## General Instructions

Make sure that you read and then follow the instructions for each task carefully.

Please make sure that you save all of your work to a safe place and that you make **regular back-ups**.

You should begin all tasks with the following steps unless otherwise stated:

- Create a new Python file with the *Project Name* specified in the file.
- It is suggested that you save the file with your completed code (and/or any other files specified) in a folder named for the tutorial week (*week01, week02,* etc.), which should itself be in a root folder called *sdam*. However, you can use whatever method you wish to organise your work – just make sure that each file can be located quickly and efficiently based on the week number and the project name.

If you are in any doubt, please check with your tutor.

## Telephone Bill
## Project Name: telephone_bill

Beginner

Write a program that prompts the user for a positive integer that represents the number of whole minutes to be billed for use of a telephone.

Your program should then calculate the cost of the calls using a variable charge per minute, the amount of VAT based on a constant and the total bill payable.

Your program must prompt the user for the number of minutes and then output the number of minutes, the basic call charge (without VAT), the VAT that is due, and the total for the bill. The following is an example of the expected input/output:

```
Enter number of minutes: 125
Number of minutes used: 125
Basic call charge: £18.75
VAT due: £ 3.75
Total bill: £22.50
```

Fig. 1 Sample Output

Data to be used:

- Call rate = 15 pence per minute
- VAT rate = 20%

Step 1: Design a test plan

- Using the information provided in the lecture, design a test plan that tests your program and store it in a file called *telephone_bill_test.docx*
- Your plan should test the following:
  - What happens when the user enters 0
  - What happens when the user enters 1
  - What happens when the user enters a value higher than 1.
- Your plan should have the following fields
  - Input value
  - Reason for test (for example why you should test 0,1 etc)

- What is the expected output (showing how you calculate the expected output, e.g. the formula you have used to calculate the amount of VAT due).
- The actual result when you ran the program with the specified input including the name of the screenshot file of the output (eg *telephone_bill_test1.jpg*)

Step 2: Write source code

- Write the code necessary to prompt the user for the required data, then output the required data as per Fig 1.

Step 3: Run, test and screenshot your application outputs

- Run the program according to your test plan, for each test input take a screen shot
- Call your screenshot *telephone_bill_testx.jpg* where x is the number of the test

Portfolio

The task contribution to your portfolio is:

- The Python source code file for the task
- *telephone_bill_test.docx* your test plan for the project
- *telephone_bill_test1.jpg*, *telephone_bill_test2.jpg*, *telephone_bill_test3.jpg* showing output from *telephone_bill.py* when tested according to your plan

## Not Quite Average
## Project Name: average

Beginner

The code in Fig. 1 has been written to input 4 integer values, calculate the average of the 4 numbers inputted and then output the average of those values as a float.

The average program:

```
num1 = int(input("Enter first integer: "))
num2 = int(input("Enter second integer: "))
num3 = int(input("Enter third integer: "))
num4 = int(input("Enter fourth integer: "))

avg = (num1 + num2 + num3 + num4) / 4

print("Average is: " + str(avg))
```

Fig. 2 average source code

Step 1: Design a test plan

- Using the information provided in lecture 5, design a test plan that tests the program and store it in a file called *average_test.docx*
- Your plan should test the following:
  - What happens when the user enters 10, 9, 8, 7
  - What happens when the user enters 1, 1, 1, 1
- Your plan should have the following fields
  - Input values

  - What is the expected output (showing how you calculate the expected output, e.g. how you have calculated the average).
  - The actual result when you ran the program with the specified input including the name of the screenshot file of the output (eg *average_test1.jpg*)

Step 2: Write source code

- Use the code provided in Fig. 2

Step 3: Run, test and screenshot your application output

- Run the program according to your test plan
- Call your screenshot *average_test1.jpg*

Step 4: Ensure the correctness of the program

- Ensure the program outputs the correct result
- If it does not, correct any errors in the program
- Run the retested program and capture the new output in a file called *average_test2.jpg*

Portfolio

The task contribution to your portfolio is:

- The Python source code file for the task
- *average_test.docx* your test plan for the project
- *average_test1.jpg*, *average_test2.jpg* showing output from *average.py* when tested according to your plan

## Height Converter
## Project Name: imperial_converter

Beginner

Write a program that asks a user to input their height (in feet and inches).

Your program should then calculate the user's height in kilometres, metres, centimetres, and millimetres. It should then output these conversions with appropriate text.

For example, if the user enters:

```
Feet: 5

Inches: 10
```

Your program should output:

```
Height in kilometres: 0.001778

Height in metres: 1.778

Height in centimetres: 177.8

Height in millimetres: 1778.0
```

Run the program with your height entered and take a screenshot as *imperial_converter.jpg*.

Portfolio

The task contribution to your portfolio is:

- The Python source code file for the task
- *Imperial_converter.jpg* showing the conversion of your height

## Bandwidth Headache
## Project Name: bandwidth

Intermediate

A Network Administrator needs to ascertain whether a new application will have a negative impact on the current performance of their network. To do so they will need to calculate the theoretical maximum bandwidth, the current usage and the new applications usage.

For information, 1 Mbps = 1,000,000 bps.

The calculation should make use of the following data:

- Maximum network bandwidth = 1000 Mbps
- Number of concurrent users = 200;
- Current application bandwidth requirements
  - Application A = 200000 bps
  - Application B = 100000 bps
- The new application will require 350000 bps

Write a program that will calculate the following outputs:

- The network capacity in bps
- The current usage in bps
- The current availability in bps
- The new applications requirements in bps
- The bandwidth available if the new application is installed (in Mbps)

Your program should find that the bandwidth available if the new application is installed is 870 Mbps.

Take a screen shot of the finished output saved as *bandwidth.jpg*.

Portfolio

The task contribution to your portfolio is:

- The Python source code file for the task
- *bandwidth.jpg* showing output from *bandwidth.py* when tested according to your plan

## All portfolio requirements for this tutorial

Beginner

*telephone_bill_test1.jpg*

*telephone_bill_test2.jpg*

*telephone_bill_test3.jpg*

*telephone_bill_test.docx*

*telephone_bill.py*

*average_test1.jpg*

*average_test2.jpg (if required)*

*average_test.docx*

| | |
|---|---|
| | *average.py* |
| | *imperial_converter.py* |
| | *imperial_converter.jpg* |
| Intermediate (opt) | *bandwidth.jpg* |
| | *bandwidth.py* |