Chap- 1: Basics of Python Programing

Features of Python:

- Python is a 'high level', 'free and open source' programming language.
- It is portable, simple and easy to learn
- Python do not require 'compilation', it is interpreted
- It is "Object Oriented Programming language"
- It is embeded within 'C' or 'C++' programs.
- 'Rich set of functionality' available in it's huge standard library.
- Python has a powerful set of 'buil-in data types' and easy to use control constructs.

Top companies are using Python as their business application development.
Central Intelligence Agency - (CAI) is using python to maintain their website.
Google's first search engien was written in Python.
Facebook uses Python language.
NASA use 'Workflow Automation Tool' which is written in Python.
Instagram uses Python for it's front-end.
YouTube also uses scripted Python for their websites.

History and Future of Python:

Python is a General purpose programing language.
Python was created by "Guido van Rossum" during 1985-1990.
It has many versions like 2.7, 3.0, 3.1, 3.1-3.
3.1.4, 3.1.7, etc.
3.9.0 is the latest version of Python.

Future of Python:

Python Programming language is extensively used for 'web development', 'application development', 'system development', 'developing games' and so on.

- In Artificial Inteligence, there are plenty of Python frameworks, libraries and tools are specially developed to direct the Artificial Intelligence.
- Machine Learning with Python has made it possible to recognize images, videos, speech recognition and much more.
- The future scope of Python programming language can also be predicted by the way it has helped 'Big Data' technology to grow through it's high performance toolkits and libraries.
- Networking is another field in which Python has a brighter scope in future.
- Further development on Python pramgramming are also in process which will make it more compatible and fast processing.

Programming Topics

#1. Comment: (#)

Coment in Python is used to give the additional or extra information about the instruction or code.

It is a part of a program but not executable.

Comment cannot be used as output

There are two type of Comment.

a) Single Line Comment:

This Comment is specified or written in single line only.

Hash - # symbol is used to represent Single line Comment.

Example: # write a program to add 2 no.
print the result as sum.....

b) Multi Line Comment:

This comment is specified or written in more than one line.

Triple single quote ''' comment ''' are used to represent Multi Line Comment.

Example: ''' python is object oriented programing language.
it has vast library and rich control statements.
it is embeded within the C/ C++ programing language '''

#2. Reserved Keywords:

Reserved Keywords are those words which are already known to Python interpreter. Keywords are used to perform some specific purpose or particular task.
There are 35 keywords in Python.

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

#3. Identifiers:

The name of the 'variable' that we assign are the Identifiers.
Identifier are used to identify the entity uniquly in entire program.
Name of variable, constant, function or class are the example of identifiers.

There are some rules for assigning the variable (Identifier).

Rule-1: Identifier should always start with 'alphabate' or underscore '_' followed by any combination of alphabet or number.

Example: name = 'Lucky', name_2 = 'Nikhil'
num1 = 57, num2 = 89
_food = 'pizza', ...

Rule-2: Identifier should never start with number or any special cherector.

Example: 1_number = 57, @_name = 'Lucky', ...

Rule-3: Identifier name should not contain any blank-space and any special character.

Example: name of_person = "Lucky"
mobile@games = "Dominance"

Rule-4: Keyword are not used as Identifier.

Example: True = 57, False = 0, and = 7,
import = 'apple', global = "Pandemic", ...

Rule-5: The Identifier like NUMBER and number are different as in case python is sensitive.

Example: Number = 5 ≠ number = 5

name = 'Joker' ≠ NAME = "Joker"

#4. Variable and Constant:

➤ Variables:

Variables are nothing but reserved memory locations to store values.
Value stored in variable can be changed through out program.
Variable name are given as per Identifier rule
Generally variable name denote it's purpose

Example: number = 59, name = "Ryuga",
game = "X-caliber", OS = "pheonix", ...

