Computer Code for Beginners

Week 1

Variable Swap

The file `varSwap.py` declares two variables, `x` and `y`, with different values and prints them twice.

Try running the program and you should see:

```
x = 10 y = 20
x = 10 y = 20
```

Add to this program to swap the variables x and y, so that the program prints:

```
x = 10 y = 20
x = 20 y = 10
```

Without changing the rest of the program.

You will need to use:

- A new, temporary, variable to let you swap the values over

Logo Shapes

The file `triangle.py` contains a program to draw a green triangle. It uses the `turtle` library, which gives us some simple drawing commands, based on a language called Logo. It imagines that we have a cursor (called the turtle) that we can move across the screen that can draw a line behind it.

Run the program and you should see a window appear with a green triangle.

- `from turtle import *` imports everything from the turtle library
- Turtle Commands this program uses:
  - `penup()` and `pendown()` control whether the turtle draws as it moves
  - `setpos(-200,200)` Moves the turtle to a new position
  - `color('green')` Sets the colour we want to fill the shape with
  - To fill a shape we use `begin_fill()` before drawing the shape and `end_fill()` when we're done
  - `forward(length)` Moves the turtle forwards by the value of length (which is 100)
  - `left(angle)` Turns the turtle left by 120 degrees

Using these commands, add some code to the `square.py` file to make it draw a blue square.

- You can use `colour('blue')` to change the fill colour.
- Think about how to draw a square, step-by-step

Now, add some code to the `rectangle.py` file to make it draw a purple rectangle.

- This will be very similar to the square, but you will need another variable.
- Again, think step-by-step how you draw a rectangle

Maths Quiz

Write a program that quizzes the user on multiplication. It should pick two random numbers, ask the user what the result of multiplying them together is, and tell the user if they got it right or wrong.

The module in the file `quiz.py` picks two random numbers `num1` and `num2`. It uses `import random` to import the `random` library, and `random.randrange(1,10)` to generate a random number between 1 and 9 (because it works like the `range(x, y)` function).

Add to the program to:

- Ask the user for their guess and store it in `guess`
- Check if the user guessed correctly
- Tell the user if they answered correctly or incorrectly

You will need to use:

- `print(string)` to print to the screen
- `str(num)` if you want to convert `num` to a string
- `int(input())` to convert the result of `input()` to an integer
- An `if`...`else` branch to check if the user got the answer right
  - What two things do you need to compare, to check if the user guessed correctly?

When is Multiplication not Multiplication?

We can multiply two numbers by repeated addition. For example 5 * 2 can be rewritten as 5 + 5 and 6 * 4 can be rewritten as 6 + 6 + 6 + 6. Make a new file called `repeatedAddition.py` and write a program that:

- Ask the user for two numbers
- Multiplies them using repeated addition
- Print the result

You will need to use:

- `int(input())` to convert the result of `input()` to an integer
- `str(num)` if you want to convert `num` to a string
- A `while` loop to repeat the addition
  - Which number will you add to itself repeatedly?
  - How many times will you need to repeat the addition?

Logo Shapes 2

Now that we've used a `while` loop (in the exercise above) we're going to update the programs in the Logo Shapes exercise to use loops as well. This will remove some of the repeated code in these programs.

- Open the `triangle.py` file again and look at the lines that draw the triangle (16-20)
  - You'll notice that it repeats the same commands three times
- Make a new variable `sides` and give it the value `3`(since we're drawing a triangle)
- Identify the two commands that we repeat on lines 16-20 and put them into a `while` loop that repeats them for each side
- Run the program and check that it works as you expect
- Can you do the same thing with the `square.py` program?
- Now can you do the same thing with the `rectangle.py` program?
  - Hint: either change the number of times you loop or use an `if`. . . `else` branch inside the loop. . .

What's that in Old Money?

Write a program to convert a temperature in Celsius to Fahrenheit. The formula for the conversion is `F = (9 / 5) * C + 32`, where `F` is the Fahrenheit temperature you want and `C` is the Celsius temperature.

You need to:

- Ask the user for the temperature in Celsius
- Convert this temperate into Fahrenheit using the formula above
- Print the converted temperature

Check that your program works:

- 0C = 32F
- 20C = 68F
- 100C = 212F

Be careful:

- You must use brackets to make sure the division is performed first
- Use a `/` in the conversion, because it produces a decimal number (`float`)
  - Using `//` always produces a whole number (`int`)
  - Change the `/` to a `//` and see how the program now produces the wrong temperature
- What happens if you try to convert 15.5C into F?
  - Change the `int(input(...))` to `float(input(...))` and try again

Challenge:

- Can you change the program so that it continues to ask for a Celsius temperature to convert until the user types in "e"?
  - How will you check that the user has typed in "e"?
  - How will you stop the loop?

Tea Totaller

The module in the file `teaTotaller.py` contains a simple program representing a hot drinks vending machine. The program asks the user how many cups of tea they want and prints the total price. It calculates the total by multiplying the number of cups of tea by the price, which is declared at the beginning of the program.

- Run the program using IDLE and make sure that it works.
  - Note how we use `int(input())` to convert the result of `input()` to an integer
  - Note how we use `str()` to convert integer variables to strings in the `print()` statement

Now, we're going to add some more functionality to our program.

First, we're going to modify it to also ask the user how many cups of coffee they want. It will then print the total price of the requested coffee, and the total price of the requested tea and coffee.

- We need to store a price for each cup of coffee
- We need to duplicate the lines of code that ask the user for their input
  - Change the input statement so that it asks for coffee
  - Store the result in a new variable
- We need to duplicate the lines of code that print the total price of tea
  - Change it so that it now prints the total price of coffee
- Now add a line to print the total cost of both the teas and coffees

Next, we're going to improve the algorithm and the output of the program.

- If the user has asked for 0 teas or 0 coffees, then we don't need to calculate the total price
  - We can use an `if` statement to check the number of teas the user has input
- If the user asks for exactly 1 tea or coffee, we should print "tea" or "coffee" when printing the totals, otherwise we should print "teas" or "coffees"

Finally, we're going to make our program a little more useful by altering it to accept orders of tea and coffee until told to stop.

- We can use a `while` loop to repeat parts of our program so that it:
  - Asks if the user wants to order a drink; continuing if they say yes, or exiting the loop if they say no
  - Asks the user for the number of teas and coffees they want
  - Calculates and prints the totals
- Be careful:
  - Make sure that the user can exit the loop
  - Make sure that only the things we need to do multiple times are inside the loop
- *Challenge*:
  - Print some user feedback if they don't reply yes or no when asked if they want a drink
  - Use `'{:.2f}'.format(num)` to format the float `num` to two decimal places, useful for currency
  - Use `s.lower()` (where `s` is a string) to make sure that the user's input is recognised, even if they use capitals
  - Use `s.isdecimal()` (where `s` is a string) to make sure that the user can't crash the program if they input a non-integer value