Python Programming

(ESTD-1995)

RGM College of Engineering & Technology (Autonomous)

Department of Computer Science & Engineering

Academic Year : 2020-2021

PYTHON LANGUAGE FUNDAMENTALS-6

Guido Van Rossum

Learning Mantra

If you really strong in the basics, then remaining things will become so easy.

Agenda:

1. Data types: complex data type
2. Data types: bool data type
3. Data types: str data type
4. Data types: str data type - positive and negative index

Data types: complex data type

Now, we'll discuss about Python specific special data type known as Complex data type.

Why Python having this special data type?

- If you want to develop scientific applications, mathematics based applications and Electrical engineering applications, this complex type is very helpful.

How can we represent a complex number?

a + bj is the syntax for representing a complex number.

- Here, **a** is called real part and **b** is called imaginary part.

you may get one doubt that in the complex number representation, is it compulsory **j**?

In mathematics we seen **i** instead of **j**.

- It is mandatory, it should be **j** only in Python.

Key Points:

- In the real part if we use int value then we can specify that either by decimal, octal, binary or hexa decimal form.
- imaginary part must be specified only by using decimal form.
- Assume that, we have two complex numbers. Can we perform arithmetic operations between these two complex numbers?
    - Yes, we can perform without any difficulty.

Note :

- This is about basic introduction about complex data type.
- It is not that much frequently used data type in Python.
- It is very specific to Scientific, Mathematical and Electrical Engineering Applications.

Data types: bool data type

- We can use this data type to represent boolean values.
- The only allowed values for this data type are: **True** and **False** (true & false are not allowed in Python)
- Internally Python represents True as 1 and False as 0

Data types: str data type representations by using single, double and triple quotes

- **str** represents String data type.
- It is the most commonly used data type in Python.

String:

- A String is a sequence of characters enclosed within single quotes or double quotes.
- In Python to represent a string, can we use a pair of single quotes ('') or double quotes ("")?
  - The answer is, We can use either single quotes or double quotes.

s = 'Karthi'

print(type(s))

<class 'str'>

s = 'a'

print(type(s)) # in Python there is no 'char‘ data type

<class 'str'>

s = "a"

print(type(s))

<class 'str'>

s = 'a'

print(s) # value of 's‘ ➔ **a**

print(type(s)) #type of 's‘ ➔ **<class 'str'>**

In Python, we can use triple quotes also in the following 3 situations.

1.By using single quotes or double quotes we cannot represent multi line string literals.

For example,

```
s = "Karthi
sahasra"
```

For this requirement we should go for triple single quotes(''') or triple double quotes(""").

2.We can also use triple quotes, to use single quotes or double quotes as normal characters in our String.

3.To define doc string, triple quotations will be used. (We will discuss this later)

Data types: str data type - positive and negative index

- One speciality is there in Python indexing, which is not available in C or Java.
- The characters of the string is accessed by using it's relative position in the string, that is called as index.
- In Python, indexing starts from 0.

Eg:

s = "karthi"

print(s[0])   # The character location at 0 index is displayed ➔k

print(s[3])   ➔ t

print(s[100])   ➔ **IndexError:** string index out of range

Data types: str data type - positive and negative index

Up to this is similar in C or Java like languages. Now we will see what is the speciality regarding indexing in Python.

- **Python supports both positive indexing and negative indexing.**
- As we are already discussed, **positive indexing moves in forward direction of string and starts from 0.**
- **Negative indexing moves in reverse direction of string and starts from -1.**

Data types: str data type - positive and negative index

Eg:

```
s = "karthi”
print(s[-1])        ➔i
print(s[-6])        ➔k
print(s[-7])        ➔IndexError: string index out of range
```

Any question?

If you try to practice programs yourself, then you will learn many things automatically

Spend few minutes and then enjoy the study

Thank You