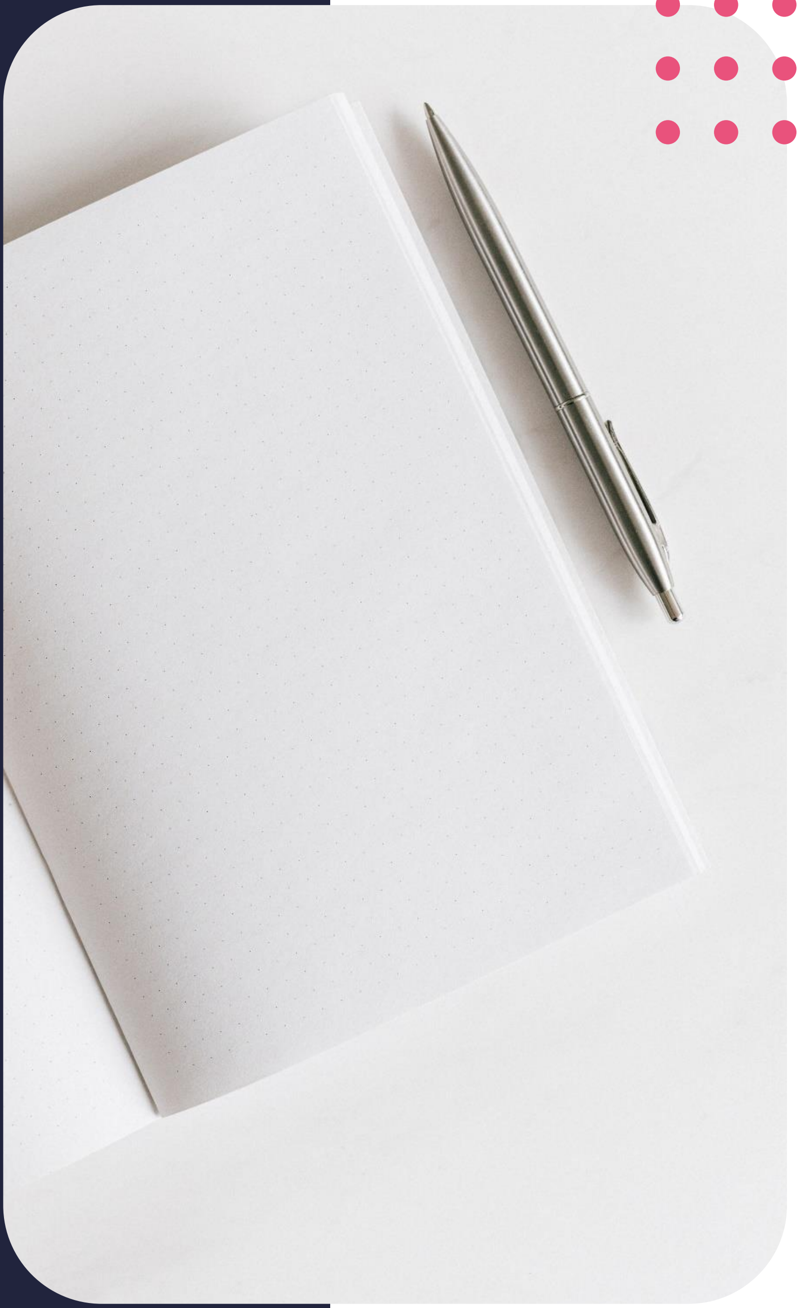
Diploma in Python Programming

Python Basic Data Types

Contents



Introduction

We had an in-depth look the basic Python data types and their use cases. We also concluded that everything in Python programming is an object and every value in Python has a data type. Data types are called “instances of these classes’ and an instance of a class is then referred to as an ‘object’.

Python basic data types

Numbers

We started by identifying the three main number data types which is:

- Integers
- Floating-point number and
- Complex numbers

Integers

In Python, integers are whole numbers that can be negative or positive and this includes naught. With this data type you can perform standard addition, subtraction, division, and multiplication operations in Python.

In our example below we are performing an addition operation, and the same can be done for the rest as mentioned above.

Note in this example, the print() function prints the given object to the standard output device (screen) or to the text stream file.

$$Input \tag{1}$$
$$print(2 + 3) \tag{2}$$
$$Output \tag{3}$$
$$5 \tag{4}$$

For our following example you use the Python built-in function type() to check which class variable a value belongs to and you use the built-in function isinstance() to check whether an object belongs to a particular class.

$$Input \tag{17}$$
$$print(type(4)) \tag{18}$$
$$Output \tag{19}$$
$$< class \qquad {}^0int^0 > \tag{20}$$

Floating-point numbers

Floating-point numbers are values with decimal points, such as 5.0 and 3.2. Just like integers, you can perform standard arithmetic operations with floating-point numbers.

$$Input \tag{25}$$
$$print(type(5.0)) \tag{26}$$
$$Output \tag{27}$$
$$< class \quad {}^0float^0 > \tag{28}$$

Complex numbers

Complex numbers are represented as <real part> + <imaginary part>j, where the real part and imaginary are real numbers.

