CAP776

PROGRAMMING IN PYTHON

Created By:
Kumar Vishal
(SCA), LPU

Python basics

- introduction,
- data types and operators,
- control statements,
- functions,
- strings,
- lists,
- sets,
- tuples and dictionaries

Python?

Python is a general purposehigh-level programming language which supports Object Oriented programming approach to develop applications.

We can develop using Python:

- **Desktop applications** (Tkinter library, toolkits such as the wxWidgets, Kivy, PYQT)
- **Web applications**(frameworks are Django, Flask, Pyramid) web-framework named Django is used on **Instagram**
- **Machine Learning and Artificial Intelligence**

(Library: Pandas, Scikit-Learn, NumPy)

- **Data Science and Data Visualization (** Matplotlib, Seaborn)
- **Enterprise Applications** (Odoo, Tryton)
- **Scientific and Numeric** (SciPy, Pandas, Numpy)

Who developed?

Python was developed in 1989 by **Guido Van Rossum**

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.7	June 27, 2018
Python 3.8.0	Oct. 14, 2019
Python 3.8.5	July 20, 2020

Release version	Release date
Python 3.10.5	June 6, 2022
Python 3.9.13	May 17, 2022
Python 3.10.4	March 24, 2022
Python 3.9.12	March 23, 2022
Python 3.10.3	March 16, 2022
Python 3.9.11	March 16, 2022
Python 3.8.13	March 16, 2022

basic syntax

Comments :

Python supports two types of comments:

1) **Single Line Comment:**

This is single line comment.

2) **Multi Line Comment:**

Multi lined comment can be given inside triple quotes.

“””” (This is also called docstring in python)

Indentation

- To define a block of code. Python uses indentation.
- A code block (body of a function, loop etc.) starts with indentation and ends with the first un-indented line.

```
for i in range(1,11):

print(i)
```

Error

```
for i in range(1,11):
        print(i)
```

Correct

Keywords in Python programming language

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

Declaring and assigning a value to a variable

```
name = “kumar“
print(name)
```

Assigning multiple values to multiple variables

```
a, b, c = 5, 3.2, "Hello“
print (a)  print (b)  print (c)
```

Data Types

- Numbers
- String
- List
- Tuple
- Dictionary

Python provides us the **type()** function, which returns the type of the variable passed.

```
a=10
b=“hello“
c = 10.5
print(type(a))

print(type(b))

print(type(c))
```

Numbers

Number stores numeric values. The integer, float, and complex values belong to a Python Numbers data-type.

Int - Integer value can be any length such as integers 10, 2, 29, -20, -150 etc. Python has no restriction on the length of an integer. Its value belongs to **int**

Float - Float is used to store floating-point numbers like 1.9, 9.902, 15.2, etc. It is accurate up to 15 decimal points.

complex - A complex number contains an ordered pair, i. e., x + iy where x and y denote the real and imaginary parts, respectively. The complex numbers like 2.14j, 2.0 + 2.3j, etc.

