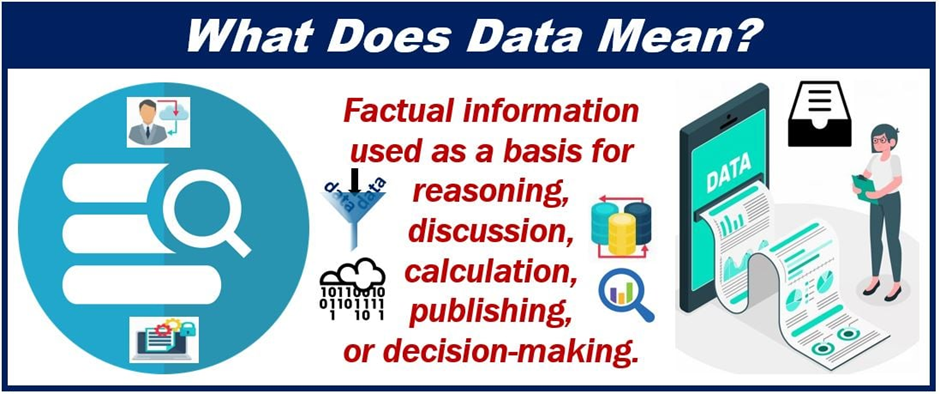
# **what is a data?**

# **Data is a measure of fact or information**



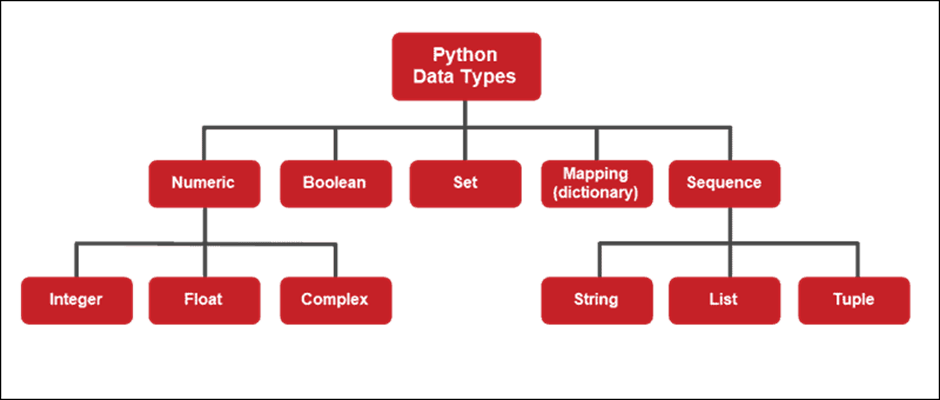
# **What is a data type?**

In general data type is nothing but which form actual data is in

Data means generally what we see around us like in social media and PDFS and documents

Everything is data from that data are there any differences between the data yes there will be a difference between the different types of data.so that difference is nothing but data type

So datatype is the type of data.



In general data type is nothing but which form actual data is in

Data means generally what we see around us like in social media and PDFS and documents

Everything is data from that data are there any differences between the data yes there will be a difference between the different types of data.so that difference is nothing but data type

## **Overview**

In Python, Data types are the classification of different types of values.  
There are several built-in data types in Python. They are mainly categorized as Numeric, Sequence Type, Dictionary, Boolean and Set data types.

## **Introduction to Data types in Python**

In programming, we encounter many types of values, some are sentences, some are [numbers](https://www.scaler.com/topics/python/numbers-in-python/), some contain more than one item, and so on. The marks of a student are a number, therefore an integer data type in Python, similarly the name of a student is a word, therefore a string data type in Python.  
In Python every value has a data type, since everything is an object in Python, data types are actually classes, variables being their objects.

Before proceeding further it's important to know that data types are also classified on the basis of their mutability. [Mutable data types](https://www.scaler.com/topics/mutable-data-types-in-python/) can be modified after creation, whereas Immutable data types can't be modified after creation.

## **Data Types in Python**

Python is an object-oriented high-level programming language that offers a vast variety of data types helping in the implementation of various applications.

Following are the data types in Python

* Numeric Data types
* Sequence Data types
* Dictionaries in Python
* Booleans in Python
* Sets in Python

## **1. Numeric Data types in Python**

These data types represent data with a numeric value, there are 3 types of numeric value data types in Python:

* **Integers in Python**Integers, as you know in mathematics are numbers without any fractional part. They can be 0, positive or negative. There is no limit to how long an integer can be in Python. They are represented by int class.

Syntax:

Integer numbers are of int type. It is just written as a number. Here's an example of Integer data type:

# Printing an integer in Python

num = 5

print("number =", num)

output:-

number = 5

* **Floating Point numbers in Python**Floating point numbers (float) are real numbers with floating point representation, they are specified by a decimal point. They are represented by the float class.

