## Python Data Types

**Python data types** are actually classes, and the defined variables are their instances or objects. Since Python is dynamically typed, the data type of a variable is determined at runtime based on the assigned value.

In general, the data types are used to define the type of a variable. It represents the type of data we are going to store in a variable and determines what operations can be done on it.

Each programming language has its own classification of data items. With these datatypes, we can store different types of data values.

## Python Numeric Data Types

Python numeric data types store numeric values. Number objects are created when you assign a value to them. For example −

a = 10      # int data type

b = 10.023  # float data type

c = 10+3j   # complex data type

print(a)

print(b)

print(c)

output:

10

10.023

(10+3j)

Python supports three different numerical types and each of them have built-in classes in Python library, called int, float and complex respectively −

* int (signed integers)
* float (floating point real values)
* complex (complex numbers)

Here are some examples of numbers −

|  |  |  |
| --- | --- | --- |
| **int** | **float** | **complex** |
| 10 | 0.0 | 3.14j |
| 0O777 | 15.20 | 45.j |

## Python Sequence Data Types

Sequence is a collection data type. It is an ordered collection of items. Items in the sequence have a positional index starting with 0. It is conceptually similar to an array in C or C++. There are following three sequence data types defined in Python.

* List Data Type
* Tuple Data Type
* Range Data Type

Python sequences are bounded and iterable - Whenever we say an iterable in Python, it means a sequence data type (for example, a list).

### (a) Python List Data Type

[Python Lists](https://www.tutorialspoint.com/python/python_lists.htm) are the most versatile compound data types. A Python list contains items separated by commas and enclosed within square brackets ([]). To some extent, Python lists are similar to arrays in C. One difference between them is that all the items belonging to a Python list can be of different data type where as C array can store elements related to a particular data type.

languages = ["Swift", "Java", "Python"]

print(languages)

output:

['Swift', 'Java', 'Python']

### (b) Python Tuple Data Type

[Python tuple](https://www.tutorialspoint.com/python/python_tuples.htm) is another sequence data type that is similar to a list. A Python tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses (...).

A tuple is also a sequence, hence each item in the tuple has an index referring to its position in the collection. The index starts from 0.

product = ('Xbox', 499.99)

print(product)

output:

('Xbox', 499.99)

The main differences between lists and tuples are: Lists are enclosed in brackets ( [ ] ) and their elements and size can be changed i.e. lists are mutable, while tuples are enclosed in parentheses ( ( ) ) and cannot be updated (immutable). Tuples can be thought of as **read-only** lists.

### (c) Python Range Data Type

A Python range is an immutable sequence of numbers which is typically used to iterate through a specific number of items.

It is represented by the **Range** class. The constructor of this class accepts a sequence of numbers starting from 0 and increments to 1 until it reaches a specified number. Following is the syntax of the function −

range(start, stop, step)

Here is the description of the parameters used −

* **start**: Integer number to specify starting position,(Its optional, Default: 0)
* **stop**: Integer number to specify ending position (It's mandatory)
* **step**: Integer number to specify increment, (Its optional, Default: 1)

### Example of Range Data Type

Following is a program which uses for loop to print number from 0 to 4 –

for i in range(5):

  print(i)

output:

0

1

2

3

4

Again, let's modify the program to print the number starting from 1 but with an increment of 2 instead of 1:

for i in range(1, 10, 2):

  print(i)

output:

1

3

5

7

9

1. Python String Data Type

String is a sequence of characters represented by either single or double quotes. For example,

name = 'Python'

print(name)

message = 'Python programmers'

print(message)

output:

Python

Python programmers

In the above example, we have created string-type variables: name and message with values 'Python' and 'Pythonprogrammers' respectively.

## Python Set Data Type

Set is an unordered collection of unique items. Set is defined by values separated by commas inside braces { }. For example,

student\_id = {112, 114, 116, 118, 115}

print(student\_id)

print(type(student\_id))

output:

{112, 114, 115, 116, 118}

<class 'set'>

Here, we have created a set named **student\_id** with **5** integer values.

Since sets are unordered collections, indexing has no meaning. Hence, the slicing operator [] does not work.

## Python Dictionary Data Type

Python dictionary is an ordered collection of items. It stores elements in key/value pairs.

Here, keys are unique identifiers that are associated with each value.

Let's see an example,

capital\_city = {'Nepal': 'Kathmandu', 'Italy': 'Rome', 'England': 'London'}

print(capital\_city)

output:

{'Nepal': 'Kathmandu', 'Italy': 'Rome', 'England': 'London'}

In the above example, we have created a dictionary named capital\_city. Here,

1. **Keys** are 'Nepal', 'Italy', 'England'
2. **Values** are 'Kathmandu', 'Rome', 'London'

## Python Boolean Data Type

Python **boolean** type is one of built-in data types which represents one of the two values either **True** or **False**. Python **bool()** function allows you to evaluate the value of any expression and returns either True or False based on the expression.

A Boolean number has only two possible values, as represented by the keywords, **True** and **False**. They correspond to integer 1 and 0 respectively.

### **Example of Boolean Data Type**

Following is a program which prints the value of boolean variables a and b −

a = True

print(a)

print(type(a))

output:

true

<class 'bool'>