Complex number

- Square root of negative number: For example:√-25

√25X-√1=5i

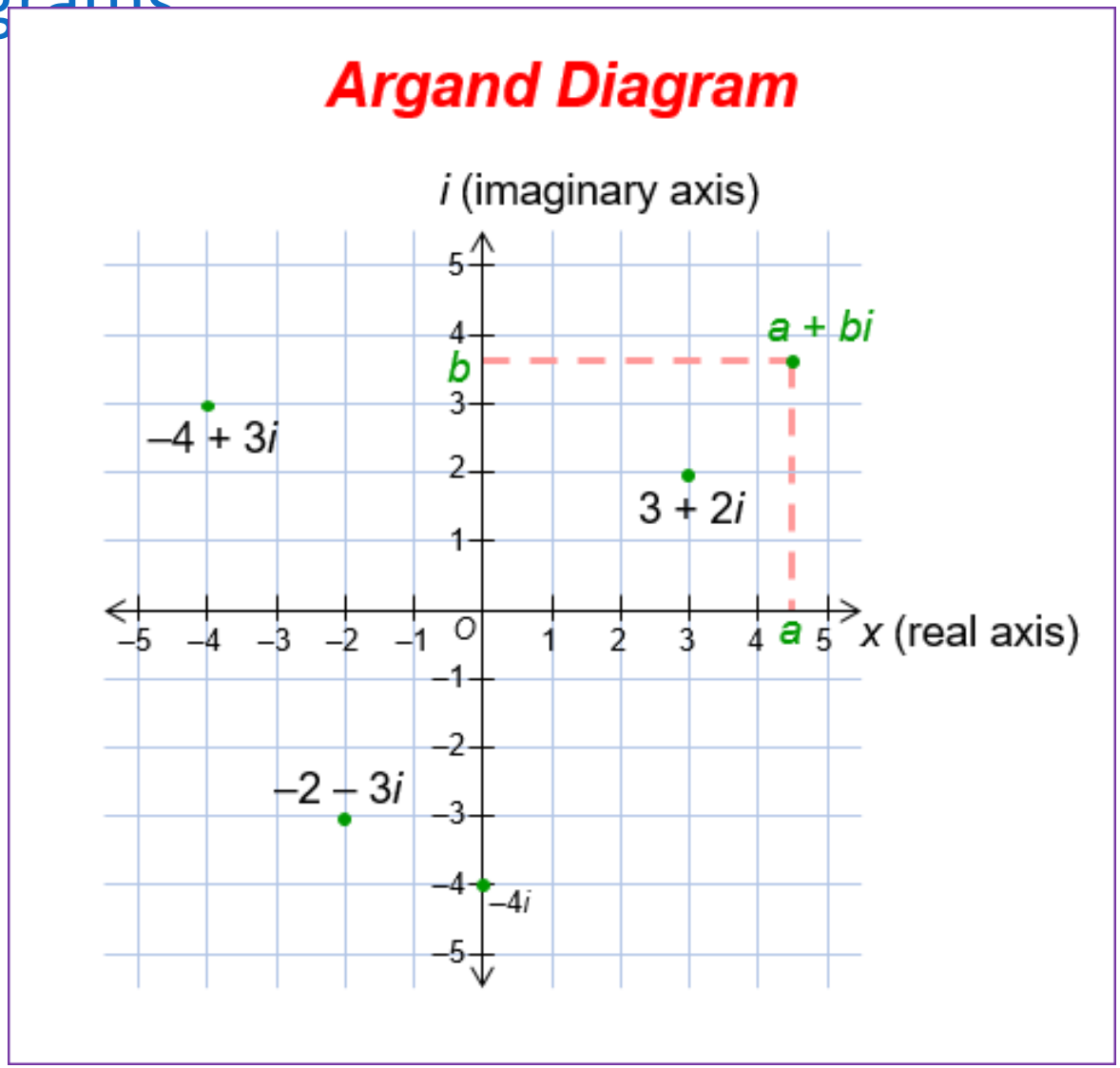
An imaginary number is multiple of i

= √-1

 We can add real number to imaginary number to form a complex number:

Ex: 5+5j

Complex number use is the use of Argand diagrams



Complex Numbers used in Applications:

- Signal Processing
- AC Circuit Analysis
- Quantum mechanics(It is field of physics which deals with motions and interaction between subatomic particles)

String

The string can be defined as the sequence of characters represented in the quotation marks. In Python, we can use single, double, or triple quotes to define a string.

Str=“hello”

str=“Hello”+123

print(str)

What will be print?

A. Hello123
B. Hello
C. Nothing will print
D. Error

constant

Declaring and assigning value to a constant

File->New->Create a constant.py file PI = 3.14

Create a main.py import constant

print(constant.PI)

What is a Module?

- A file containing a set of functions you want to include in your application.
- we can use the module we just created, by using the import statement:

```
import module_name
```

Reading Input from keyboard

The **input() method** reads a line from input, converts into a string and returns it.

```
inputString = input()
print('The inputted string is:', inputString)
```

Get input from user with a prompt

```
Num1=input("Enter num1\n")
print('The inputted number is:', Num1)
```

Taking multiple inputs from user in Python

By using split()

```
a, b = input("Enter a two value: ").split()
```

For displaying output using {} and .format()

```
txt ="first number is{},second no is{}".format(1,2)
print(txt)
```

