



PYTHON Programming - Introduction

www.infomaxacademy.com

Python Introduction

Python is Programming language.

- A Powerful “High Level Language”.
- Object-Oriented language
- Easy to read and program
- Scripting Language
- Free and Open source
- Interpreted language
- Compact Code
- Portable (Cross Platforms(OS))
- Huge collection of libraries

History of Python

- Python is a popular programming language. It was created in ***1991*** by ***Guido Van Rossum*****.**
- Developed at **National Research Institute of Mathematics and Computer Science** in *Netherland*
- Python 1.0 was released in November 1994.
- Python 2.0 was released in November 2000.
- Python 3.0 was released in November 2008.
- The most recent version 3.10.7 release on ***04-10-2021***

Application of Python

- Web and Internet Scripting
- Scientific and Numeric analysis
- Embedded Scripting
- Image processing
- Artificial intelligence
- Desktop GUI’s Application
- Database Programming
- Automation
- Weather Forecasting

Users of Python

- YouTube: Originally written in Python and mysql
- Yahoo!: Yahoo acquired Four11, whose address and mapping lookup services are implemented in Python
- Yahoo! Maps: Uses Python
- DropBox: A cloud based file hosting service runs on Python
- Google: Many components of the Google spider and search engine are written in Python

How to Download and Install Python

- There are multiple distributions available today
  - Default Python distribution (CPython)
  - Anaconda

- CPython https://www.python.org/
- Anaconda https://www.anaconda.com/

IDE (Integrated Development Environment)

- An IDE (Integrated Development Environment) understand your code much better than a text editor.
- It usually provides features such as build automation, code linting, testing and debugging.
- This can significantly speed up your work.

- Some popular IDE's for Python are:
  - IDLE
  - PyCharm
  - Visual Studio
  - Atom
  - Jupyter
  - Spyder
  - PyDev

Popular IDE's for Python

- IDLE (Integrated Development and Learning Environment) is a default editor. It is suitable for beginner level developers.
- PyCharm is a widely used Python IDE created by JetBrains. It suitable for professional developers and facilitates the development of large Python projects.
- Visual Studio Code is an open-source (and free) IDE created by Microsoft. It finds great use for Python development.
- Atom is an open-source code editor by Github and supports Python development.
- Jupyter is widely used in the field of data science.
- Spyder is an open-source IDE most commonly used for scientific development

Python Character Set

Character set is a set of valid characters that a language can recognize. A character represents any letter, digit or any other symbol.

- Letters : A-Z, a-z
- Digits : 0-9
- Special Symbol : + - * / & > < = ! % ! @ # ( ) { } _ etc.
- Whitespaces : blank space, Tabs, New line
- Others Characters : Python can process all ASCII and Unicode characters as part of data or literals.

Python Tokens

The smallest individual unit in a program is known as **Token** or a **lexical** unit.

- Keywords
- Identifiers (Names)
- Literals
- Operators
- Punctuators

Keywords

Keywords ate the words that convey a special meaning to the language compiler/interpreter.

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

Identifiers (Names)

Identifiers are used as the general terminology for the names given to different parts of program viz. variables, objects, classes, function, lists, dictionary etc.

Rules for forming identifier

- An identifier can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- An identifier must start with a letter or the underscore character
- An identifier can’t start with a number
- An identifier must not be a keyword.
- Identifier are case-sensitive (age, Age and AGE are three different variables)

Identify the valid Identifiers

1. simple_interest
2. Simple interest
3. 123num
4. num_123
5. num123
6. num@123
7. While
8. ELSE
9. True
10. NUMBER

Identify the valid Identifiers

1. simple_interest
2. Simple interest # contain the space
3. 123num # start with Digit
4. num_123
5. num123
6. num@123 # contained the special Char
7. While # while is keyword but it is While (i.e. start with cpas)
8. ELSE # else keyword but is in upper case
9. True # it is Keyword
10. NUMBER

Literals/Values

Literals (often referred to as *constant-Value*) are the data items that have fixed value. Python allows several kinds of literals:

- String Literals
- Numeric Literals
- Boolean literals
- Special Literal (None)
- Literal Collections

String Literals

- The text enclosed in quotes forms a string literal in Python. Ex. ‘a’, ’abc’, “abc” etc.
- Python allows you have certain nongraphic-characters in string values.
- Nongraphic characters are those characters that can’t be typed directly from keyboard. E.g. backspace, newline, tabs, etc.

