

Data Type

Datatype represents the type of data stored into a variable or memory.

Type of Data type :-

- Built-in Data type
- User Defined Data type

Built-in Datatype

These datatypes are provided by Python Language.

Following are the built-in data type:-

- None Type
- Numeric Types
- Sequences
- Sets
- Mappings

User Defined Data type

- Array
- Class
- Module

None Type

None datatype represents an object that doesn't contain any value.

Numeric Type / Number

Following are the Numeric Data type:-

Int

Float

Complex

Numeric Type / Number

Int – The int datatype represents an integer number. An integer number without any decimal point or fraction part. In Python, It is possible to store very large integer number as there is no limit for the size of an int datatype.

Ex:-

20, 10, -50, -1002

y = 10

pin_code = 564512

int type variable



```
graph TD; A[int type variable] -- blue arrow --> B[pin_code = 564512]; C[int value] -- purple arrow --> B;
```

pin_code = 564512

int value

Numeric Type / Number

Float – The float data type represents floating point numbers. A floating point number is a number that contains a decimal point.

Ex:-

25.56, 10.5, -45.69, -0.8

price = 25.56

run_rate = -0.8

value = 5.1e5

5.1×10^5

float type variable

run_rate = -0.8

float value

5.1e5

It's scientific notation where e or E represents exponentiation which represents the power of 10

Numeric Type / Number

Complex – A complex number is a number that is written in the form of $a + bj$ or $a + bJ$. Where,

a = Real Part of the number

b = Imaginary part of the number

j or J = Square root value of -1

a and b may contain integer or float number.

Ex:- $5+7j$, $0.8+2j$

$com = 5+7j$

complex type variable



$com = 5+7j$

complex number



Bool type

The bool datatype represents boolean value True or False. Python internally represents True as 1 and False as 0.

Ex:- True, False

True + True = 2

True – False = 1

Sequence Type

Following are sequence type:-

String

List

Tuple

Range

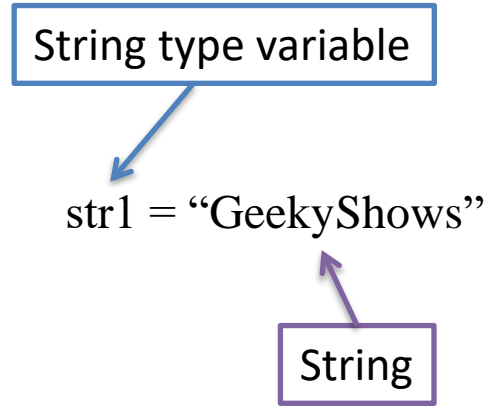
Sequence Type

String – String represents group of characters. Strings are enclosed in double quotes or single quotes.

Ex:- “Hello”, “GeekyShows”, ‘Rahul’

str1 = “GeekyShows”

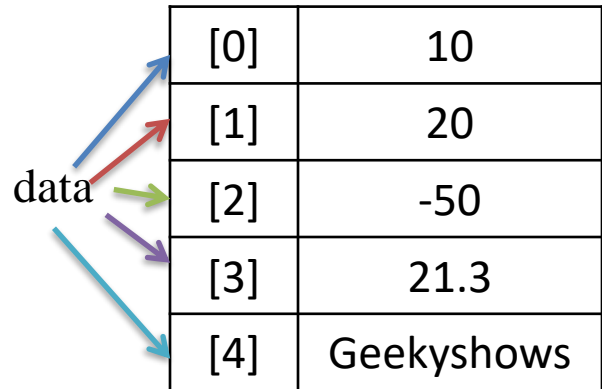
str1 = ‘GeekyShows’



Sequence Type

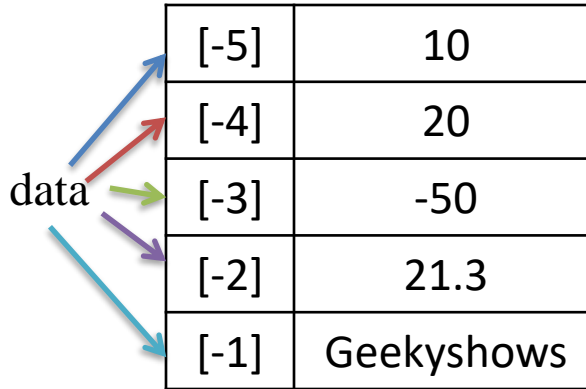
List – A list represents a group of elements. A list can store different types of elements which can be modified. Lists are dynamic which means size is not fixed. Lists are represented using square bracket [].

Ex:- `data = [10, 20, -50, 21.3, 'Geekyshows']`



A diagram illustrating list indexing. On the left, the word 'data' is written. Five colored arrows point from 'data' to the index column of a table: a blue arrow to [0], a red arrow to [1], a green arrow to [2], a purple arrow to [3], and a cyan arrow to [4].

[0]	10
[1]	20
[2]	-50
[3]	21.3
[4]	Geekyshows



A diagram illustrating list indexing. On the left, the word 'data' is written. Five colored arrows point from 'data' to the index column of a table: a blue arrow to [-5], a red arrow to [-4], a green arrow to [-3], a purple arrow to [-2], and a cyan arrow to [-1].

