INTRODUCTION TO BASIC DATA TYPES IN PYTHON

Variables

- Variables can be considered as Containers of Data.
- In technical terms, a Python Variable is a Reserved Memory Location that stores a Value. Typically, a Variable is a Pointer of some Location in Memory.
- Variables in Python are not “Statically Typed”. There is no need to declare Variables before using those. A Variable is created the moment a value is assigned to it.
- There is also no need to declare the Data Type of the Variable to create in Python. Example -

```
# Integer Variable
number = 6789
print(number)

number = 7890
print(number)
```

```
# String Variable
name = 'Oindrila'
print(name)

name = "Soumyajyoti"
print(name)
```

```
# Floating-Point Variable
salary = 19005.87
print(salary)

salary = 34864.82
print(salary)
```

Output -

```
6789
7890
```

```
Oindrila
Soumyajyoti
```

```
19005.87
34864.82
```

- Variable of different Data Types cannot be concatenated, like an Integer Variable cannot be concatenated with a String Variable.

```
weight = 67
sentence = "My weight is " + weight + " Kgs."
print(sentence)
```

Output -

```
Traceback (most recent call last):
  File "D:\Tutorials\Python Tutorial\Python_Project_Folder\VariablesFile.py", line 26, in <module>
    sentence = "My weight is " + weight + " Kgs."
TypeError: can only concatenate str (not "int") to str
```

To concatenate Variable of different Data Types, the Variables have to be Explicitly Type-casted to the intended Data Type of the Expression, involving with the Variables.

```
weight = 67
sentence = "My weight is " + str(weight) + " Kgs."
print(sentence)
```

Output -

```
My weight is 67 Kgs.
```

Basic Data Types in Python

- ***Integer*** - In Python 3, there is effectively no limit to how long an Integer Value can be, i.e., the Value of an Integer can only be constrained by the amount of Memory the System has.
  There can be the following four types of Integer Values in Python -
  - ***Decimal Number*** - Python interprets a sequence of Decimal Digits without any Prefix to be a Decimal Number.

```
decInt = 78
data_type = type(decInt)
print(data_type)
```

Output -

```
<class 'int'>
```

- ***Binary Number*** - When the string, “0” appended to lowercase letter “b” or uppercase letter “B”, i.e., “0b” or “0B”, is prepended to an Integer Value, it indicates an Integer Value of Base “2”, i.e., a Binary Number.

```
binInt = 0b1001
data_type = type(binInt)
print(data_type)

binInt = 0B101011
data_type = type(binInt)
print(data_type)
```

Output -

```
<class 'int'>
<class 'int'>
```

- ***Octal Number*** - When the string, “0” appended to lowercase letter “o” or uppercase letter “0”, i.e., “0o” or “0O”, is prepended to an Integer Value, it indicates an Integer Value of Base “8”, i.e., an Octal Number.

```
octInt = 0o1060
data_type = type(octInt)
print(data_type)

octInt = 0O1254
data_type = type(octInt)
print(data_type)
```

Output -

```
<class 'int'>
<class 'int'>
```

- ***Hexadecimal Number*** - When the string, “0” appended to lowercase letter “x” or uppercase letter “X”, i.e., “0x” or “0X”, is prepended to an Integer Value, it indicates an Integer Value of Base “16”, i.e., a Hexadecimal Number.

```
hexaInt = 0x109F
data_type = type(hexaInt)
print(data_type)

hexaInt = 0X234DE
data_type = type(hexaInt)
print(data_type)
```

Output -

```
<class 'int'>
<class 'int'>
```

The underlying Type of a Python Integer, irrespective of the Base used to specify it, is called “int:”.

- ***Floating-Point Number*** - The “float” Type in Python designates a Floating-Point Number. “float” Values are specified with a Decimal point. Optionally, the characters “e”, or, “E”, followed by a positive or negative Integer may be appended to specify Scientific Notation.

```
floatVal = 67.098
data_type = type(floatVal)
print(data_type)

floatVal = 45.
data_type = type(floatVal)
print(data_type)

floatVal = .115
data_type = type(floatVal)
print(data_type)
```

```
floatVal = .64e3
data_type = type(floatVal)
print(data_type)

floatVal = .21E78
data_type = type(floatVal)
print(data_type)

floatVal = 33.6E-8
data_type = type(floatVal)
print(data_type)
```

Output -

```
<class 'float'>
<class 'float'>
<class 'float'>
<class 'float'>
<class 'float'>
<class 'float'>
```

- ***Complex Number*** - The "complex" Type in Python designates a Complex Number. "complex" Values are specified as "<Real Part>+<Imaginary Part>j".

```
complexVal = 6+3j
data_type = type(complexVal)
print(data_type)

complexVal = 52-81j
data_type = type(complexVal)
print(data_type)

complexVal = -49+12j
data_type = type(complexVal)
print(data_type)
```

Output -

```
<class 'complex'>
<class 'complex'>
<class 'complex'>
```

- ***Boolean*** - The “bool” Type in Python designates a Boolean Data Type. Objects of Boolean Data Type may have one of the two Values - “True”, or, “False”.

```
isValid = True
data_type = type(isValid)
print(data_type)

isValid = False
data_type = type(isValid)
print(data_type)
```

Output -

```
<class 'bool'>
<class 'bool'>
```

- ***String*** **-** Strings are Sequences of Character data. The String Type in Python is called “str”. String Literals may be delimited using either Single, or, Double Quotes. All the Characters between the opening delimiter and matching closing delimiter are part of that String.

```
name = 'My name is Oindrila Chakraborty Bagchi'
data_type = type(name)
print(data_type)

name = "My name is Oindrila Chakraborty Bagchi"
data_type = type(name)
print(data_type)
```

Output -

```
<class 'str'>
<class 'str'>
```

A String in Python can contain as many characters as needed. The only limit is the System’s Memory resources. A String can also be Empty.

```
name = ''
print(name)
```

Output -

```

```

Can One Variable be Assigned Multiple Times with Different Data Types

- Yes. A Variable can be assigned with Values of different Data Types successively in Python. This is known as “Dynamic Typing” in Python. The same Variable can be used to store different types of data.

```
isAdult = True
data_type = type(isAdult)
print(data_type)

isAdult = 67
data_type = type(isAdult)
print(data_type)

isAdult = "Not adult"
data_type = type(isAdult)
print(data_type)
```

Output -

```
<class 'bool'>
<class 'int'>
<class 'str'>
```