List

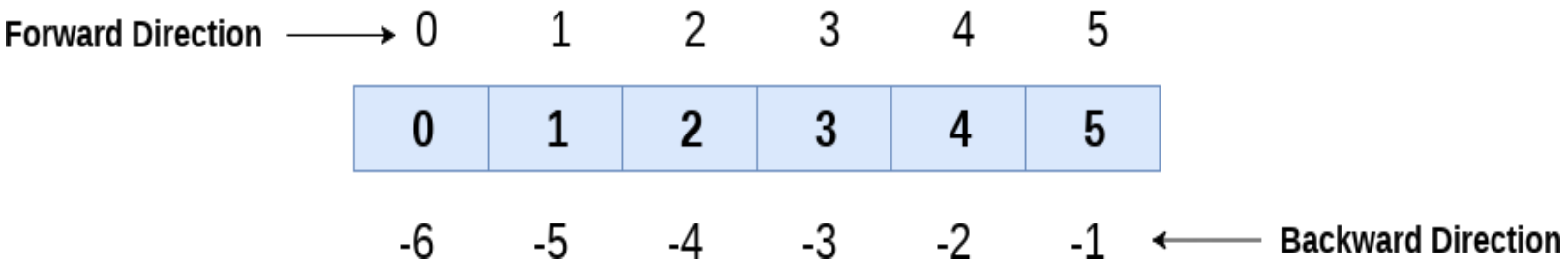
The items stored in the list are separated with a comma (,) and enclosed within square brackets [].

The list can contain data of different types.

list1 = [1, "hi", "Python", 2.5]

Accessing List By using indexing in Python

List = [0, 1, 2, 3, 4, 5]



Accessing List By using the slicing operator

List = [0, 1, 2, 3, 4, 5]

0	1	2	3	4	5

List[0] = 0

List[1] = 1

List[2] = 2

List[3] = 3

List[4] = 4

List[5] = 5

List[0:] = [0,1,2,3,4,5]

List[:] = [0,1,2,3,4,5]

List[2:4] = [2, 3]

List[1:3] = [1, 2]

List[:4] = [0, 1, 2, 3]

List[1:6:2]={2,4,6}

list_varible(start:stop:step)

(print(list[2:4])

Creating a list in Python by using the list() function

```
r=range(0,
10)  l=list(r)
print(l)
```

Output:[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Modifying List

```
l = [1, 2, 3, 4, 5]
print(l)
print("Before modifying l[0] : ",
l[0])  l[0]=20
print("After modifying l[0] : ",l[0])
print(l)
```

IMPORTANT FUNCTIONS OR METHODS OF LISTS IN PYTHON

len() function in python:

This function when applied to a list will return the length of the elements in it.

```
n = [1, 2, 3, 4, 5]
print(len(n))
```

count() method in Python:

This method returns the number of occurrences of a specific item in the list

```
n = [1, 2, 3, 4, 5, 5, 5, 3]
print(n.count(5))
```

append() method in Python:

Using this method we can add elements to the list. The items or elements will be added at the end of the list.

```
l=[]
l.
append(“CAP776")
l.
append(“CAP615")
l.
append(“CAP444")
print(l)
```

insert() method in Python:

We can add elements to the list object by using the insert() method. The insert() method takes two arguments as input: one is the index and the other is the element. It will add the elements to the list at the specified index.

```
n=[10, 20, 30, 40, 50]
n.insert(0, 76)  print(n)
```

extend() method in Python:

Using this method the items in one list can be added to the other.

```
l1 = [1,2,3]
l2 = ['Rahul', 'Rakesh', Kumar'] print('Before extend l1 is:', l1) print('Before extend l2 is:', l2)
l2.extend(l1)
print('After extend l1 is:', l1) print('After extend l2 is:', l2)
```

remove() method in Python:

We can use this method to remove specific items from the list.

```
n=[1, 2, 3]
n.remove(1)  print(n)
```

If the item exists multiple times, then only the first occurrence will be removed. If the specified item not present in list, then we will get ValueError.

pop() method in Python:

This method takes an index as an argument and removes and returns the element present at the given index. If no index is given, then the last item is removed and returned by default.

```
n=[1, 2, 3, 4, 5]
print(n.pop(1))  print(n)  print(n.pop())  print(n)
```

What will be output? n=[1, 2, 3, 4, 5]

print(n.pop(10))

