

## Unit I

Overview of Programming : Structure of a Python Program, Python Interpreter, Using Python as calculator, Python shell, Indentation. Atoms, Identifiers and keywords, Literals, Strings, Operators(Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment, Operator, Ternary operator, Bit wise operator, Increment or Decrement operator).

---

### Python

**Python** is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Python is *easy to learn* yet powerful and versatile scripting language, which makes it attractive for Application Development.

Python supports *multiple programming pattern*, including object-oriented, imperative, and functional or procedural programming styles.

Python is not intended to work in a particular area, such as web programming. That is why it is known as *multipurpose* programming language because it can be used with web, enterprise, 3D CAD, etc.

We don't need to use data types to declare variable because it is *dynamically typed* so we can write `a=10` to assign an integer value in an integer variable

Python makes the development and debugging *fast* because there is no compilation step included in Python development, and edit-test-debug cycle is very fast.

## History

Python was invented by **Guido van Rossum** in 1991 at CWI in Netherland. The idea of Python programming language has taken from the ABC programming language or we can say that ABC is a predecessor of Python language.

There is also a fact behind the choosing name Python. Guido van Rossum was a fan of the popular BBC comedy show of that time, "**Monty Python's Flying Circus**". So he decided to pick the name **Python** for his newly created programming language.

## Why learn Python

Python provides many useful features to the programmer. These features make it most popular and widely used language. We have listed below few-essential feature of Python.

- Easy to use and Learn
- Expressive Language
- Interpreted Language
- Object-Oriented Language
- Open Source Language
- Extensible
- Learn Standard Library
- GUI Programming Support
- Integrated
- Embeddable
- Dynamic Memory Allocation
- Wide Range of Libraries and Frameworks

## Where is Python used

Python is a general-purpose, popular programming language and it is used in almost every technical field. The various areas of Python use are given below.

- Data Science
- Data Mining
- Desktop Applications
- Console-based Applications
- Mobile Applications
- Software Development
- Artificial Intelligence
- Web Applications
- Enterprise Applications
- 3D CAD Applications
- Machine Learning
- Computer Vision or Image Processing Applications.
- Speech Recognitions

## Python Popular Frameworks and Libraries

Python has wide range of libraries and frameworks widely used in various fields such as machine learning, artificial intelligence, web applications, etc. We define some popular frameworks and libraries of Python as follows.

- **Web development (Server-side)** - Django Flask, Pyramid, CherryPy
- **GUIs based applications** - Tk, PyGTK, PyQt, PyJs, etc.
- **Machine Learning** - TensorFlow, PyTorch, **Scikit-learn**, Matplotlib, Scipy, etc.
- **Mathematics** - Numpy, Pandas, etc.

## Python Statements

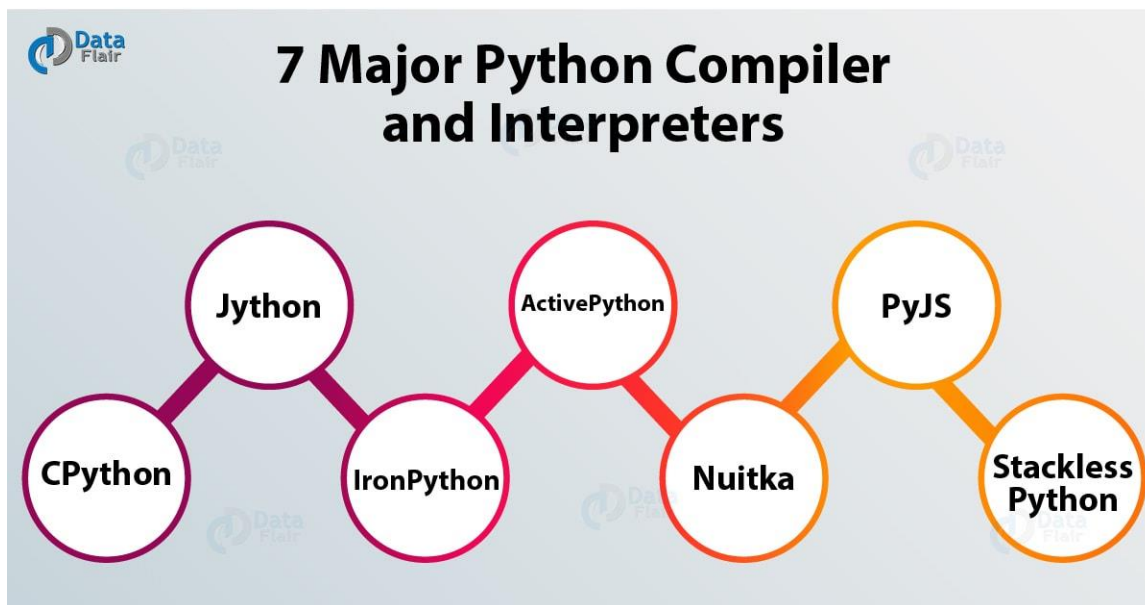
**Python Statements** In general, the interpreter reads and executes the statements line by line i.e sequentially. Though, there are some statements that can alter this behavior like conditional statements. Mostly, python statements are written in such a format that one statement is only written in a single line. The interpreter considers the ‘new line character’ as the terminator of one instruction.

