2. Basics of Python

Table of Contents

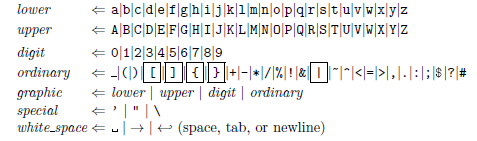
|  |  |  |
| --- | --- | --- |
| Sl. No | Topics | Page No |
| 2.1 | **Python Basics** | **2** |
| 2.2 | **Basic Syntax** | **5** |
| 2.3 | **Variable** | **7** |
| 2.4 | **Datatypes** | **8** |
| 2.5 | **Operators** | **14** |
| 2.6 | **Lab – Basics of Python** | **18** |

## 2.1 Python Basics

**What is Python Character Set?**

Python program contains words or statements which follow a sequence of characters. When these characters are submitted to the python interpreter, they are interpreted or uniquely identified in various contexts, such as Characters, Identifiers, names or constants. Python uses the following character set:-

* **Letters:** Upper case and Lower case letters
* **Digits**: 0,1,2,3,4,5,6,7,8,9
* **Special Symbols:** (\_), (,), [,] {,} ,+, - ,\* ,&, %, $ ,# ,! ,(‘), (“), (\), (:), (;)
* **White Spaces:**  (‘\t \n \x0b \x0c \r’ ), Space, Tab

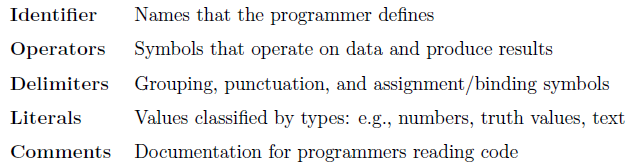


**What is a Token?**

A program in Python contains a sequence of instructions. Python breaks each statement into a sequence of lexical components known as Tokens. Each token corresponds to a substring of a statement.

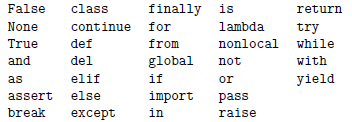
Python contains various types of tokens listed below:-

* **Keywords**
* **Identifiers/Variables**
* **Operators**
* **Delimiters**
* **Literals**



**Keywords**

* Keywords are special identifiers that have predefined meanings that cannot change in Python.
* Keywords cannot be used as identifiers or variables.



**Identifier**

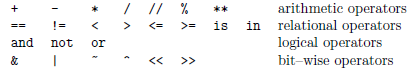
* A Python Identifier is a name used to identify a variable, function, class, module or other object
* An identifier starts with a letter A to Z or a to z or an underscore (\_) followed by zero or more letters, underscores and digits (0 to 9)
* Python is a case sensitive programming language
* Python does not allow special characters such as @, $ and % within identifiers

**Identifiers – Naming Conventions**

* Class names start with an uppercase letter. All other identifiers start with a lowercase letter
* Starting an identifier with a single leading underscore indicates that the identifier is private
* Starting an identifier with two leading underscores indicates a strongly private identifier
* If the identifier also ends with two trailing underscores, the identifier is a language-defined special name

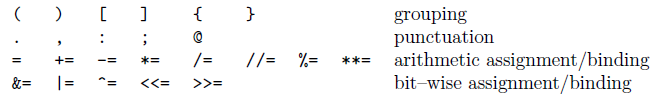
**Operators**

* Operators compute a result based on the value of their operands e.g., Addition Operator



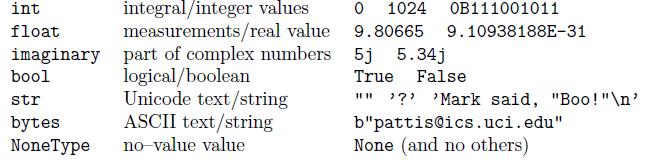
**Delimiters**

* Delimiters are either grouping symbols, punctuation symbols, or symbols that assign/bind object to names.



**Literals**

* Computers store all information digitally, as bits; the type of information determines how these bits are interpreted.
* We write values as literals in Python; each literal belongs to exactly one of Python's built in types.



**What are Command Line arguments in Python?**

* It is possible to pass arguments to Python programs when they are executed
* The brackets which follow main are used for this purpose:-
* argv refers to the number of arguments passed, and argv[] is a pointer array which points to each argument which is passed to main
* The Python sys module provides access to any command-line arguments via the sys.argv.
* This serves two purposes:

- sys.argv is the list of command-line arguments

- len(sys.argv) is the number of command-line arguments

import sys

print('Number of Args:',

len(sys.argv),'arguments.')

print ('Argument List:', str(sys.argv))

**Output**

Number of Args: 3 arguments.

Argument List: ['C:\\Users\\VIJAY\\Anaconda3\\lib\\site-packages\\ipykernel\_launcher.py', '-f', 'C:\\Users\\VIJAY\\AppData\\Roaming\\jupyter\\runtime\\kernel-7366686e-ac71-4b06-8ed8-be79a0e88595.json']