2) • We can assign one value to multiple variable at one time.

e.g. pen = pencil = eraser = 5

1) • We can assigne one value to one variable at a time

e.g. cake = "chocklate", drink = "Red Bull"

3) • We can assign multiple value to one variable at a time

e.g. Vowels = 'a', 'e', 'i', 'o', 'u'
(considered as tuple)

4) • We can assign multiple value to multiple variable at a time.

e.g. first_1, second_2, third_3 = 'gold', 'Silver', 'Bronze

Constant:

Constant are those variable which are given fixed value.
Value stores in constant can not be changed through out the program.
In general Constant are written in Capital letters.

Example: PI = 3.14, LAMBDA = 5.378,
RADIUS = 64000, MU = 4.4, ...

Note # Literal Constant:

Literal are the Raw Data given to Variable or Constant.
(for more info. see pg.no.--)

#6 Data Types and Data Type Conversion:

Data Type:

Data Type are used to specify which type of value is stored in a variable.
The Python Interpreter take it automatically depend on value assigned
type() function is used to identify the datatype stored in a variable.
We need not to declare the datatype while declaring a variable.

In Python there are five types of Data-type that are used commonly

1] Numeric: Numeric datatype is used to hold Numeric value.

a) Int: It holds all the integer value also Binary, Octal, Decimal, Hexadecimal numbers

b) long: It holds the long Integer value.

c) float: It holds numbers with decimal point. it's accuracy is upto 15 decimal place

d) complex: It holds complex number (real + imagi -nary value)

2] String data-type: String datatype holds collection of numbers, alphabate and characters.
String data are always enclosed in ' ', " ", ''' ''', """ """.

Example: name = "Lucky", num = 2.9, char = "$"
nam_num = "2 is even", num_char = "@120"
nam_char = "Hello!"
nam_num_char = "I got 99.99%"

There are two operator that we can use with string '+' or '*'

a) '+' - concatenation: it is used to concatenate two or more string i.e. to join the string
Example: str_1 = "Lucky", str_2 = "awesome"
print (str_1 + str_2)
→ Luckyawesome.

b) '*' - repitition: It is used to repeate a string.

Examples: str_1 = "Hello"
print (str_1 * 3)
→ HelloHelloHello

3] List: It is a collection of same or different type of data, enclosed within [ ] square bracket and separated by ';'

Example: list = ['a', 'e', 'i', 'o', 'u']
list_1 = [10, 20, 'a', 'b', 30, '&']
It is like array in C/C++

4] Tuple: It is also a collection of same or different types of data.
The elements are enclosed within round bracket( ) and separated by ','
Tuple are immutable data-type i.e we cannot change the size of tuple or we cannot change the elements of tuple.

Example: tuple_1 = ('h', 'o', 'h', 'o', 'u', 'r')
tuple_2 = ('one', '=', 1, '₹', 'two', '=', '2')

5] Dictionary:
Dictionary is a collection of elements in the form of 'key: value' pair.
First value is a key value and second is it's value.
It is enclosed in curly bracket { } and separ ','

Example: dict_1 = {'one': 1, 'two': 2, 'three': 3}
dict_2 = {'pen': '5₹', 'pencil': '5₹'}

6] Set: It is a collection of same or different type of unique data.
It is enclosed in curly bracket { } and separated by commas ','
It will take the value only once.

Example: set_1 = {'a', 'e', 'i', 'o', 'u'}
set_2 = {'a', 'e', 'i', 'a', 'e', 'o', 'u'}

› Data-type Conversion:
Some we need to convert the data-type of variable. data-type conversion function are used to convert one type of data into another type. following are data-type conversion function.

int(), float(), complex(), bin(), oct(), hex()
str(), bool(), list(), tuple(),
dict(), set()

1) float, string, bin, oct, hex, bool - all these datatype can be converted to 'int'.

bin - 0b , oct - 0o , hex - 0x

2) int, string, bool, bin, oct, hex - can be converted into float.

3) int, float, bool, bin, oct, hex - converted to complex string.

4) int, float, complex, bool, bin, oct, hex - converted to list, tupple, dict, set. String.

5) 0/False is only considered as False and rest of everything in python database is considered as True.

