CSB110MC – Python Programming

Lauren Powell

The lab classes consist of tasks that are designed to be completed during the lab class. If you do not complete the tasks during the lab class then you should do them at home and have them ready to be checked in the next lab class.

Each task should be stored in its own `.py` file.

Do not use the interpreted mode for doing the tasks (only use the interpreted mode for experimentation). If you store your python code as `.py` files then you can go back and look at them at a later date.

CSB110MC Lab Class 1 Thursday 02/02/2023

☐ Task 1.1

Attempt to execute the following python program. It contains various errors. Your goal is to understand the errors and the error messages. Fix the errors and get the program to execute. It is natural to encounter many of these errors as you program, so it is useful to be able to spot and fix them quickly.

```
print <The answer to the ultimate question is "43>
```

☐ Task 1.2

Start a new python file. Store your age in a variable named `age`.

Get python to print the following:

```
I am <age> years old. In 2 years I will be <age + 2>.
```

where the angular brackets represent the variables.

☐ Task 1.3

Write a program which prompts the user to enter their first and last names (as two separate inputs). The program should then print the initial of their first name followed by a dot and their last name, e.g.,

```
L.Powell
```

☐ Task 1.4

Imagine your are writing a piece of software for a shop. You need to calculate prices of items with VAT added. Assume VAT is charged at 20%.

Write a program that:

- allows the user to input the price in pounds of an item without VAT (as a floating point number), then
- displays the amount of VAT that should be added to the item, then
- displays the price of the item with VAT added.

An example run of the program might be:

```
Please enter the price of the item without VAT: 36.94
VAT: 7.388
Item price inc VAT: 44.33
```

where the number on the first line is input by the user.

Your program should round the final item price to 2 decimal places.

It is common to write you programs in three parts: input, processing and output.

Input This is where you should get the users input. For this task you should prompt the user for the price of the item and store the result in a variable.

Processing This is where you should perform all the calculations. For this task, you will need to calculate the VAT amount and the new price (which includes the VAT). Store both values in new variables.

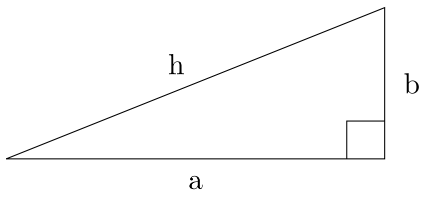
Output This is where you print the output of the program. For this task you should print the VAT amount on one line, followed by the new calculated price (which includes VAT).

It's best to group all the processing together and all the output together (as opposed to mixing processing and output statements).

You should avoid using magic numbers in your program, so the VAT rate should be a constant at the top of the program.

☐ Task 1.5

Write a program that calculates the length of the hypotenuse of a right-angled triangle:



where a, b and h are the length of the respective sides.

h can be calculated as follows:

$$h = \sqrt{a^2 + b^2}$$

An example run of the program might be:

```
Enter the length of side a: 4.3
Enter the length of side b: 2.3
The length of the hypotenuse is 4.88
```

To do this you can import the `sqrt` function from the `math` library. Split your code into three sections again:

Input Ask the user to input the lengths of the sides.

Processing Calculate the length of the hypotenuse.

Output Print the length of the hypotenuse.

☐ Task 1.6

Write a program that reads a number between 10,000 and 99,999 from the user, where the user enters a comma in the input. Then print the number without a comma.

An example run of the program might be:

```
Enter an integer between 10,000 and 99,999: 45,711
45711
```

☐ Task 1.7

In the game of Nim, we start with some number of piles of tokens, and two players alternate in removing one or more tokens from any one pile. The player that takes the last token wins the game.

Write a program that prompts the user to input an integer. This will represent the number of tokens in a pile. The program should print a representation of this pile to the screen. You can do this using string replication.

The program should then **randomly** choose a number of tokens to remove from the pile before printing the new pile to the screen.

An example run of the program might be:

```
How many tokens would you like in the pile?  6

token pile: ******

I remove 4 tokens from the pile!

token pile: **
```