Introduction

Late submission and plagiarism not allowed; the work handed in must be entirely your own.

Criteria

Score:
70% or more:

• Excellent code documentation (i.e. commens using the "#" in your Python scripts and documentation
strings for all functions that specify precisely what the function does and returns and what the arguments are)
• Excellent use of Python's native methods, code standards, standard data structures and NumPy and/or Pandas dataframes where needed
• Excellent use of relevant data types
• Follows carefully the specification provided (where applicable)
• Excellent code optimisation in terms of result/outcome production and readability
• Generally, an excellent solution, carefully worked out, producing and demonstrating clearly and correctly results/outcomes

60%:
• Good code documentation
• Good use of Python's code standards, standard data structures and NumPy and/or Pandas dataframes where needed
• Good use of relevant data types
• Follows the specification provided (where applicable)
• Good code optimisation in terms of result/outcome production and readability
• Generally, a good solution which produces and demonstrates correct results/outcomes

50% or less:
• No meaningful code documentation
• Code tends to be lengthier and more verbose than needed, or at times difficult to read
• No real thought on the relevance of data types
• Does not follow the specification provided (results/outcomes are produced without addressing the
question).
• A solution that only seems to deliver part of the results without showing some logical code Developments

If you submit solutions to only some of the exercises the mark awarded will be proportional to the percentage of the assignment that has been submitted

1.Write a program that asks the user to input his/her date of birth in the format mm/dd/yyyy.
(Note that this exercise uses the American format, not the European one so 03/25/2022 would be 25th March.)
If the input is not in the correct format or the date is invalid the program should output an appropriate error message. If the input is a valid date the program

should calculate and output the user's age (in years). (You will need to check whether the user has had a birthday this year e.g. someone born in January 2000 will be 22 on 25th March 2022, but someone born in April will still be 21.) The program should additionally output the date in European format (e.g. 25/03/2022 for 25th March).
To calculate the age you will need to obtain today's date; to do this you need to use the line `from datetime import date` and then use `date.today()` to obtain a `date` object containing today's date. You can access the day, month and year of a date `d` using `d.day`, `d.month` and `d.year` (these are all integer values).

2. Write a function that returns a list of all non-prime numbers between two positive integers supplied as arguments. Use this is in a program that asks the user to supply two positive integers, checks that the input is valid, then calls the function and outputs the numbers in the returned list, with 10 numbers per output line.
If the user enters negative numbers or supplies non-numeric input the program should output an appropriate error message; the two numbers should be accepted in either order.
The range should be inclusive; if the user inputs 200 and 351 (or 351 and 200) these two numbers (which are both non-prime) should be included in the output.

3.The three functions for this exercise should be written in a single .py file. You should not submit any code that calls the functions, although it is strongly recommended that you do produce such code in order to test your functions

a) Write a function called `fun1` that takes a string as a parameter and returns True if and only if the string is a palindrome

b) Write a function called `fun2` that takes a string as a parameter, converts the string to upper-case and returns the second-most most frequent letter/digit. (If there are equally frequent letters and digits you may return any one of the second-most frequent letters/digits.) Characters that are neither letters nor digits should be ignored. The function should return `None` if there are fewer than 2 different letters/digits in the string

c) Write a function called `fun3` that takes a string as a parameter, counts the number of upper-case letters, lower-case letters and digits in the string and returns a tuple containing the three counts.

4.Write a function that takes 3 arguments, a list of tuples and two integers denoting salaries.
Each tuple in the list will contain the name, job title and salary of an employee. The function should output the names and job titles of all employees in the list whose salary is greater than or equal to the smaller of the integer arguments and less than or equal to the larger. The output should be displayed one employee per line, sorted by salary (largest first), in a neatly formatted table. If there are no matches an appropriate message should be output.
Write a program that asks the user to supply a file name and attempts to open the file for reading. (If the file could not be opened the user should be asked to supply another filename; this process should continue until a file has been opened successfully.) Each line of the file should contain the name, job title and salary of a single employee (in that order, separated by commas). A typical line may be

```
Gareth Southgate,manager,2500000
```

The program should create an empty list of tuples and then convert the contents of each line of the file into a tuple with 3 elements, which should be added to the list. (You may assume that the contents of the file are in the correct format so there is no need to perform validation).
After the conversion of the file contents to tuples has been completed the program should print the list of tuples (this is simply to verify that the tuples have been created correctly so no formatting is required) then enter a loop in which the user is asked to supply the start and end values of a salary range (inn either order). Assuming the input is valid, the salary values supplied by the user and list of tuples should be passed as arguments in a call to the function described above. (If the input is invalid, e,g, non-numeric or negative salaries, an appropriate error message should be displayed) After returning from the function the user should be asked if he/she wishes to quit or to supply another salary range.
Hint: the elements of the tuple do not have to be stored in the order described above. A different order may make the sorting easier.

5. To be added to this document in next week