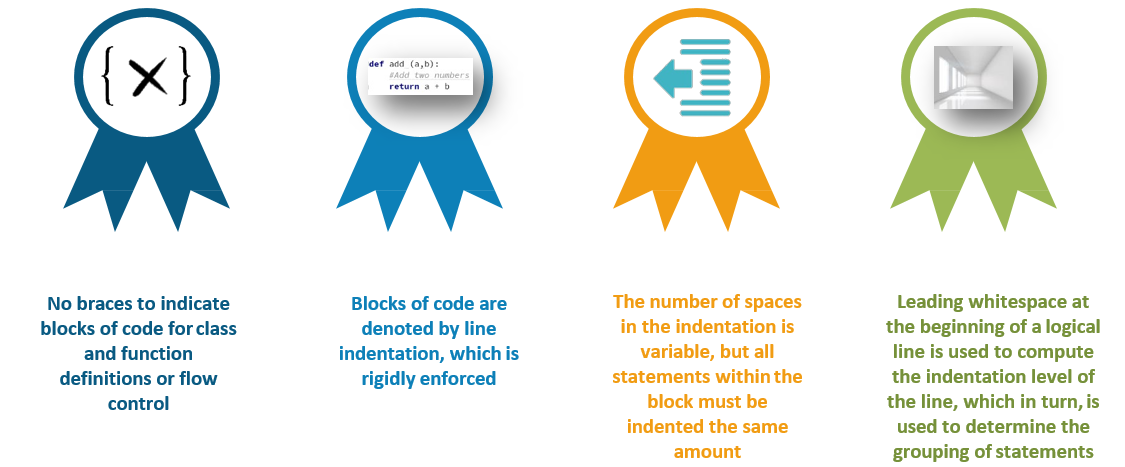
## 2.2 Basic Syntax

**Comments and Literals  
Comments:** Any text to the right of the # symbol is mainly used as notes for the readers. Statements on right side of # does not get executed. It gives more information about function

**Bulk Comments:** Enclose the code in triple quoted strings (“””)

**Literal Constants:** Any number or character, or set of characters

**Indentation**



**print() function**

Every character in Python should be enclosed within single or double quotes

->Output after running new.py

print(**‘Hello World’**) print(**"Welcome to Python"**)

Hello World

Welcome to Python

**input() function**

* The input() function is used to accept an input from a user.
* A programmer can ask a user to input a value by making use of input()

**Syntax:** Variable\_Name = input or Variable\_Name = input (‘String’)

Enter String1: Hello

Enter String2: Welcome to Python

String1= Hello

String2= Welcome to Python

Str1 = input(**‘Enter String1:’**)

Str2 = input(‘**Enter String2:’**)

print(**'string1 = ‘, str1**)

print(**'string2 = ‘, str2**)

**eval() function**

* The eval() function is used to evaluate.
* Takes a string as parameter and returns it as if it Python expression.

**Example:** eval(‘print(“Hello”)’) -> Hello

* We know that input() function returns every input by the user as string, including numbers and this problem was solved by making use of type before input() function. **Example:** X=eval(input(‘Enter Number’))

Enter Name: Sriram

Enter Age: 100

Name: = Sriram

Age: = 100

Name = input(**‘Enter Name :’**)

Age = eval(input(‘**Enter Age :’**)

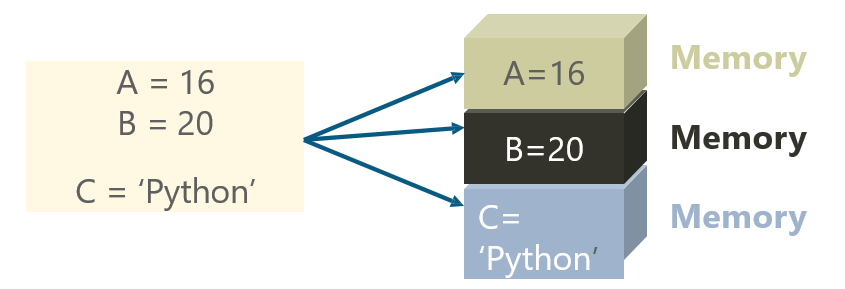
print(**‘Name: = ‘, Name**)

print(**‘Age: = ‘, Age**)

