**Python**

In this course we will be using Wingware's Python IDE for our demonstrations. Wingware's Python IDE is an Integrated Development Environment designed specifically for the Python programming language.

Python is an interpreted language because Python programs are executed by an interpreter. There are two ways to use the interpreter, the interactive mode and the script mode.



***Script Mode***

***Interactive Mode***

*Interactive Mode*

In interactive mode, you type Python commands and the interpreter processes the result, for example:

**>>> print (1+2) *Result:* 3**

The chevron **>>>** is the prompt the interpreter uses to indicate that it is ready. In the above example **print (1+2)** was entered and the interpreter returned **3**. Note **print()** is a python built-in function.

*Script Mode*

Code is stored in a file and uses the interpreter to execute the contents of the file. Python scripts end with the extension .py. From here on we will be using script mode for all our examples.

Traditionally the first program new programmer write in a new language is called ‘Hello, World!’ because it displays the text 'Hello, World!'.

Try it:

**print ('Hello, World!') *Result:* Hello, World!**

**Data Types**

A value(like a letter or a number) is one of the basic things a program works with. These values belong to different data types:

**2** is an integer ***Integers belong to the type int***

**'Hello, World!'** is a string ***Strings belong to the type str***

**3.5** is a float ***Decimal point belong to a type called float***

Strings can be identified because they are enclosed in quotation marks. For example:

**'1+2'** ***This is a string***

**1+2** ***This is an integer***

*Exercise: Determine the output of the following print statements:*

**print (1,000,000) *Result:* 1 0 0**   ***Have we printed one valve or 3 values?***

*Try this:*

**print (1,100,100) *Result:* 1 100 100**

***Is this one or 3 values?***

*Try this:*

**Print (1000000)**

**1000000**  ***This is one value.***

Note: The comma-separated sequence of integers, are interpreted as separate integers.

**Variables**

A variable is a name that refers to a value. An assignment statement creates new variables and gives them values:

**a = 'Hello, World!' Hello, World! *is assigned to the variable*****a**

*To identify the variable type for variable* **a** *(above), enter the following command:*

**Print(type(a))**

***Result:***

**<class 'str'>**

**b = 2 2 *is assigned to the variable* b**

**c = 3.5 3.5 *is assigned to the variable* c**

Variable names should be meaningful. They can contain letters and numbers, but they have to begin with a letter. They can contain uppercase letters, but always good practice the variable name begins with a lowercase letter.

The underscore character is often employed to separate variable names containing multiple words, for example **my\_variable**.

Camel case is another naming convention used for variable names in programming, with this convention multiple words are joined together (concatenated) as a single word. The first letter of the first word is lowercase, and the first letter of each subsequent word is capitalised, for example **myVariable**.

Python keywords cannot be used as variable names; Python has a number of keywords:

**and del from not while**

**as elif global or with**

**assert else if pass yield**

**break except import print**

**class exec in raise**

**continue finally is return**

**def for lambda try**

*Try this (this is a script):*

|  |
| --- |
| **numA = 5****numB = 6****total = numA + numB** **name='Donald Duck'****print( name, ' insists that ', numA, ' + ', numB , ' = ', total )** |

**The Input Function**

The python built-in input function is used to accept a string value entered by a user from a keyboard.

*Try this:*

**name = input('Enter your name: ')**

**print ('Your name is: ', name)**

*Check the data type:*

**type(name)**

**<class 'str'>**

*Let us try another example:*

**num1 = input('Enter your favourite number: ')**

**print ('Your favourite number is: ', num1)**

*Check the data type:*

**type(num1)**

**<class 'str'>**

Python has a built-in function to convert a string into a number. The function is **int()**.

*Try this and check the data type:*

**num2 = input('Enter your favourite number: ')**

**num=int(num2)**

**type(num2)**

**<class 'int'>**

*Will this work? Try it and check the data type:*

**num3 = int(input('Enter your favourite number: '))**

**type(num3)**

**The Basic if Statement**

The **if** statement used check if a condition is true.

*Exercise: Attempt to explain the* if *statement by creating and running the following program:*

**value = int(input('Please enter a number: '))**

**if value < 20:**

 **print('You entered a value less than 20')**

**if value > 20:**

 **print('You entered a value greater than 20')**

**if value ==** **20:**

 **print('You entered the value 20')**

**print('Good bye')**

Please note, the two **==** are used to compare two bits of data.

**User Defined Functions**

User defined functions (also known as methods) allow you to group blocks of code that are executed together. A function can be called (used) over and over again. The keyword **def** (definition used to define a function) is followed by the function name, parentheses **()** and then a colon **:**.

*Try this:*

**def abc1():**

**print ('Hello, World!')**

**abc1()**

The user defined name of the above function is **abc1**, when the **abc1** function is called **abc1()** it will print the text **'Hello, World!'.**

Functions can hold arguments (args). An argument is a value provided to a function when it is called. Arguments contain parameters, parameters are names used to identify the arguments in a function. When a function is called, each parameter is assigned one of the argument values.

*Try this:*

**def abc2(text):**

 **print(text,'is enjoying programming.')**

**abc2('Michael')**  ***Result:*** **Michael is enjoying programming.**

*A closer look at the above function:*

**Abc2('Michael')**calls the user defined functioned named **abc2()** and passes the string value **'Michael'** to the function. Note: the parameter **text** within the **abc2()** function is assigned the value **Michael**.

The **abc2()** function contains yet another function, the **print()**function. The **print()function** prints the value assigned to the parameter **text** (the value being **Michael)**, and the string **is enjoying programming.**

*Exercise: Determine the output of the following function:*

*Function abc3:*

**def abc3 ( x, y ):**

 **print( x + y)**

**abc3( 5, 10 )**

The function contains 2 arguments. The first argument is assigned to the parameter **x** and the second is assigned to the parameter **y**.

Python functions can also contain keyword arguments (kwargs), keyword arguments are assigned default values.

*Attempt to describe the following function:*

**def abc4(x, y='likes programming'):**

 **print(x)**

 **print(y)**

 **print(x, y)**

**abc4('Michael')**

Can you identify the arg and the kwarg within the above function?

*Work out what is happening with the following statements?*

|  |  |
| --- | --- |
| **#Function abc5:****def abc5 ( x, y ):** **return x + y** **print(abc5( 5, 10 ))** | **#Function abc6:** **def abc6():** **return "Hello, world!"****print(abc6())** |

The **return** statement will be discussed in a latter session.