Escape Character	Non-graphic character	Escape Character	Non-graphic character
\\	backslash	\n	New Line
\’	Single Quote	\t	Horizontal Tab
\”	Double Quote	\r	Carriage Return

Types of String Literals

- **Single Line String :** these strings that you created by enclosing text in single quotes (‘ ’) or double quotes (“ ”) are normally single-line string i.e. they must terminate in one line.

- **Multiline String :** Sometimes you need to store some text spread across multiple lines as one single line string. For that Python offers multiline Strings.
  - Multiline string can be created in two ways:
    - By adding backslash at the end of normal single-quote / double-quote strings.
    - By typing the text in triple quotation marks.

```
>>> 'infomax\
computer\
academy'
```

```
>>> '''infomax
computer
academy'''
```

Numeric Literals

The numeric literals in python can belong to any of the following three different numerical types:

- Int (Integer Number) ex: 4,-5,0
- Float (floating point real values) ex: 4.5,9.6,-5.6
- Complex (complex number) ex: 2+3i

```
>>> x=3+2j
>>> print(x.real,x.imag)
3.0 2.0
>>>
```

Boolean Literals

- A Boolean literals in python is used to one of the two Boolean values i.e. True or False. A Boolean literal can either have value as True or as False.

```
>>> flag=True
>>> print(flag)
True
>>>
```

Special Literal None

- Python has one special literal, which is None.
- The None literal is used to indicate absence of value.
- It is also used to indicate the end of list in Python.

```
>>> ans=None
>>> print(ans)
None
>>>
```

Literal Collections

- List
  - [5,6,7,8,8]
- Tuple
  - (5,6,7,8,8)
- Dictionary
  - {‘roll’:344,’age’:22}

type() Function

- type() function is used to know the datatype of specified value.

```
>>> a=4
>>> b=5.6
>>> c="rajesh"
>>> d=False
>>> print(type(a))
<class 'int'>
>>> print(type(b))
<class 'float'>
>>> print(type(c))
<class 'str'>
>>> print(type(d))
<class 'bool'>
```

Punctuators

- Punctuator are the symbol that are used in programming language to organize sentence structures, and indicates the rhythm and emphasis of expressions, statements, and program structure.
- Most common punctuators of python programming are:

‘	“	#	\
( )	[ ]	{ }	@
,	.	=	:

Python Variables

- Creating Variables
  - Unlike other programming languages, Python has no command for declaring a variable.
  - A variable is created the moment you first assign a value to it.

```
x = 5
y = "John"
print(x)
print(y)
```

Variables do not need to be declared with any particular type and can even change type after they have been set.

Execute Python Syntax

- Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")
```

Or

- by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```


25

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

```
if 5 > 2:
    print("Five is greater than two!")
```

- Python will give you an error if you skip the indentation:

```
if 5 > 2:
print("Five is greater than two!")
```


26

Comments In Python

- Python has commenting capability for the purpose of in-code documentation.
- Comments start with a #, and Python will render the rest of the line as a comment:

Docstrings

- Python also has extended documentation capability, called docstrings.
- Docstrings can be one line, or multiline.
- Python uses triple quotes at the beginning and end of the docstring:

```
"""This is a
multiline docstring."""
print("Hello, World!")
```


27

Command-line String Input

- Python allows for command line input.
- That means we are able to ask the user for input.
- The following example asks for the user's name, then, by using the input() method, the program prints the name to the screen:

```
print("Enter your name:")
x = input()
print("Hello, " + x)
```


28

Python typecasting

- Typecasting is a process of explicitly specifying types of variables in a programming language.

Function for Typecasting in Python.

- **int()** - constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number) literal, or a string literal (providing the string represents a whole number)
- **float()** - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
- **str()** - constructs a string from a wide variety of data types, including strings, integer literals and float literals

Working with ASCII Value

- **ord()** function is used to get ASCII value of any given character.
- **chr()** function is used to get corresponding character from ASCII value.