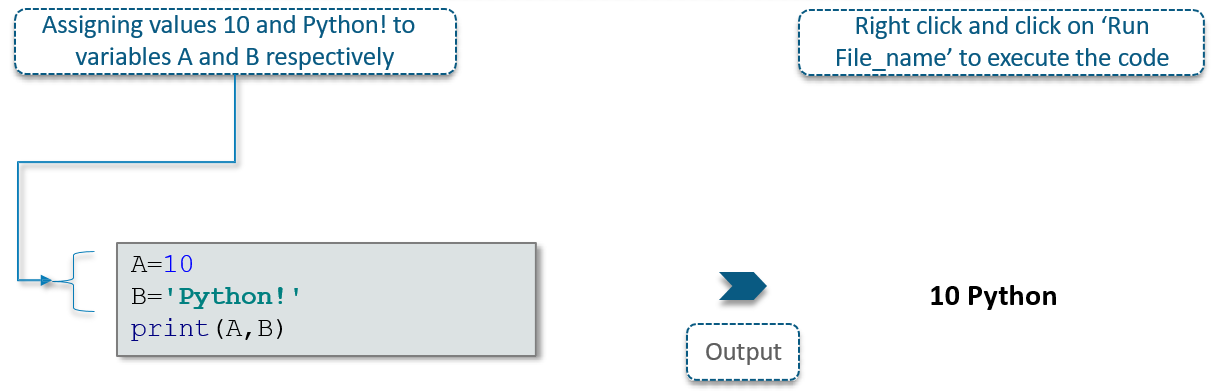
## 2.3 Variables

**What is a Variable? Give an example?**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

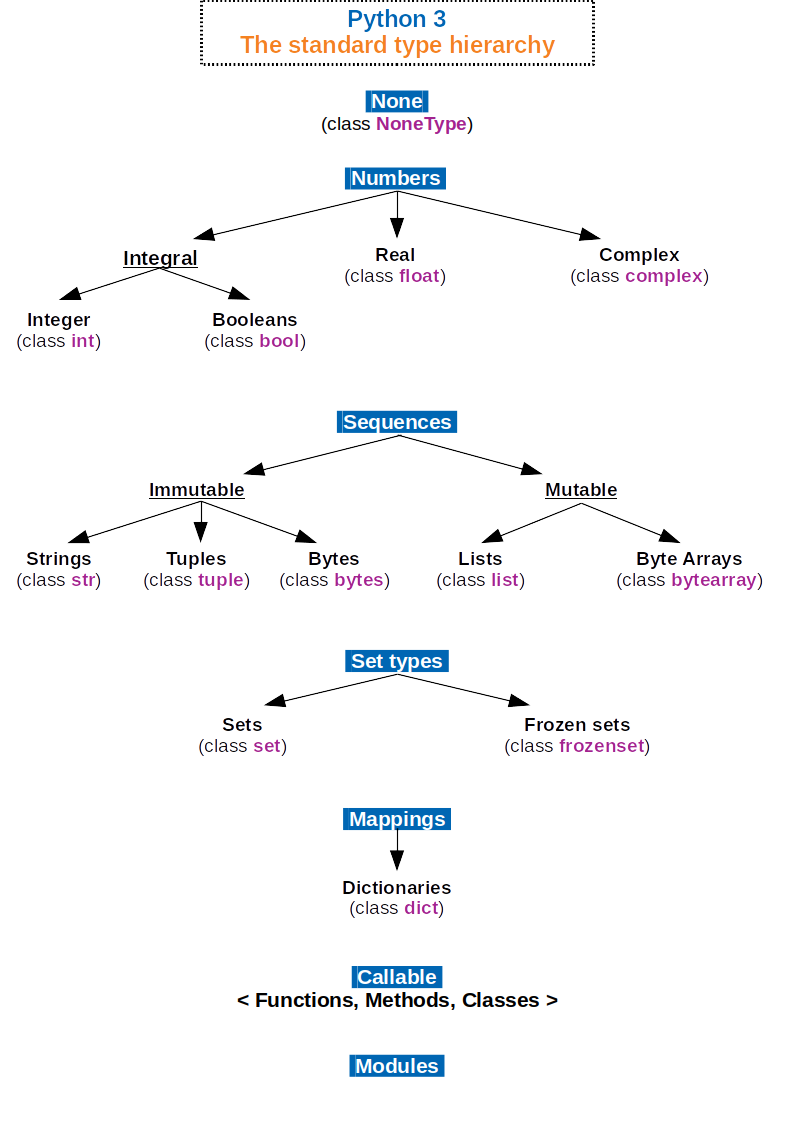


**Example**

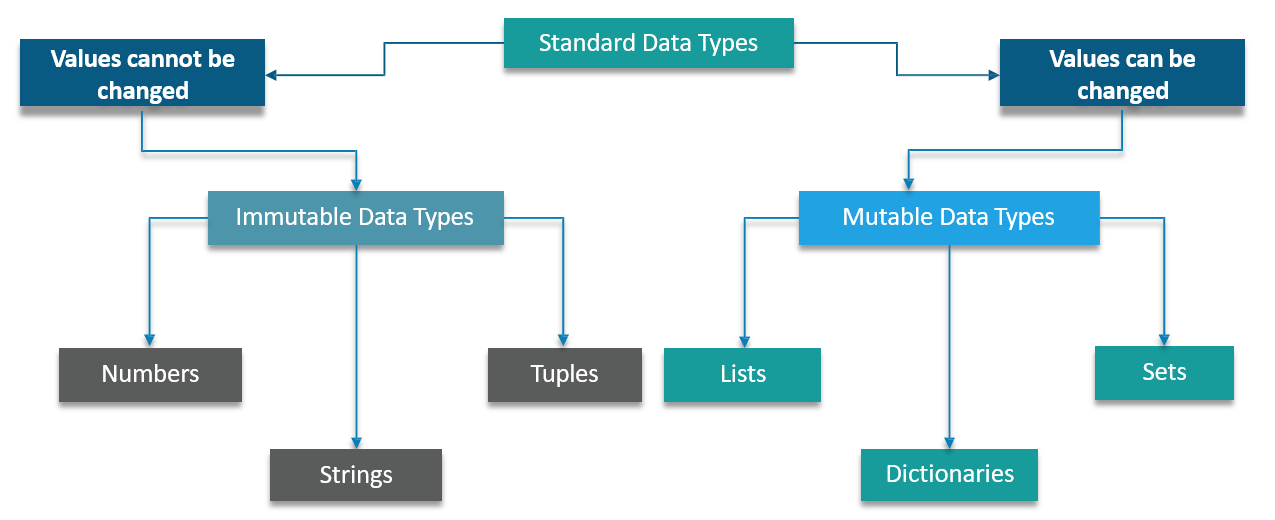


## 2.4 Datatypes

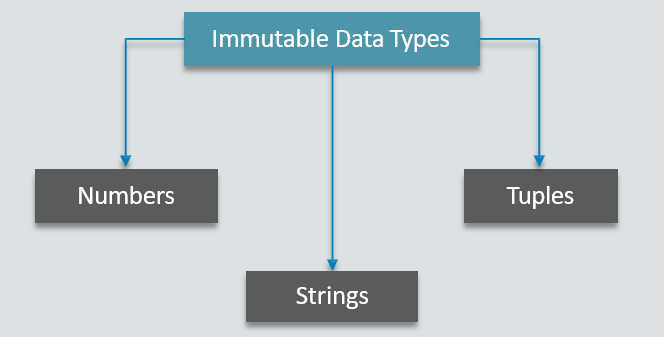
**What are the Python Standard Datatypes?**



