The purpose of functions is to write code that is:

cleaner, meaning it is easier to understand and maintain

shorter

and more re-usable, within the same program and in other programs

Let’s say we write a program

And there are two blocks of code of which there are multiple identical instances in the program.

We can use functions

We write a function containing one of the blocks of code

And a function containing the other

And then we ‘call’ the functions wherever we want the blocks of code to be used in the program.

This makes the overall code shorter

and means that if we edit a function, the change will occur in every place in the program that it is called.

Note that functions must be defined in the program before they are called.

\*

A function definition begins with the def keyword and includes the function name and any input arguments.

The indented code that follows is executed every time the function is called.

\*

Pause the video and attempt these exercises on functions.

Videos showing worked solutions, and the code, can be found on blackboard.

\*

Any number of arguments can be input, where a \* appears between the parentheses in the function definition.

The arguments are encapsulated in a tuple which is assigned to the name that follows the star.

This function will print the sum of any number of arguments.

---------

Default arguments have a default value given in the function definition.

If a new value is not assigned when the function is called, the default value is used.

Here the default values of y and z are used.

Here a new value of z is set.

\*

Pause the video and attempt these exercises on default arguments and multiple arguments. Worked solutions can be found on blackboard.

\*

Global variables can be accessed anywhere in your program, after the are created, including inside of functions.

Local variables can only be accessed inside of functions.

A global variable can be created and assigned a value within a function that can then be accessed outside of the function.

But ordinarily, if a variable is called outside of the function in which it was created, an error will be generated. For example, if this code is a continuation of the program above.

\*

Let’s review reading and writing files using python.

We can open a file with the open function, specifying the path to the file and the mode in which we want to open it.

\*

Python returns an iterable ‘file’ object containing each line in the file.

This can be used like other iterable objects such as lists and strings.

For example we can use a for loop to iterate through, performing an operation on each ilne.

-----------

In this example, the file is instead opened in read mode

the key and value of each item in the dictionary, ‘scores’ is written to the file.

Remember, files must always be closed after they are opened.

\*

To help with this, we can use the structure shown, where the keyword, with is used before the function, open. The file closes automatically at the end of the indented code block that follows.