6) int, bool, oct, hex converted into bin

7) int, bool, bin, oct converted into hex

8) int, bool, bin, hex converted into oct

9) string, tuple, dict, set converted into list

10) string, list, dict, set converted into tuple.

11) No data-type can be converted into dict

15) ~~string~~ ~~No~~ ( ) data-type can be converted into set.
String, list, tuple, dict.

6 Input Operation:

In python it is possible to input the data in the program from the users.

The function input( ) is used to input data.

Syntax: input([Prompt:])

where Prompt is the string datatype which indicates which type of data is to input.

Everything taken from input( ) function is in the form of String data-type.

So before input( ) function specific type of data-type conversion function is to be used.

Example:
```
name = input()
addres = input()
number = int(input())
number_2 = int(input("Enter no.: "))
```

7. Output Operation:

In Python the output is displayed on monitor by using print function.

Syntax: `print( )`

Everything written in print() function is considered as output, value separated by commas.
All datatype written in print function can be taken as output.

Example:
```
pint("Hello World")
print("Enter the number:")
print("How are you")
print(7,"+",3,'=',10)
print(3+7j,"is a complex no.")
```

- Format function:

This function is used to format the output.
By using format function we can assign the value according the index.

Syntax: `format(variable)`

Example: `print("{0}is a son of {1}".format(sam,Harry))`

#8. Indentation[:]:

Indentation is a blank space at the begining of a logical line
Indentation is always followed by colon (:)
it consist of four blank spaces.
It is usually used with control flow statement and looping statements.
Python program get structured through indentation
Block of code are defined by their indentation.

Example:
```
a = int(input( ))
if a<=0:
    print("Positive no.")
else:
    print("Negative no.")
```

9. Operators and Expression:

- **Operator:**
  Operator are symbol that are used to perform a particular operation.
  e.g. +, -, *, /, etc.

- **Operand:**
  The value that operator uses for performing operation or computation are called Operands.
  e.g. 5 + 7 = 12, 8 / 2 = 4, 3 - 2 = 1, etc.

- **Expression:**
  Expression is a combination of Operand and Operator.
  e.g. 9//2 = 4, 7/3 = 2.33, 3-3 = 0, etc.

1) Arithmetic Operators:

These operators are used to perform arithmetic Operation and mathematical calculations.

Operator	Read as	Meaning	Example
+	plus	Addition operator used for performing addition of two number.	3+2=5
−	minus	Subtration operator used for performing subtraction of two number.	10-8=2
*	asterisk	Multiplication Operator used for performing multiplication of two number	5*2=10
/	slash	Division Operator used for performing division of two number Quotient.	8/2=4
%	mod	Modulus Operator used for division and return the remainder value.	9%2=1
**	double asterisk	Exponential Operator (power) used to take power of number	2**4 =16
//	double slash	Floor division Operator used for division and return the value in integer form	10//3 =3

2) Comparision Operators:

These operators are used to compare the values and establish relationship among them, result is always either True or False.

Operator	Meaning
==	equal to operator, used to show that two values are equal.
!= , <>	not equal to operator, used to show that two operands are not equal.
<	less than operator, show that left operand is less than right operand
>	greater than operator, show that left operand is greater than right operand
<=	less than equal to operator, show that left operand is either less or equal to right operand.
>=	greater than equal to operator, show that left operand is either greater than or equal to right operand.

3> Assignment operators:

These operator are used to assign a value to variable.

Operator	Meaning and Purpose.
+=	a+=b is equivalent to a=a+b adding right side operand to left side
-=	a-=b is equivalent to a=a-b subtracting right operand to left operand
=	assignment operator, used to assign a value or expression to a variable.
*=	a*=b is equivalent to a=a*b multiplying right operand with left operd
/=	a/=b is equivalent to a=a/b dividing right operand by left operand
%=	a%=b is equivalent to a=a%b mod dividing right operand by left
**=	a**=b is equivalent to a=a**b taking exponential of left operand to power of right operand
//=	a//=b is equivalent to a=a//b taking floor division of left operand by right operand.