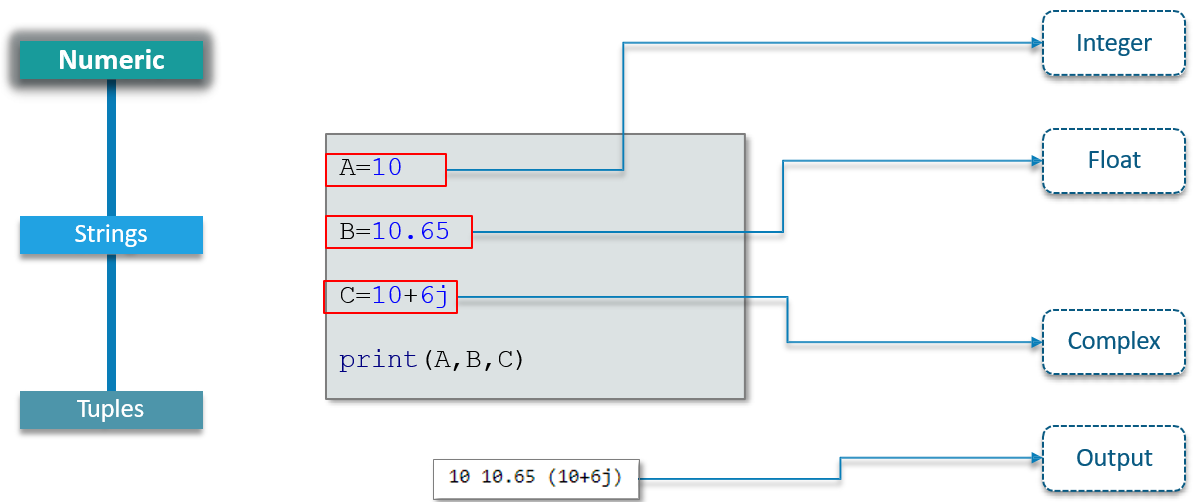
**Standard Data Types**



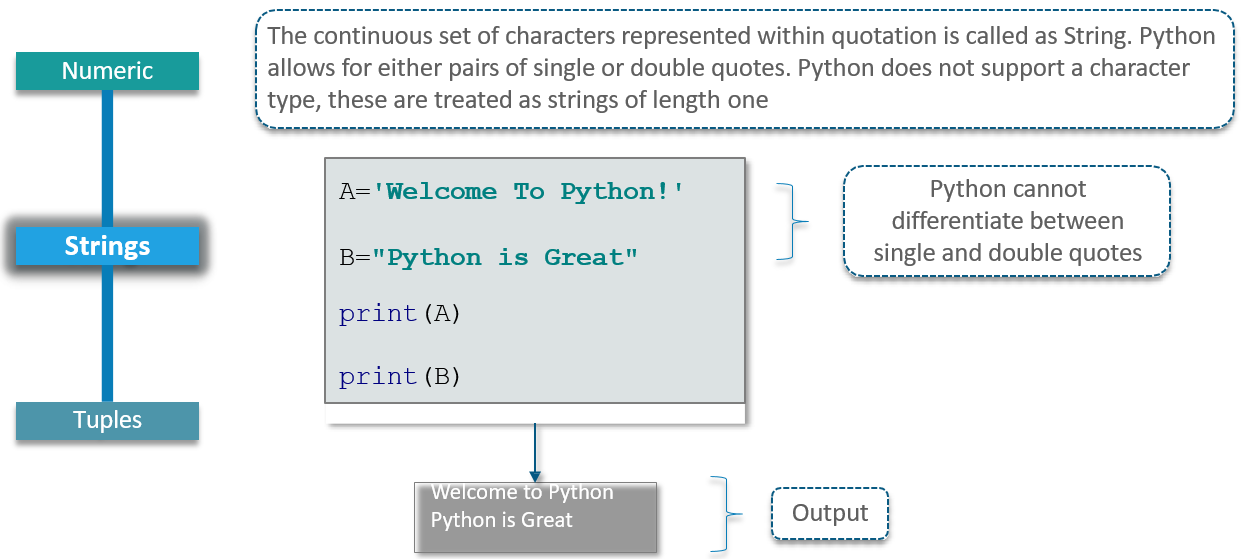
**Immutable Data Types**



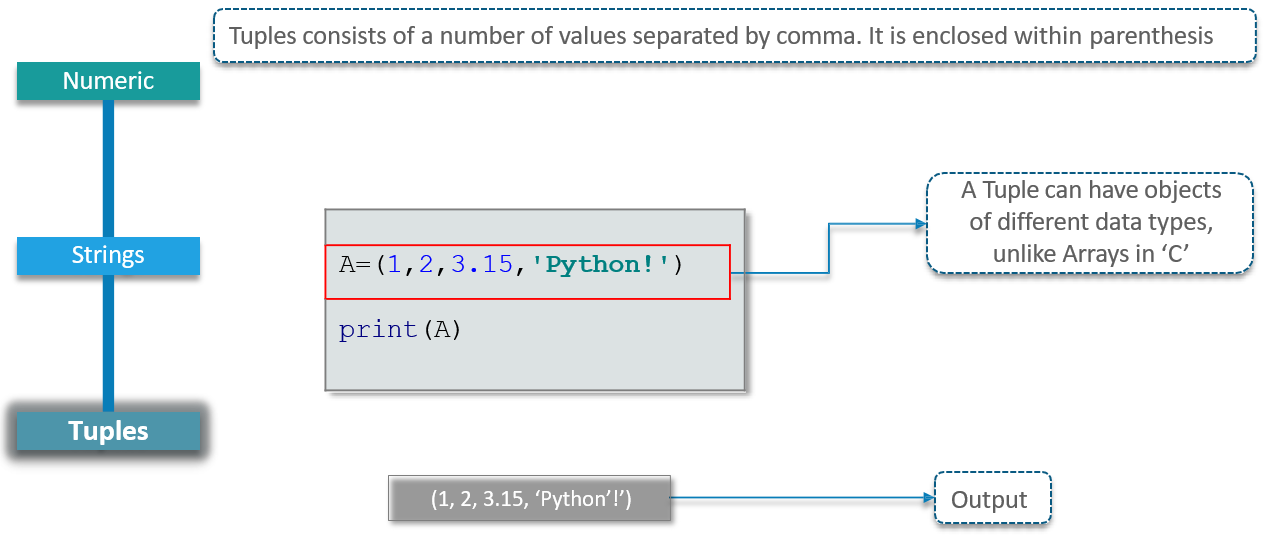
**Numeric Data Type**



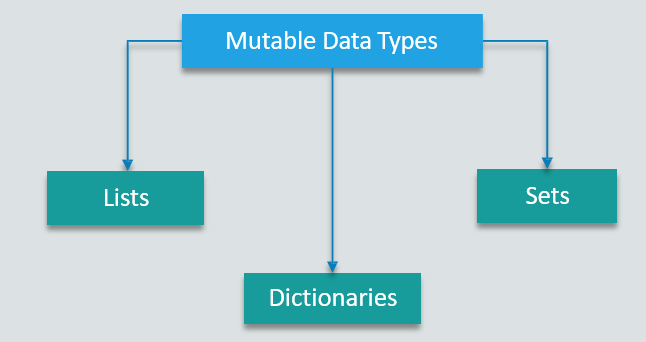