Ex :

**Print(“hello python”)**

**Multiple Statements per Line** We can also write multiple statements per line, but it is not a good practice as it reduces the readability of the code. Try to avoid writing multiple statements in a single line. But, still you can write multiple lines by terminating one statement with the help of ‘;’. ‘;’ is used as the terminator of one statement in this case.

```
a = 10; b = 20; c = b + a  
print(a); print(b); print(c)
```



## Variables

Variable is a name that is used to refer to memory location. Python variable is also known as an identifier and used to hold value.

In Python, we don't need to specify the type of variable. Variable names can be a group of both the letters and digits, but they have to begin with a letter or an underscore.

## Identifier Naming

Variables are the example of identifiers. An Identifier is used to identify the literals used in the program. The rules to name an identifier are given below.

- The first character of the variable must be an alphabet or underscore ( \_ ).
- All the characters except the first character may be an alphabet of lower-case(a-z), upper-case (A-Z), underscore, or digit (0-9).
- Identifier name must not contain any white-space, or special character (!, @, #, %, ^, &, \*).
- Identifier name must not be similar to any keyword defined in the language.
- Identifier names are case sensitive; for example, my name, and MyName is not the same.
- Examples of valid identifiers: a123, \_n, n\_9, etc.
- Examples of invalid identifiers: 1a, n%4, n 9, etc.

## Declaring Variable and Assigning Values

Python does not bind us to declare a variable before using it in the application. It allows us to create a variable at the required time.

We don't need to declare explicitly variable in Python. When we assign any value to the variable, that variable is declared automatically. The equal (=) operator is used to assign value to a variable.    `x=y=z=50    a,b,c=5,10,15`

## Keywords

Python Keywords are special reserved words that convey a special meaning to the compiler/interpreter. Each keyword has a special meaning and a specific operation. These keywords can't be used as a variable. Following is the List of Python Keywords.

True	False	None	and	as
assert	def	class	continue	break
else	finally	elif	del	except
global	for	if	from	import
raise	try	or	return	pass
nonlocal	in	not	is	lambda

## literals

literals are a notation for representing a fixed value in source code. They can also be defined as raw value or data given in variables or constants.

Python has different types of literals.

1. **String literals**
2. **Numeric literals**
3. **Boolean literals**
4. **Literal Collections**
5. **Special literals**

### String literals

A string literal can be created by writing a text(a group of Characters ) surrounded by the single(""), double(""), or triple quotes. By using triple quotes we can write multi-line strings or display in the desired way.

**Character literal**

It is also a type of string literals where a single character surrounded by single or double-quotes.

**Numeric literals**

They are immutable and there are three types of numeric literal :

1. Integer
2. Float
3. Complex.

**Integer :**

Both positive and negative numbers including 0. There should not be any fractional part.

**Float**

These are real numbers having both integer and fractional parts.

**Complex Literal**

The numerals will be in the form of **a+bj**, where '**a**' is the real part and '**b**' is the complex part.

**Boolean literals**

There are only two boolean literals in Python. They are **true** and **false**. In python, **True** represents the value as **1** and **False** represents the value as **0**.

Every variable in python holds an instance of an object. There are two types of objects in python i.e. **Mutable** and **Immutable objects**.

**Immutable Objects :** These are of in-built types like **int, float, bool, string, unicode, tuple**.

In simple words, an immutable object can't be changed after it is created.

