:

1. Add new items to the inventory.
2. Sell items and deduct the quantity from the inventory.
3. Handle errors, such as attempting to sell more items than are available in stock.
4. Display the current inventory at any given time.

Write a Python program that meets the following requirements using a **dictionary** to track inventory.

Requirements:

1. **Add Items to Inventory**:

   The program should allow items to be added to the inventory. Each item will have a unique name (e.g., "apple", "banana", etc.) and an associated quantity. If the item already exists in the inventory,

the program should increase the quantity by the specified amount. If the item is new, it should be added to the inventory.

2. **Sell Items from Inventory**:
   The program should allow items to be sold by decreasing the quantity of the item in the inventory. If the quantity of the item is insufficient (i.e., trying to sell more than are available), the program should print an error message indicating that there is not enough stock.
3. **Display Current Inventory**:
   The program should allow the store owner to view the current inventory. The program should print out a list of all items and their quantities in the inventory.
4. **Error Handling**:
   If the user tries to sell more items than are available or enters an invalid action, the program should print an error message explaining the problem.

Practice Task 6

Produce a sample file bank_accounts.txt, that contains a list of at least 10 sample account numbers that follow a 7-digit format 501xxxx (where you replace xxxx with unique numbers for each account), comma separated from a corresponding account balance (£) e.g.

```
# Ac No, Balance (£)
5015687, 24400
5014875, 13580
```

Part A. Write a program that reads the contents of the file (using appropriate file I/O code) into a **2-dimensional list**. The program should request the user to enter, via the command line interface, a charge account number before the program checks whether the number is valid by searching for it in the 2d list. If the account number is found in the list, the program should display an output message indicating the number is valid and return the account balance. If the account number is not found in the list, the program should display a message indicating the number is invalid and offer the user a further attempt to enter a correct number. E,g.

```
>> Enter an account number:  5015687
>> Account 5015687 found.
```

Part B. Develop a function that can pickle the 2-d list of bank accounts created in Part A i.e. write the entire 2-d data structure to a binary file *(bank.dat*). Finally, write another function that can unpickle the binary data into the prescribed list format specified by you in Part A. Test the program.

Object Oriented Programming

Practice Task 7

(a) The following pseudo code represents a superclass. Implement it in Python.

Class: Book

Instance Attributes:

- title (string)
- author (string)
- publisher (string)
- page_count (integer)
- price (float)

Method: init (title, author, publisher, page_count, price)

- Set the title
- Set the author
- Set the publisher
- Set the page_count
- Set the price

Method: __str__()

- Return a string in the format:

"Title: [title], Author: [author], Publisher: [publisher], Pages: [page_count], Price: [price]"

Write a test program that will put the following books (assume they are stored in a text file) into a list of Book objects. Ensure that the program also prints out the details of the Book objects and that provision is made for books to be added to the existing text file.

- The Great Gatsby,F. Scott Fitzgerald,Charles Scribner's Sons,218,10.99

- 1984,George Orwell,Secker & Warburg,328,8.99
- To Kill a Mockingbird,Harper Lee,J.B. Lippincott & Co.,281,7.99
- Pride and Prejudice,Jane Austen,T. Egerton,279,9.50
- The Catcher in the Rye,J.D. Salinger,Little, Brown and Company,214,6.99

(b) Following on from part (a), create a subclass called EBook. This will have two additional attributes, file_size (float) and file_format (string). Write appropriate init and str methods. Write a test program that will allow the user to enter in the details of as many EBooks as possible. The Ebooks should be stored in a list.