**String Data Type**



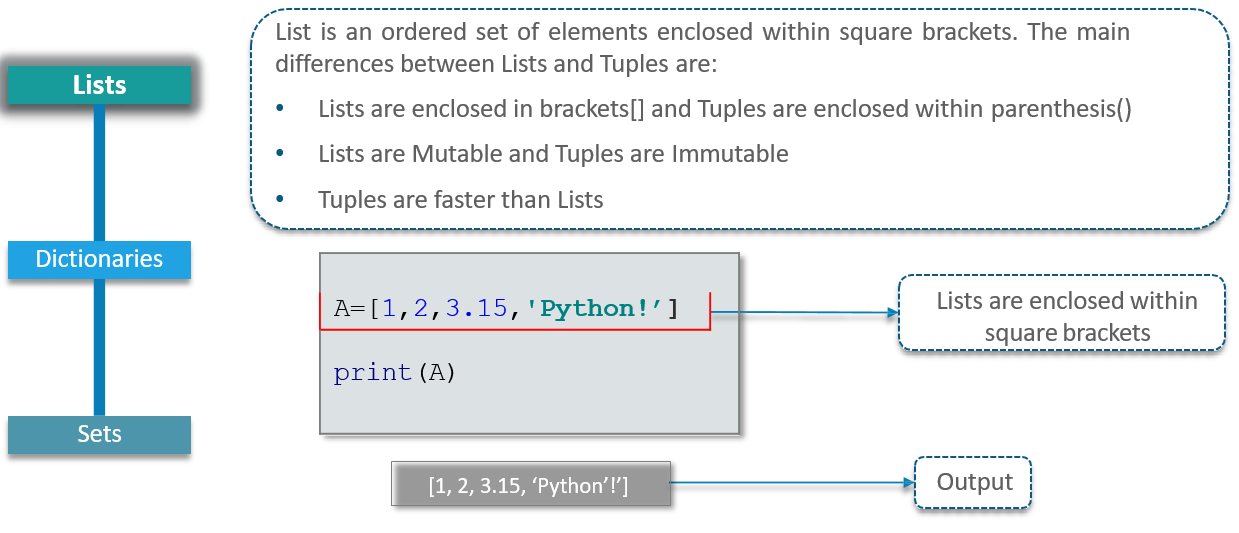
**Tuples Data Type**



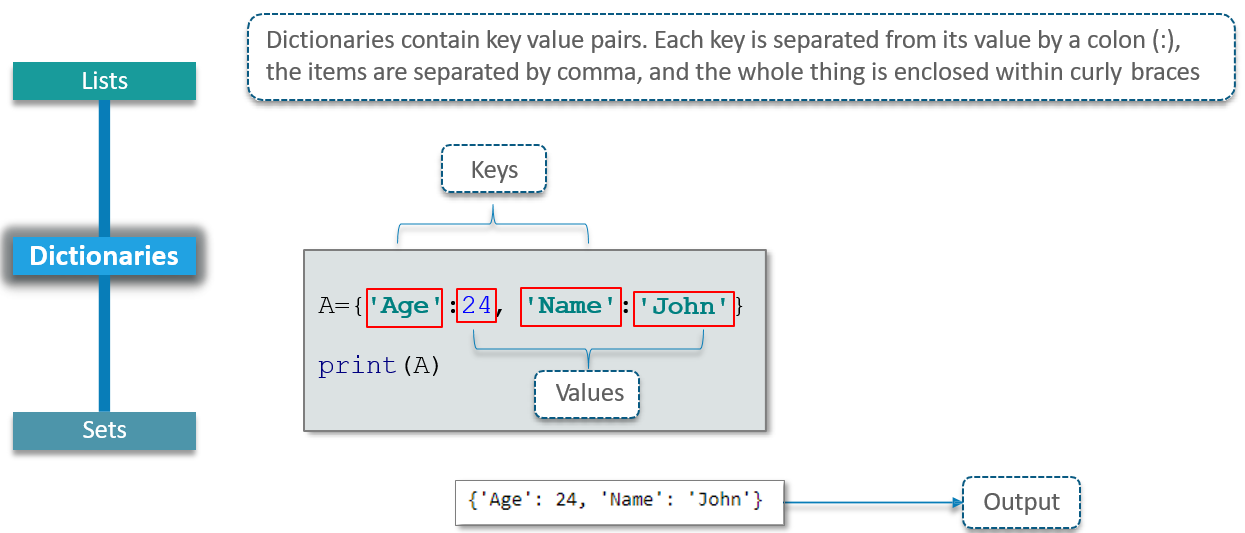
**Mutable Data Type**



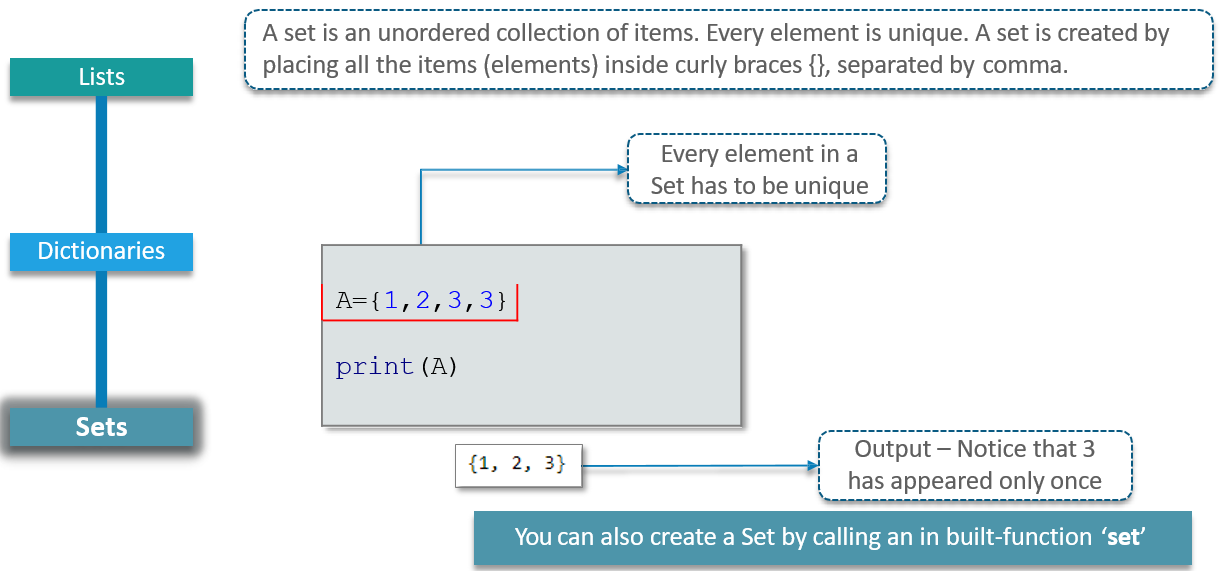