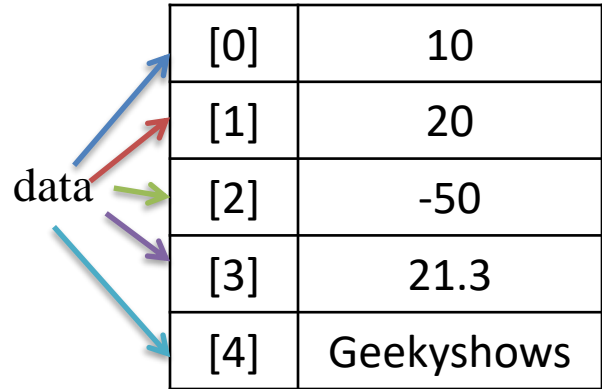
[-5]	10
[-4]	20
[-3]	-50
[-2]	21.3
[-1]	Geekyshows

`data[1] = 40`

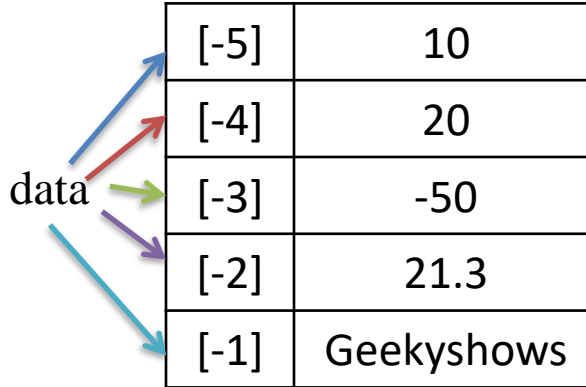
Sequence Type

Tuple – A tuple contains a group of elements which can be different types. It is similar to List but Tuples are read-only which means we can not modify its element. Tuples are represented using parentheses ().

Ex:- data = (10, 20, -50, 21.3, 'Geekyshows')



[0]	10
[1]	20
[2]	-50
[3]	21.3
[4]	Geekyshows



[-5]	10
[-4]	20
[-3]	-50
[-2]	21.3
[-1]	Geekyshows

data[1] = 40

Sequence Type

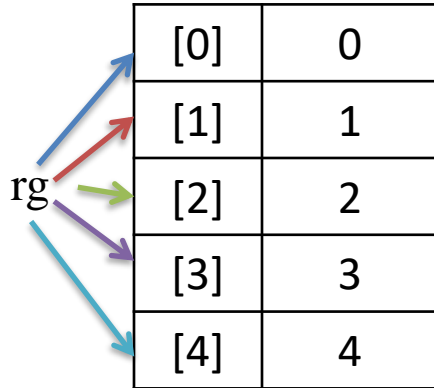
Range – Range represents a sequence of numbers. The numbers in the range are not modifiable.

Ex:- `rg = range(5)`

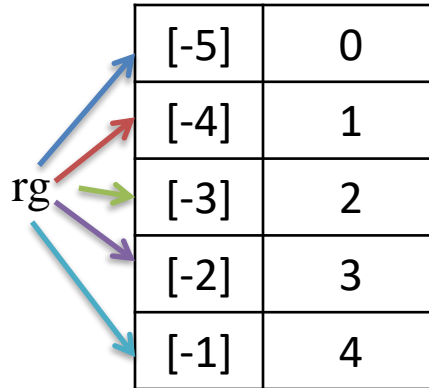
`rg = range(10, 20, 2)`

0 1 2 3 4

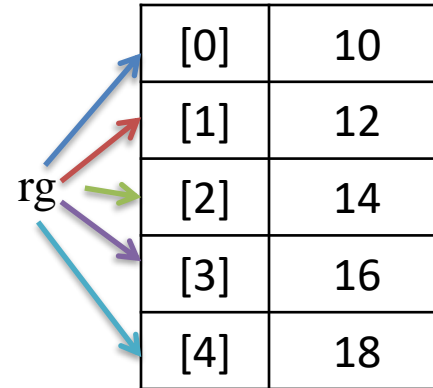
10 12 14 16 18



[0]	0
[1]	1
[2]	2
[3]	3
[4]	4



[-5]	0
[-4]	1
[-3]	2
[-2]	3
[-1]	4



[0]	10
[1]	12
[2]	14
[3]	16
[4]	18

Set Type

A set is an unordered collection of elements much like a set in mathematics.

The order of elements is not maintained in the sets. It means the elements may not appear in the same order as they are entered into the set.

A set does not accept duplicate elements.

Sets are unordered so we can not access its element using index.

`data[0] = 10`

Sets are represented using curly brackets { }.

Ex:-

`data = {10, 20, 30, "GeekyShows", "Raj", 40}`

`data = {10, 20, 30, "GeekyShows", "Raj", 40, 10, 20}`

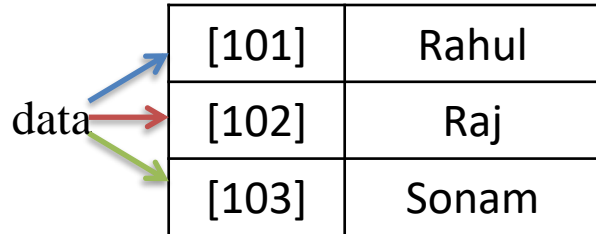
Mapping Type/ dict / Dictionary

A map represents a group of elements in the form of key value pairs.

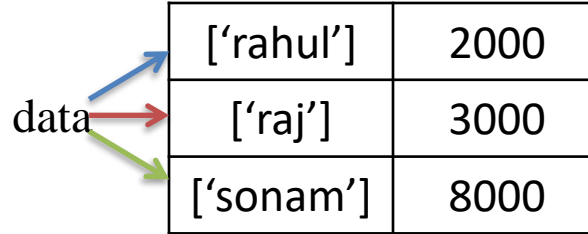
Ex:-

```
data = {101: 'Rahul', 102: 'Raj', 103: 'Sonam' }
```

```
data = {'rahul':2000, 'raj':3000, 'sonam':8000, }
```



[101]	Rahul
[102]	Raj
[103]	Sonam



['rahul']	2000
['raj']	3000
['sonam']	8000

Character

There is no concept of char data type in Python to represent individual character.