4) Logical Operators:

These operator used to perform logical operations on variable having True or False value where the result is always either True or False. The value 0 is considered as False and all other data-type are considered as True.

Operator	Meaning.
and	a and b: True if both operand are True; otherwise False.
or	a or b: False if both operand are False, otherwise True.
not	a not: True if False False if True.

5) Membership Operator:

These Operators are used to find whether the specified value is present or not present in the sequence, string.
The result of Membership Operator is either True or False.

Operator	Meaning.
in	a in sequence: return True value if value 'a' is present in sequence.
not in	a not in sequence: return True value if value 'a' is not in sequence.

6) Identity Operator:

These operator are used to check whether one variable is indentical to other or not. OR.
Identity Operators are used to check whether one operand is point to same memory location.
The result of Identity Operator is either True or False

Operator	Meaning.
is	a is b : return True value if operand a is having same memory location as that of b
isnot	a is not b: return True value if operand a have different Memory location as that b.

7) String Operator:

String is collection of alphabate, number and character, there are two operation that can performed on String.

Operator	Meaning:
+	concatenation: used to join two or more strings.
*	repetition: used to repeat a string

8) Bitwise Operators:

Bitwise operator are used to perform Bitwise operation / Boolean operation.
0 and 1 are considered as bit, Bitwise operato work on bit / Binary of given value.
In this, given value is first get converted into Binary and then operation is performed, the result in Binary is produced which then converted to normal value.

Operator	Meaning
&	a & b: Bitwise and = multiplication used to perform logical and i.e (boolean multiplication) between a and b.
|	a | b: Bitwise or = addition used to perform logical or i.e. addition between a and b.
~	~a, ~b: Bitwise not = negation used to perform logical negation i.e. negative of given value
^	a ^ b: Bitwise X-OR = $\bar{a}b + a\bar{b}$ used to perform logical X-OR operation on a and b.
<<	a<<1 : left shift operator, a = 2 = 010 010a<<1, mean left shift all digits and add a zero at the last
>>	a>>1: right shift operator, 010 >>1, means make right shift by one digit and add zero at starting.

#10. Expression's in Python:

Expression is a combination of operator and operand (value)

The expressions in Python are evaluated according to precedence, the right side of equation i.e expression is evaluated first and then result is stored in left side ie in variable.

Example: a = (3 + 2) → expression

operator

operand

a+3, b+2*3, a=a/b, c=a/b, .. etc.

- Order of Execution:-

When more than one operator occurs in an expression, the order of evaluation depends on the rule of execution / Precedence.

Python follows same rule of Precedence as followed by Mathematics.

- The acronym PEMDAS is useful to remember that

1) Parentheses: P ( ) has the heighest precedence and can be used to force an expression in the order you want. 1*(5-2) = 3

2) E: Exponential has next heighest precedence
∴ 3+(2**3) = 11

3) M: multiplication and D: Division has same precedence and then A: addition and S: subtraction has same precedence

Operators with same precedence are evaluated from left to right.

• Python Operator Precedence:

Precedence order	Operator sign	Operator name
Heighest	( )	Parenthesys.
	**	Exponential
	+x, -x, ~	unary positive, unary negative, bitwise negation.
	*, /, %, //	multiplication, division modulus, floor division
	+, -	addition, subtraction
	<<, >>	left shift, right shift
	& ^ |	Bitwise and Bitwise X-OR Bitwise OR
	==, !=, <, <= >, >=	Comparision operator
	is, is not	Identity operator
Lowest	not, and, or	Boolean operator

Types of Expression:

There are three types of Expression in Python.

1) Infix Expression: it is a normal form of expression in the form of

Operand-1 operator Operand-2

e.g. a + b

2) Prefix Expression: it is a expression in the form of

Operator (Operand-1)(Operand-2)

e.g. + a b

3) Postfix Expression: it is a expression in the form of

(Operand-1)(Operand-2) Operator

e.g. a b +