**List Data Type**



**Dictionaries Data Type**



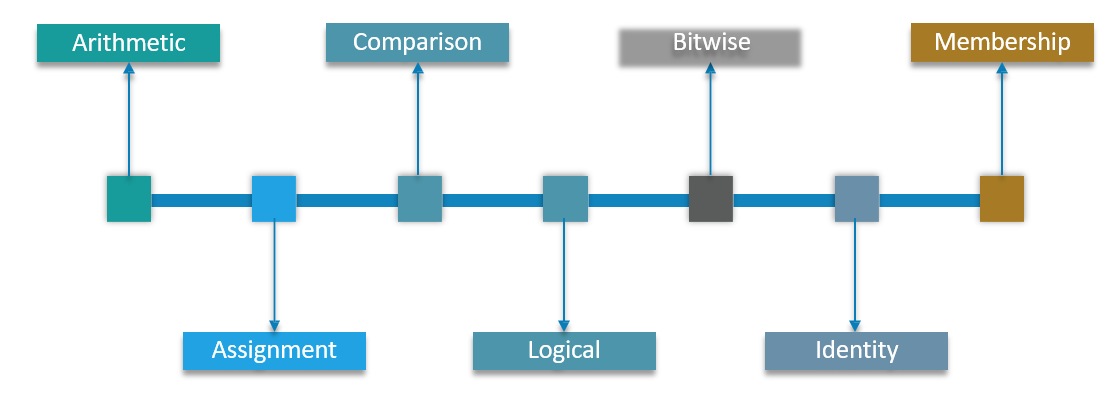
**Sets Data Type**



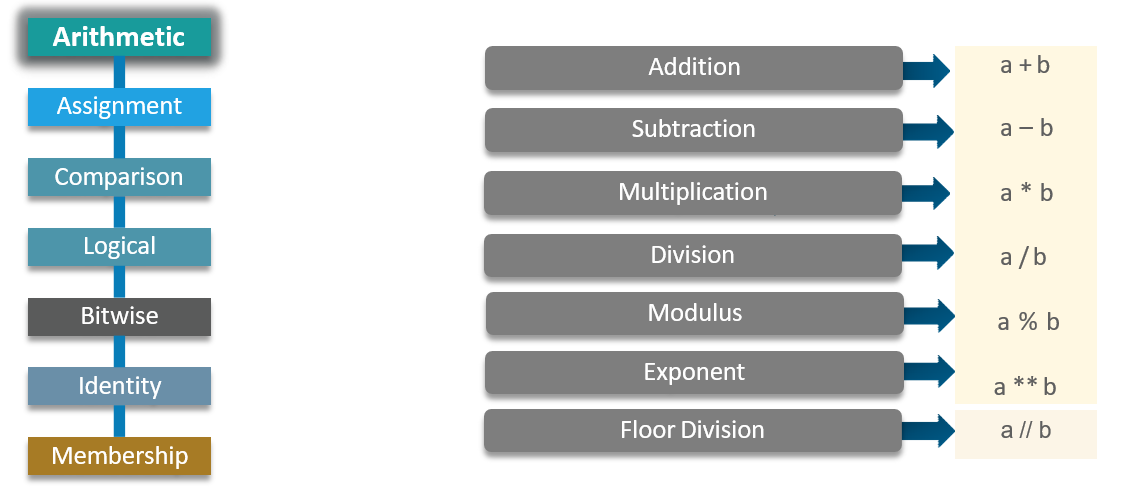
## 2.5 Operators

**What are Python operators?