Built-in Data Types in Python

Vivek K. S., Deepak G.

Information Systems Decision Sciences (ISDS)
MUMA College of Business
University of South Florida
Tampa, Florida

2017

Objectives

- To understand built-in data types such as integers, floats, strings and Boolean type.
- To learn their usage and specific behavior.
- To understand objects in Python.
- To understand dynamic typing in Python type conversion.
- To understand variables and references.

Objects in Python

- In Python, every data structure is an object.
- Integers, Strings, Lists, Functions, Modules are all implemented as objects.
- An object is in its simple definition a block(s) of memory that contains data in it.
- The data has a type associated with it and consequentially a set of behaviors and what could be done on/with it.
- The type of data also determines its mutability and immutability.
- The advantage of this implementation is consistency.

Dynamic Typing in Python

- Python is a dynamically typed language.
- Unlike statically-typed languages like Java or C++, Python does not attach the type of an object to its variable identifier.
- Instead, the assignment of an object to an identifier simply attaches a name to the block of memory containing the data and acts only a reference to it.
- In python, we use the type() method to identify the type of the variable.

```
# Code to identify data type
a = 10
print(a)
type(a)
```

Rules for Identifier Names

The following are some of the rules associated with identifier names in Python. The following characters are allowed in identifier names.

- Lowercase letters (a through z)
- Uppercase letters (A through Z)
- Digits (0 through 9)
- Underscore (_)

Their usage is as follows.

- Names cannot begin with a digit as seen in most programming languages.
- Python treats names that begin with an underscore in special ways.
- Reserved keywords in Python cannot be used for identifier names.

Unique Operations in Python

Python offers some unique flavors in the most common operations we use on a daily basis:
Python offer two types of division operation.

```
/ carries out decimal division.
7/2 = 3.5.
// carries out integer division also called
as floor division.
7//2 = 3

x = 77
x //= 10
x
[Output] 7
```

Assignment and Operation in Python

Python offers some unique flavors in the most common operations we use on a daily basis:
Python offer two types of division operation.

```
Adding two integers (numbers) could be done as simp
a = 95
a -= 3
a
[Output] 92
a *= 2
a
[Output] 184
```

Other Operations in Python

Modulo operation 9%5 = 4

Getting the reminder & quotient can be done by using **divmod**()

divmod(9,5) = (1,4)

Type Conversions in Python

Python offers way to convert one type into another:

```
Converting a Boolean to int

>>>int(True)
1
>>>int(False)
0

Converting a Float to int

>>>int(55.5)
55
>>>int(1.0e4)
10000
```

Continued..

Python offers way to convert one type into another:

```
Converting a String to int
>>>int('99')
99
>>>int('-55')
-55

Converting an int to char
>>>chr(97)
'a'
Getting the ASCII code of a character.
>>>ord('a')
97

ord() and chr() are built-in functions in Python.

footnote - Refer https://docs.python.org/3/library/
```

Integer Overflow

- Python handles really long numbers with ease.
- This is a feature which most Programming languages have a problem with commonly referred to a s "Integer overflow".
- For example 10**100 (10 raised to the power 100) will result in a huge number called the googol.
- Even if an multiplication is to be performed between two googols, Python can easily handle that math and support the operation.
- Lets try it.

Floats in Python

Floats are numbers with decimal points.

- All the operations that work on integers can be applied to floats as well.
- Float type conversion can be done using float()
- Strings can be converted to floats as well.

Strings in Python

A String is a sequence type in Python. It is basically a string of characters.

- In Python, Strings are immutable.
- The data in a String cannot be modified, but copies can be created.
- Strings in Python 3 support unicode operations.
- In Python, Strings can be written using both single and double quotes, which allows double quotes to be written within single quotes and vice versa.

```
>>>'Python'
Python
>>>"Python"
Python
```

Triple Quotes and Docstrings

Triple Quotes is a unique flavor offered by Python.

- They are most commonly used to create multi-line Strings.
- One of their most common uses is in creating docstrings.
- A docstring is a string literal specified in a function or any piece of code as a comment, to document that specific segment of code.

```
def add(x,y):
        '''The function add two integers
        and returns the sum.'''
```

Continued..

There are two ways to see the docstring.

```
def add(x,y):
        '''The function add two integers
       and returns the sum.'''

help(add)
Help on function add in module __main__:

add(x, y)
The function adds two integers
and returns the sum.
```

```
    add.__doc__
    'The␣function␣adds␣two␣integers␣\n␣and
␣␣␣␣␣returns␣the␣sum.'
```

Common String Operations

The following are some of the common operations in a String.

- String Concatenation.
- Duplication.
- Replacing a substring.
- Indexing and Slicing.
- Finding the length.
- Splitting and Stripping.
- String formatting operations.

Summary

- We understood Build-in data types in Python.
- We learned how Python works as a Dynamically typed language.
- We understood the use of integers, strings and floats and the operation that could be performed on them.
- We learned how these data types act as a building block towards building larger complex program structures.