String data type

Strings are sequences of character data and are immutable (which means it is unchangeable). You use single or double quotes to represent strings. Multi-line strings can be denoted by using triple quotes ^{000} or more ^{000000}.

$$Input \tag{25}$$
$$print({}^0Hello \quad I \quad am \quad a \quad string^0) \tag{26}$$
$$Output \tag{27}$$
$$I \quad am \quad a \quad string \tag{28}$$

Multi-line strings

If you want to extend a string over multiple lines, you cannot use single quotes, you will get an error from your code. This is when you would use triple quotes.

Boolean data type

Objects of a Boolean data type may have one of two values: True or False. Let's look at an example that displays a true value and the term ^{0}bool^0 is an abbreviation for Boolean:

$$Input \tag{69}$$
$$print(type(True)) \tag{70}$$
$$Output \tag{71}$$
$$<class \quad {}^0bool^0> \tag{72}$$

Conversion between data types

You convert between data types using different type conversion functions, such as int(), float() and str(). Lets recap by looking at this example:

$$Input \tag{77}$$

```
print(float(5))                                   (78)
Output                                            (79)
5.0                                               (80)
```

And now the integer 5 is converted into a float 5.0 data type.

Python variables

I this section we have introduced you to Python variables and literals. Here we learnt what they were and how they were applied.

Python variables

As we have discovered that everything in Python is an object and this includes variables. We also explored what a variable is. We have made the analogy of a container, thinking of a variable as a container that stores data. Here's a quick recap example of what we have discovered together.

```
Input                                             (93)
num = 5                                           (94)
Print(num)                                        (95)
Output                                            (96)
5                                                 (97)
```

Remember a variable is not only limited to a number, string or Boolean data type, there are many other data types that may be used in a variable.

Assigning multiple values to multiple variables

In this section we learned that we could assign a value to multiple variables to maximise readability. To refresh our memory, let us have a look at an example.

```
Input                                             (129)
age,name,salary = 77                              (130)
print(age)                                        (131)
Print(name)                                       (132)
Print(salary)                                     (133)
Output                                            (134)
77                                                (135)
77                                                (136)
77                                                (137)
```

Rules and naming convention for variables

We have learnt that our naming convention is very important when naming our variable. Lets have a quick recap on the 6 things we need to consider when naming our variables.

1. Variable names should have a combination of letters in uppercase (A to Z), or lowercase (a to z), or digits (0 to 9).
2. Variable names should make sense
3. Variable names with two words must include an underscore to separate the words.
4. Variable names in Python, are case sensitive.
5. Variable names should never start with a digit.
6. Variable names should never include characters such as @,!,#,%,&.

Remember

Never to use Python key words to name your variables.

Object identity

Every object in Python that is created in a program is given a unique identity and once the object is deleted and is no longer referenced in the program it loses its identifying number and becomes null and void.

Python literals

As we have discovered that there are many different types of literals in Python Programming but in this lesson, we only looked at these three:

- Numeric literal
- String literal
- Boolean literal

Numeric literal

The important thing to remember about a numeric literal is that it is unchangeable. And a numeric literal can be categorised in these three types.

- Integer
- Float-point
- complex

String literal

A string literal is a sequence of characters surrounded by either single quotes or double quotes.

Boolean literal

A Boolean literal has either a True or False value.

Python statement

In this section we learnt about the importance of indentation and comments.

Python statements

A Python statement is an instruction that a Python interpreter can execute.

Multi-line statements

You can choose to extend the Python statement onto the next line or over multiple lines by using a line continuation character (\).

Python indentation

Python uses indentation to define and write a code block. Other programming languages use curly braces {} to define and write code blocks. A code block in Python starts with an indentation and ends with the first un-indented line.

Remember

When using indentation in your code, be sure to keep it consistent throughout your code block.

Python comments

We have learned that comments help interpreters to decipher in words what is happening inside the code and they also help others to better understand what your program is doing.

Multiple line comments

It is also possible to have a comment that extends over multiple lines. Here's a quick recap.

```
Input                                   (267)
#First comment                          (268)
#Second comment                         (269)
Num='Hello'                             (270)
Print(num)                              (271)
Output                                  (272)
Hello                                   (273)
```