**

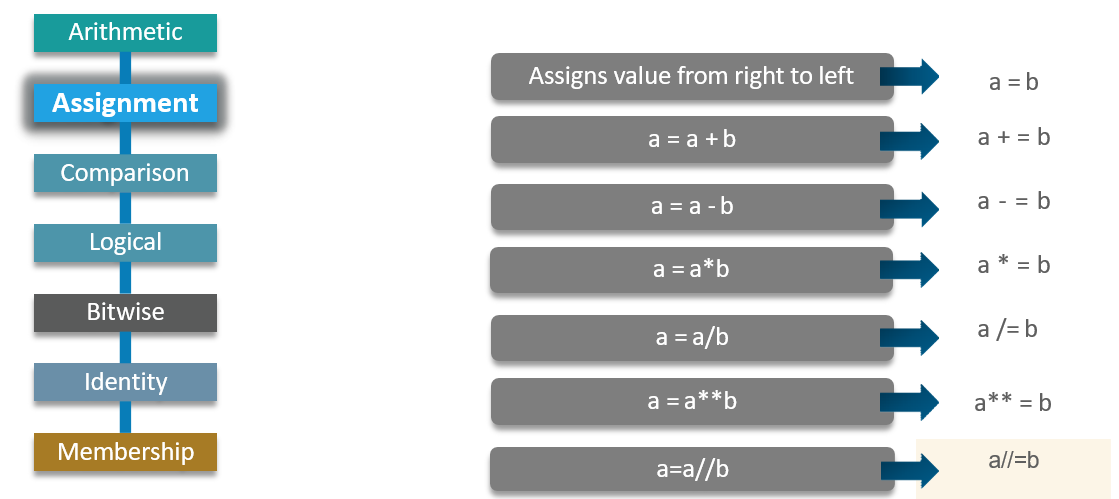
Operators are the constructs which can manipulate the values of the Operands. Consider the expression 2 + 3 = 5, here 2 and 3 are Operands and + is called Operator.