```
>>> print(ord('A'))
65
>>> print(chr(65))
A
```

Programs

- WAP to input any character from user and print its ASCII value.
- WAP to input any ASCII value from user and print its corresponding character.
- WAP to input any Alphabet in Upper case from user and print in Lower case.
- WAP to input any Alphabet in Lower case from user and print in Upper case.

Python Operators

Operators are used to perform operations on variables and values.

Python divides the operators in the following groups:

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

32

Python Arithmetic Operators

- Arithmetic operators are used with numeric values to perform common mathematical operations:

Operator	Name	Example a=5,b=2
+	Addition	a+b=7
-	Subtraction	a-b=3
*	Multiplication	a*b=10
/	Division	a/b=2.5
%	Modulus	a%b=1
**	Exponentiation	a**b=25
//	Floor division	a//b=2

Program

- WAP to input 2 Integer number form user and print its Addition, Subtraction, Multiplication, Division, Floor Division and Reminder.
- WAP to input any integer Number and Print its Square and Cube.
- WAP to input Length and Breath (in decimal) and print the area of rectangle.
- WAP to input Principal amount, rate of interest and time from user and print Simple Interest.
- WAP to input No. of days and convert in week and days format
- WAP to input Number of seconds and convert in Hour, min and second format
- WAP input 2 digit number and find its sum of digit

Python Assignment Operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3

Multiple Assignment in Python

- In python we can assign multiple value in different variable in single statement.

Syntax:

var1,var2,var3,….=value1,value2,value3…

Example:

a,b,c=4,5,6

Hear we assign 4 to a,5 to b and 6 to c in single statement.

Python Comparison Operators

- Comparison operators are used to compare two values:

Operator	Name	Example x=5, y=4
==	Equal	x == y FALSE
!=	Not equal	x != y TRUE
<>	Not equal	x <> y TRUE
>	Greater than	x > y TRUE
<	Less than	x < y FALSE
>=	Greater than or equal to	x >= y TRUE
<=	Less than or equal to	x <= y FALSE

Find the Output

Give the output of the Following when num1=4, num2=3, num3=2

- num1+=num2+num3
  print(num1)

- num1=num1**(num2+num3)
  print(num1)

- num1**=num2+num3
  print(num1)

- num1='5'+'5'
  print(num1)

Find the Output

Give the output of the Following when num1=4, num2=3, num3=2

- num1+=num2+num3
  print(num1) # 9

- num1=num1**(num2+num3)
  print(num1) #

- num1**=num2+num3
  print(num1)

- num1='5'+'5'
  print(num1)

Python Logical Operators

- Logical operators are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Python Identity Operators

- Identity operators are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

Operator	Description	Example
is	Returns true if both variables are the same object	x is y
is not	Returns false if both variables are the same object	x is not y

41

Python Membership Operators

- Membership operators are used to test if a sequence is presented in an object:

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns False if a sequence with the specified value is present in the object	x not in y

42

Bitwise Operator

Operator	Description	Example	Explanation
&	Bitwise AND	4&5 = 4	100&101=100=4
|	Bitwise OR	4&5 = 5	100|101=101=5
^	Bitwise XOR	4^5=1	100^101=001=1
>>	Bitwise Right Shift	7>>2=1	111>>2=001=1
<<	Bitwise Left Shift	5<<2=20	101<<2=10100=20

Operator Precedence and Associativity

- All operators have two important property called precedence and associativity. Both properties affect the operands are attached to operator. Precedence means the order in which two different kinds of operators should be applied in expression. Associativity means the order in which the two operators with same precedence should be applied in expression.

Operator	Description	Associativity
( )	Parentheses	left-to-right
**	Exponent	right-to-left
* / %	Multiplication/division/modulus	left-to-right
+ –	Addition/subtraction	left-to-right
<< >>	Bitwise shift left, Bitwise shift right	left-to-right
< <= > >=	Relational less than/less than or equal to Relational greater than/greater than or equal to	left-to-right
== !=	Relational is equal to/is not equal to	left-to-right
is, is not in, not in	Identity Membership operators	left-to-right
&	Bitwise AND	left-to-right
^	Bitwise exclusive OR	left-to-right
|	Bitwise inclusive OR	left-to-right
not	Logical NOT	right-to-left
and	Logical AND	left-to-right
or	Logical OR	left-to-right
= += -= *= /= %=	Assignment Sort hand assignment Operators	right-to-left

Find the Output

- print(4.00/(2.0+2.0))
- num1=2+9*((3*12)-8)/10
  print(num1)
- num=24//4//2
  print(num1)
- num1=float(10)
  print(num1)
- num1=int('3.14’)
  print(num1)
- print('Bye'=='BYE’)
- print(10!=9 and 20>=20)
- print(10+6*2**2!=9//4-3 and 29>=29/9)
- print(5%10+10<50 and 29<=29)
- print((0<6) or (not(10==6) and (10<0)))

THANK YOU

Address: G. R. Complex Preetam Nagar, Dhoomanganj
Prayagraj Uttar Pradesh 211011

Mobile No.: 9598948810,8874588766,9532878816
Website : www.infomaxacademy.com