**Mutable Objects :** These are of type list, dict, set . Custom classes are generally mutable.

## Literal Collections

There are four different types of literal collections

1. **List literals**
2. **Tuple literals**
3. **Dict literals**
4. **Set literals**

**List literals** :List contains items of different data types. The values stored in List are separated by comma (,) and enclosed within square brackets([]). We can store different types of data in a List. Lists are mutable. `number = [1, 2, 3, 4, 5]`

**Tuple literals** : A tuple is a collection of different data-type. It is enclosed by the parentheses ‘()’ and each element is separated by the comma(.). It is immutable.`even_number = (2, 4, 6, 8)`

**Dictionary literals** :Dictionary stores the data in the key-value pair. It is enclosed by curly-braces ‘{}’ and each pair is separated by the commas(.). We can store different types of data in a dictionary. Dictionaries are mutable.`alphabets = {'a': 'apple', 'b': 'ball', 'c': 'cat'}`

**Set literals** : Set is the collection of the unordered data set. It is enclosed by the {} and each element is separated by the comma(.).

**Special literals** :Python contains one special literal (None). ‘None’ is used to define a null variable. If ‘None’ is compared with anything else other than a ‘None’, it will return **false**.



## Operators

The operator can be defined as a symbol which is responsible for a particular operation between two operands. Operators are the pillars of a program on which the logic is built in a specific programming language. Python provides a variety of operators, which are described as follows.

- Arithmetic operators
- Comparison operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

### Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations between two operands. It includes + (addition), - (subtraction), \*(multiplication), /(divide), %(reminder), //(floor division), and exponent (\*\*) operators.

### Comparison operator

Comparison operators are used to comparing the value of the two operands and returns Boolean true or false accordingly. < , > <=, >=, ==

### Assignment Operators

The assignment operators are used to assign the value of the right expression to the left operand.

### Bitwise Operators

The bitwise operators perform bit by bit operation on the values of the two operands

## Logical Operators

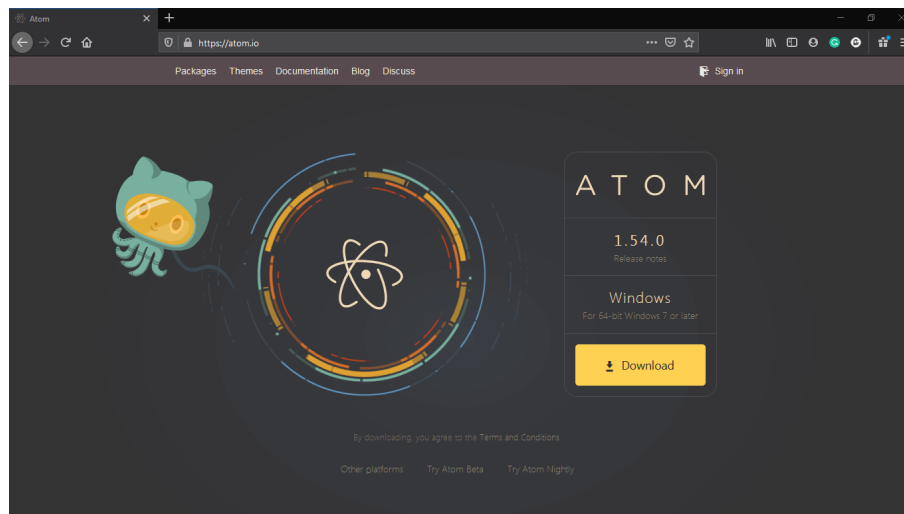
The logical operators are used primarily in the expression evaluation to make a decision.

## ATOMs

**Atom** is a text editor it doesn't come with any special functionalities. For instance **you can use** the **Atom** Editor to write your **Python** code/script but **you cannot** execute it.. **Atom** is lightweight and can be expanded with plug-ins.

**Atom** is a free, open-source, and multi-platform text editor or IDE which supports NodeJS developed packages and embedded Git control. Most of the extending packages are freely available and developed by open-source communities. Atom IDE is based on Electron Framework (previously called Atom Shell).

Atom does not contain any special features in the traditional sense; instead, it builds packages that store to its hackable core. These packages include features such as code lines, auto-complete, and code highlights.