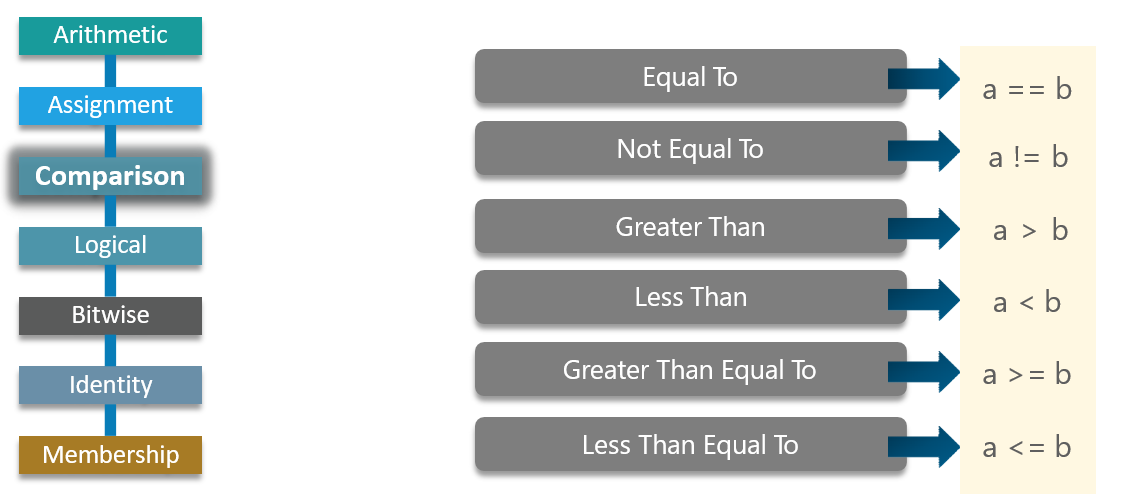
**Arithmetic Operators**



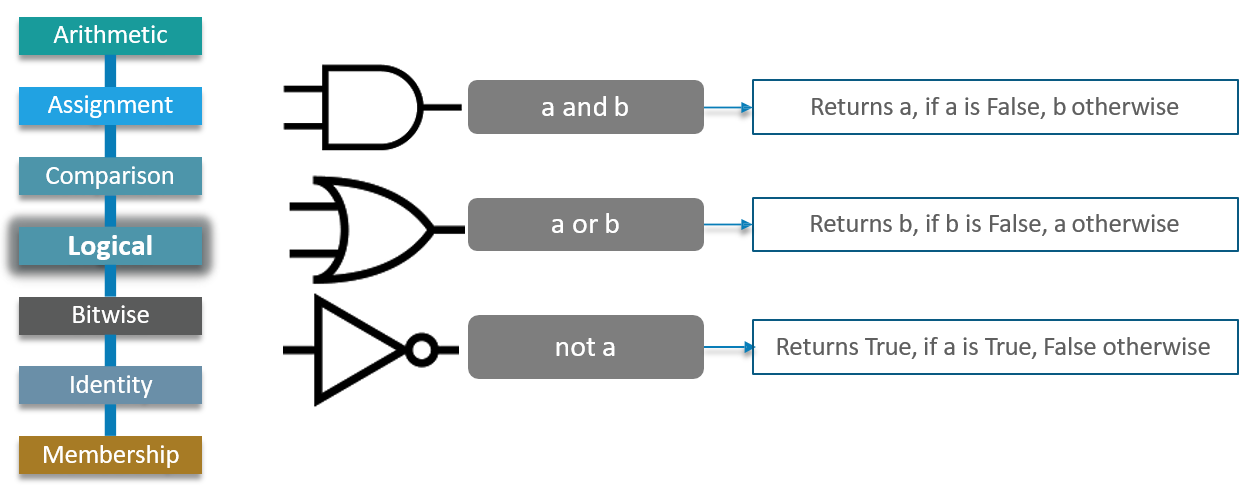
**Assignment Operators**



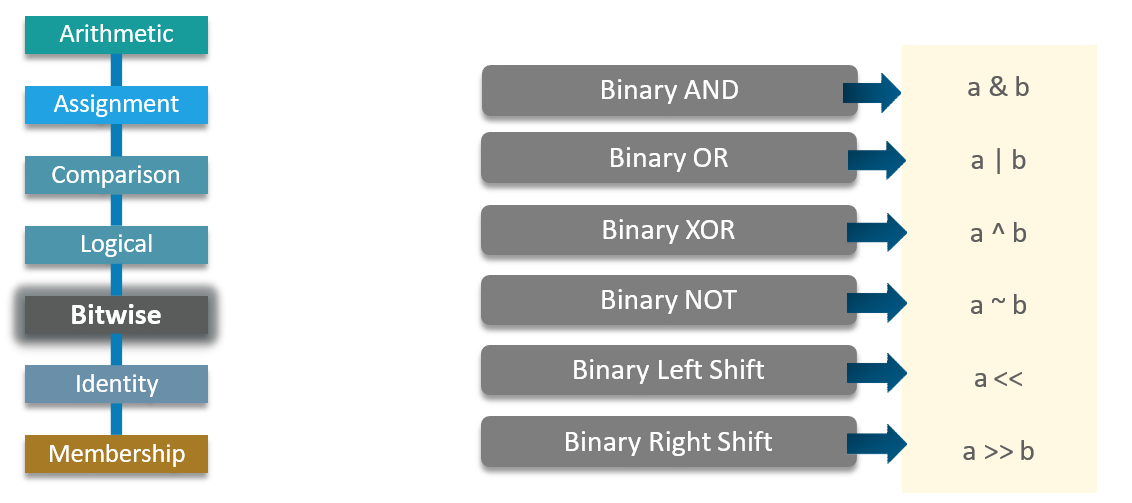
**Comparison Operators**



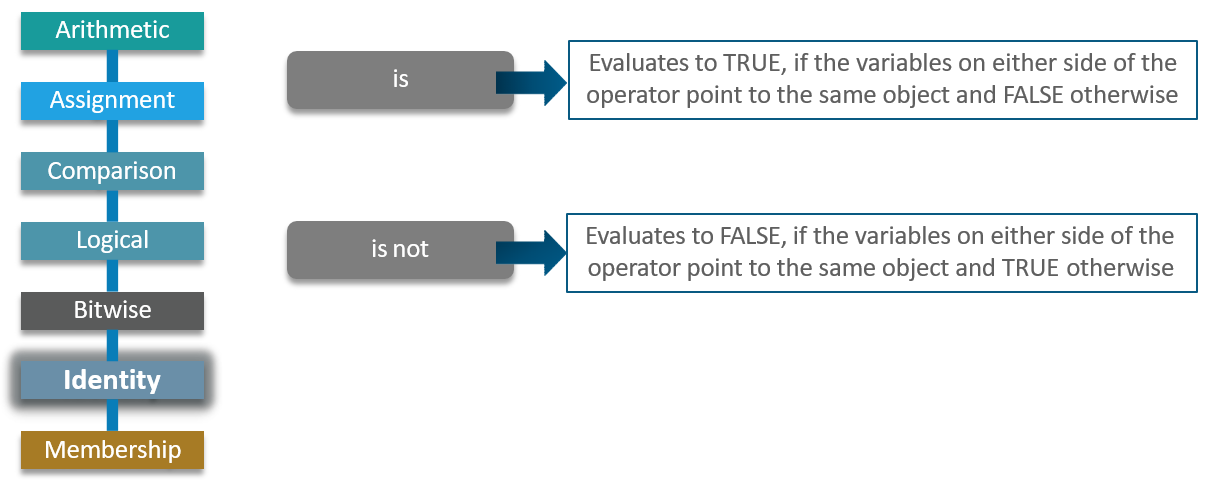
**Logical Operators**



**Bitwise Operators**



**Identity Operators**



**Membership Operators**

