Introduction to Python

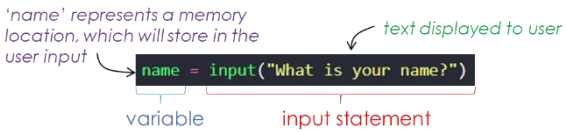
Inputs & Variables

When we code, we often need our programs to receive and store inputs from the user. For example, if our program is to add two numbers provided by the user, we need our program to request these numbers and store them, so it can later find their sum.

In python, this is achieved using the input() script, assigned to a variable.

The Input Script

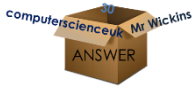
The input() script is set up like this:



- When it is executed (run), the text inside the brackets will display on the screen and the program will pause for a 'user input'.
- Once the user has entered an input, it will be stored in the variable assigned to the input statement (e.g. name).

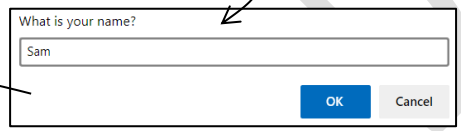
Variables

As mentioned above, we can store data in variables.
- Variables are locations in memory that can store a single piece of data.
- We give variables **labels** and use these labels when assigning data into the variable.
- In the example above, 'name' is the label for the variable that will store the user's input.
- If it helps, think of a variable as being like a storage box, with a label on the front of it, that will store something.



Example Code

```
name = input("What is your name?")
```




```
print("You just typed in " + name)
```

```
You just typed in Sam
```

Outputs

When we wish to output information (strings/numbers) onto the screen, we use the print() statement.

The Print Script

The print() script is set up like this:

text displayed to user

```
print("Hello World")
```

- When it is executed (run), the text inside the brackets will display on the screen.
- Unlike the input()script, there will be no pause and the program will immediately execute the next line of code in the program.

Printing Variables

As shown above, if the print statement contains text with quotes, it will display the text inside the quotes.
However, we can also display the contents of variables using a print() statement.
To do this, we add the variable's label between the brackets, without the quotes.

```
1 text = "Hello World"
2 print(text)
```

```
Hello World
```

Data Types & Maths

Data comes in different forms and in order for computers to store data efficiently and process data correctly, it must be made aware of the type of data that it is storing/processing at any given time.

Data Type	Description
String	Combination of different keyboard characters
Integer	Whole number
Real / Float	Decimal number
Boolean	True / False
Character	Single keyboard character

In python, when an input statement is executed, the input is always stored by the system as a string data type. Which is fine when we enter words and sentences.
However, if we are requesting that the user enters whole numbers that will then be calculated upon in our program, we must inform the system that the entered data is in fact an integer, otherwise the program will not carry out the calculations correctly.

We **cast** data to an **integer type** using the int() function.

```
not_a_number = input("Enter a number)
now_a_number = int(not_a_number)
```

When the number is entered, it is first stored as a string. We use the int() function to cast the inputted value into an integer data type.

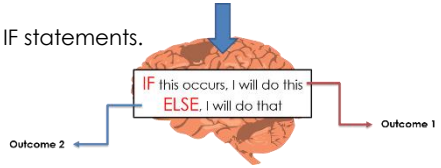
Key Vocabulary

Key Word	Definition
Input	Values which get sent from the user into the computer
Variable	The place where inputs get stored by the program
Output	The values which get sent from the computer to the user
Data Type	The type of data being used by the program
Integer	"Whole Number" data type
Selection	A control structure which allows programs to make decisions

Selection

Selection is a programming construct which allows programs to take different pathways (execute different lines of code), depending on a condition. In other words, it allows programs to make decisions.

This is achieved using IF statements.



The IF-ELSE statement

IF-ELSE statements are set up like this:

Condition being checked

```
1 password = "pa$$wrd"
2 password_attempt = input("Enter your password: ")
3
4 if password == password_attempt:
5     print("Success!")
6 else:
7     print("Incorrect Password")
```

Code executed if condition is TRUE

Code executed if condition is FALSE

In the program above, the contents of the two variables (password and password_attempt) are being compared.

- If they match, the program will run the code under the if statement.
- If they do not match, the program will run the code under the else statement.

Notice that a double equals sign is used to see if the contents of the variables match. *Remember, the single equal sign operator is used to assign values into variables and so it cannot also be used as a comparison operator – so we use 'double equals'.*

Notice also the colon that ends the IF statement line, and the indentation of code underneath. These are vital, so remember the colon and indentation when you write IF statements!