## Indentation

Indentation is a very important concept of Python because without proper indenting the Python code, you will end up seeing `IndentationError` and the code will not get compiled. In simple terms indentation refers to adding white space (spaces and tabs) before a statement.

Python indentation is a way of telling a Python interpreter that the group of statements belongs to a particular block of code. A block is a combination of all these statements. Block can be regarded as the grouping of statements for a specific purpose. Most of the programming languages like C, C++, Java use braces { } to define a block of code. Python uses indentation to highlight the blocks of code. Whitespace is used for indentation in Python.

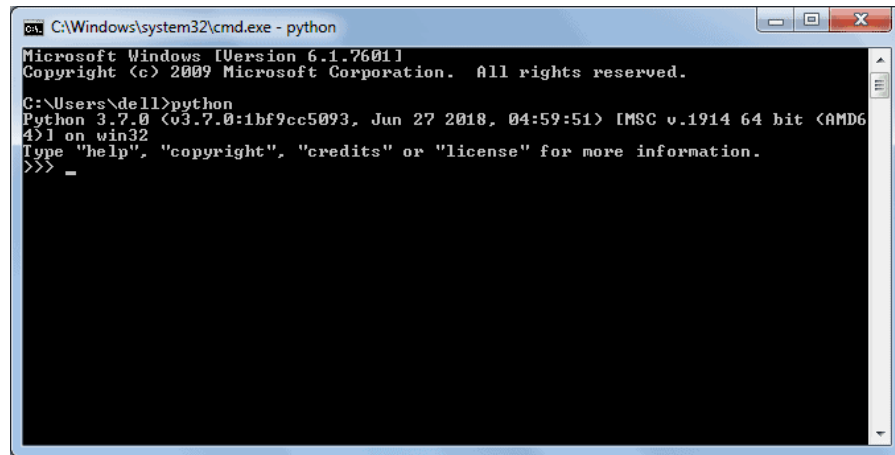
## Python Shell

**Python Shell** is a command line tool that starts up the **python interpreter**. You can test simple programs and also write some short programs. However, in order to write a more complexed **python** program you need an editor.

**Python** is an interpreter language. It means it executes the code line by line. **Python** provides a **Python Shell**, which is used to execute a single **Python** command and display the result.

It is also known as REPL (Read, Evaluate, Print, Loop), where it reads the command, evaluates the command, prints the result, and loop it back to read the command again.

To run the Python Shell, open the command prompt or power shell on Windows and terminal window on mac, write `python` and press **enter**. A Python Prompt comprising of three greater-than symbols `>>>`

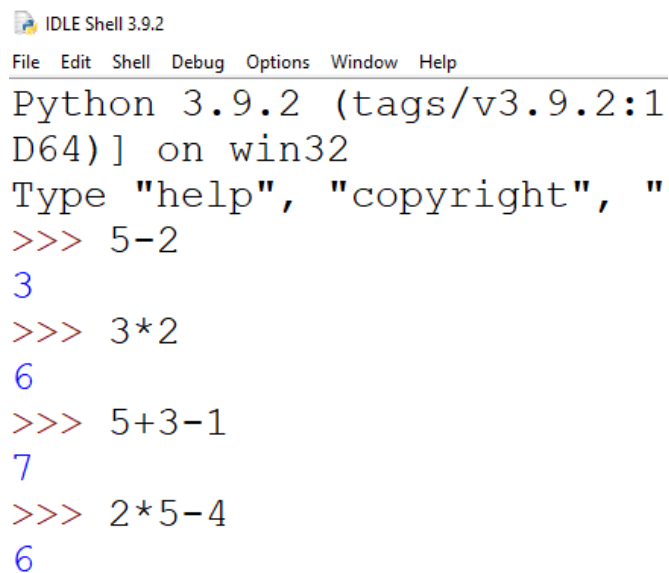


```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\dell>python
Python 3.7.0 <v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51> [MSC v.1914 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> _
```

## Simple Calculator

In Python, you can create a simple calculator, displaying the different arithmetical operations i.e. addition, subtraction, multiplication and division.



```
IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Python 3.9.2 (tags/v3.9.2:1
D64)] on win32
Type "help", "copyright", "
>>> 5-2
3
>>> 3*2
6
>>> 5+3-1
7
>>> 2*5-4
6
```