Syntax:

Floating point numbers are of [float type](https://www.scaler.com/topics/float-in-python/). It is written as a number with a decimal point. Here's an example of Float data type:

# Printing a floating point number in Python

num = 5.55

print("number =", num)

output:

number = 5.55

* **Complex numbers in Python**Complex numbers in python are specified as (real part) + (imaginary part)j. They are represented by the complex class.

Syntax:

Complex numbers are of complex type. They are written in the form a+bj in which a is the real value and b is the imaginary value, where j represents the imaginary part.

Here's an example of a Complex data type:

# Printing a complex number in Python

num = 5 + 5j

print("number =", num)

Output:

number = (5+5j)

# Program to demonstrate numerical values in Python.

x = 10

y = 10.0

z = 10 + 10j

print("Data type of", x, ":", type(x))

print("Data type of", y, ":", type(y))

print("Data type of", z, ":", type(z))

In the above example, we are using the type() function to show the type of data. It can be seen that "10" is an int type, "10.0" is a float type, and "10+10j" is a complex type.

## **2. Sequence Data types in Python**

[Sequence data types](https://www.scaler.com/topics/sequence-data-type-in-python/) in Python are an ordered collection of similar or different values. They are also called container data types as they usually contain more than one value. We can access elements of a sequence data type by indexing. String, list, and tuple are the different types of containers used to store the data in a sequential manner.

* **Strings in Python**A string can be defined as a sequence of characters enclosed in single, double, or triple quotation marks. While in most cases single and double quotation marks are interchangeable. Triple quotation marks are used for multi-line strings. It is an immutable data type, i.e. its values cannot be updated.  
  String is represented by string class.

Syntax:

Strings are of string type, and are represented by enclosing in quotation marks.

Syntax:

Strings are of string type, and are represented by enclosing in quotation marks.

# Python program to demonstrate strings.

# String with single quotes

x = 'Scaler'

print("x =", x)

print("The data type of x:", type(x))

print()

# String with double quotes

y = "Scaler"

print("y =", y)

print("The data type of y:", type(y))

print()

# String with triple quotes (Multi-line strings)

z = '''Triple quotes are used

for multi-line strings'''

print("z =", z)

print("The data type of z:", type(z))

Output:

x = Scaler

The data type of x: <class 'str'>

y = Scaler

The data type of y: <class 'str'>

z = Triple quotes are used

for multi-line strings

The data type of z: <class 'str'>

In the above example, we are demonstrating Python string, using single, double, and triple quotation marks. While single and double quotation marks are used for normal strings, triple quotation marks are used for multi-line strings.

Accessing elements of a string

Any character of a string can be accessed using indexing. Python also allows us to use negative indexing to access characters even from the back of a string.

*Note: Indexing of a sequence starts from 0.*

# Python program to access characters of a string.

my\_string = "ScalerTopic"

print("String:", my\_string)

# Printing first character of the string.

print("First character:", my\_string[0])

# Printing last character of the string.

print("Last character:", my\_string[-1])

output:

String: ScalerTopic

First character: S

Last character: c

In the above example, we are accessing elements of the string using indexing. We are printing the first character of the string using the 0th index and the last character of the string using the -1th index (as Python supports negative indexing).

* **Lists in Python**Lists are an ordered sequence of one or more types of values. They are just like arrays in other languages. They are mutable, i.e. their items can be modified.

Syntax:

Lists are of the type list. They are created by enclosing items (separated by commas) inside square brackets [].

# Python program to demonstrate lists.

my\_list = [1, 10, "Scaler", 4, "A"]

print(my\_list)

print("Its data type is", type(my\_list))

output:

[1, 10, 'Scaler', 4, 'A']

Its data type is <class 'list'>

In the above example, we are creating a list by enclosing the elements inside square brackets[] and then printing it. In the second line, we are printing its type using the type() function.

Accessing elements of a list

Elements of a list can be accessed by referring to their index numbers, negative indexes to access the list items from its back.

# Python program to access elements of a list.

my\_list = [1, 10, 'Scaler', 4, 'A']

print("List:", my\_list)

# Accessing first element of the list.

print("First element of the list:", my\_list[0])

# Accessing 4th element of the list.

print("4th element of the list:", my\_list[3])

# Accessing last element of the list.

print("Last element of the list:", my\_list[-1])

List: [1, 10, 'Scaler', 4, 'A']

First element of the list: 1

4th element of the list: 4

Last element of the list: A

In the above example, we are accessing the elements of the list using indexing. At first, we are accessing the first element of the list by referring to its index "0" (Index number = Position of the element - 1) then we are referring to the fourth element by its index "3" and the last element of the list is accessed using negative indexing.

* **Tuples in Python**Like lists, Tuples are also an ordered sequence of one or more types of values except for they are immutable, i.e their values can't be modified. They are represented by the tuple class.

Syntax:

Tuples are of tuple type. They are created by a sequence of values separated by a comma with or without parentheses "()" for grouping of the sequence.

# Python program to demonstrate tuples.

my\_tuple = (1, 10, 'Scaler', 4, 'A')

print(my\_tuple)

print("Its data type is", type(my\_tuple))

Output:

(1, 10, 'Scaler', 4, 'A')

Its data type is <class 'tuple'>

In the above example, we are creating a tuple by enclosing the elements inside parentheses and then printing it. In the second line, we are printing its type using the type() function.

Accessing elements of a tuple

Elements of a tuple can be accessed by referring to their index numbers, negative indexes to access the tuple items from its back.

# Python program to access elements of a tuple.

my\_tuple = (1, 10, 'Scaler', 4, 'A')

print("Tuple:", my\_tuple)

# Accessing first element of the tuple.

print("First element of the tuple:", my\_tuple[0])

# Accessing 3rd element of the tuple.

print("3rd element of the tuple:", my\_tuple[2])

# Accessing last element of the tuple.

print("Last element of the tuple:", my\_tuple[-1])

Output:

Tuple: (1, 10, 'Scaler', 4, 'A')

First element of the tuple: 1

3rd element of the tuple: Scaler

Last element of the tuple: A

In the above example, we are accessing the elements of a tuple using indexing. At first, we are printing the first element using the 0th index, then we are printing its third element using the 2nd index, at last, we are printing its last element using negative indexing.

Check out this [article](https://www.scaler.com/topics/python/tuples-in-python/) to learn more about Tuples in Python.

## **3. Dictionaries in Python**

In Python, Dictionaries are an unordered collection of pair values (key: value). This helps in retrieving the data and makes the retrieval highly optimized, especially in cases of high volume data. Keys can't be repeated in a dictionary while values can be repeated. Dictionaries are mutable, i.e. their values can be modified.

Syntax:

Dictionaries are of the type dict. The elements of a dictionary are enclosed in curly brackets {}, where each element is separated by a comma , and key-value pair by a colon :.

# Python program for demonstrating dictionaries.

my\_dict = {"Name": "Tom", "Age": 50, "Movie": "Mission Impossible"}

print(my\_dict)

print("Its data type:", type(my\_dict))

Output:

{'Name': 'Tom', 'Age': 50, 'Movie': 'Mission Impossible'}

Its data type: <class 'dict'>

In the above example, we are creating a dictionary by enclosing the key-value pair in curly brackets {} then we are printing it. In the second line, we are printing its type using the type() function.

Accessing values of a dictionary

Values of a dictionary are accessed by referring to their keys.

# Python program for accessing values of a dictionary.

my\_dict = {'Name': 'Tom', 'Age': 50, 'Movie': 'Mission Impossible'}

print(my\_dict["Name"])

print(my\_dict["Age"])

print(my\_dict["Movie"])

Output:

Tom

50

Mission Impossible

In the above example, we are accessing the values of a dictionary by referring to their keys, in the first line we are referring to Tom using the keyName, then we are referring to "50" using the keyAge, at last, we are referring to "Mission Impossible" using the keyMovie.

## **4.Booleans in Python**

Boolean is a data type that has one of two possible values, True or False. They are mostly used in creating the control flow of a program using conditional statements.

Syntax:

The True value in the boolean context is called "truthy" and False value is called "falsy".

# Python program to demonstrate boolean.

a = True

b = False

# Printing boolean values

print("a =", a)

print("b =", b)

print()

# Data type of True and False

print("Data type of 'True':", type(a))

print("Data type of 'False':", type(b))

Output:

a = True

b = False

Data type of 'True': <class 'bool'>

Data type of 'False': <class 'bool'>

n the above example, we are printing the boolean variables a (True) and b (False), then we are printing their types using the type() function

## **5. Sets in Python**

[Sets in Python](https://www.scaler.com/topics/set-in-python/) are an unordered collection of elements. They contain only unique elements. They are mutable, i.e. their values can be modified.

Syntax:

Sets are of the type set. They are created by elements separated by commas enclosed in curly brackets {}.

# Python program for demonstrating sets.

my\_set = {1, 8, "Scaler", "F", 0, 8}

print(my\_set)

print("Its data type:", type(my\_set))

Output:

{0, 1, 'F', 8, 'Scaler'}

Its data type: <class 'set'>

In the above example, we are creating a set by enclosing the elements inside curly brackets "{}" then we are printing it. In the second line, we are printing its type using the type() function. While creating the set, we have put two duplicate values 8 but when we printed it, 8 occurred only once, that's because sets only have unique elements.

## **Conclusion**

* Data types are classes in Python, whereas variables are objects of these classes.
* Each and every value or variable has a data type in Python.
* Numeric data types in python represent data with numeric values.
* Sequence data types in python are an ordered collection of items.
* Dictionary (also called maps, in other languages) helps in the optimized retrieval of values.
* Boolean has two types of values, True or False, True for truthy values, False for falsy values.
* Sets are an unordered collection of unique elements (like set theory in mathematics).