Difference between remove() and pop()

remove()	pop()
To remove based on the element.	To remove based on index
It doesn't return any value	It returns the popped out element.
If the item is not present, the we get ValueError	If the index is not in the range, then we get IndexError

reverse() method in python:

This method reverses the order of list elements.

```
n=[1, 2, 3, 4, 'two']
print(n)  n.reverse()  print(n)
```

sort() method in Python:

- ✔ In a list by default insertion order is preserved.
- ✔ If we want to sort the elements of the list according to the default natural sorting order then we should go for the sort() method.
- ✔ For numbers, the default natural sorting order is ascending order.
- ✔ For strings, the default natural sorting order is alphabetical order.
- ✔ To use the sort() method, the list should contain only homogeneous elements otherwise, we will get TypeError.

Taking Input of a Python List

```
# input the list as string
string = input("Enter elements (Space-Separated):
")  lst = string.split()
print('The list is:', lst)  # printing the list  Using
```

append():

```
>>> l=[]
>>> str=input()
hi
>>> l.append(str)
>>> print(l)
['hi']
>>> str=input()
ki
>>> l.append(str)
>>> print(l)
['hi', 'ki']
>>>
```

range() Function

range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default)

Syntax:

- range(stop)
- range(start, stop,)
- range(start, stop, step)

Note: Python range() function doesn't support the float numbers. i.e. user cannot use floating-point or non-integer number in any of its argument. Users can use only integer numbers.

Tuple:

Like lists, tuples also contain the collection of the items of different data types. The items of the tuple are separated with a comma (,) and enclosed in parentheses ().

A tuple is a read-only data structure as we can't modify the size and value of the items of a tuple.

Mytuple=(1, "hi", "Python", 2)

Tuple = (0, 1, 2, 3, 4, 5)

0	1	2	3	4	5

Tuple[0] = 0

Tuple[1] = 1

Tuple[2] = 2

Tuple[3] = 3

Tuple[4] = 4

Tuple[5] = 5

Tuple[0:] = (0, 1, 2, 3, 4, 5)

Tuple[:] = (0, 1, 2, 3, 4, 5)

Tuple[2:4] = (2, 3)

Tuple[1:3] = (1, 2)

Tuple[:4] = (0, 1, 2, 3)

Functions/methods in tuple:

- max()
- min()
- len()
- tuple()
- count() method
- index() method

SN	List	Tuple
1	The literal syntax of list is shown by the [].	The literal syntax of the tuple is shown by the ().
2	The List is mutable.	The tuple is immutable.
3	The List has the a variable length.	The tuple has the fixed length.
4	The list provides more functionality than a tuple.	The tuple provides less functionality than the list.
5	The list is used in the scenario in which we need to store the simple collections with no constraints where the value of the items can be changed.	The tuple is used in the cases where we need to store the read-only collections i.e., the value of the items cannot be changed. It can be used as the key inside the dictionary.

Dictionary

Dictionary is an unordered set of a key-value pair of items.

The items in the dictionary are separated with the comma (,) and enclosed in the curly braces {}.

dictionary1={1:'Jimmy', 2:'Alex', 3:'john', 4:'mike’}

Employee = {"Name": “Rohan", "Age": 29,

"salary":25000,"Company":"GOOGLE"}

The values can be accessed in the dictionary by using the keys as keys are unique in the dictionary.

Keys can be immutable values. Like: int,float,string, Boolean,tuple

Updating Dictionary

employee = {“name": “Rohan", “age": 29, "salary":25000,“company": "GOOGLE"}

employee[‘age’]=34

employee[‘location’]=‘Bangalore’

Delete Dictionary Elements

employee = {“name": “Rohan", “age": 29, "salary":25000,“company":"GOOGLE"}

del employee['name’] # remove entry with key 'name'

employee. # remove all entries in

clear() del dict # delete entire

employee dictionary

Functions/Methods in Dictionary

len() gives the total length of the dictionary. This would be equal to the number of items in the dictionary.

clear() :dictionary method **clear()** removes all items from the dictionary.

fromkeys() :dictionary method **fromkeys()** creates a new dictionary with keys from *seq* and *values* set to value.

```
seq=('name','age')
dict1=dict1.fromkeys(seq)  print(dict1)
# {'name': None, 'age': None}
```

Functions/Methods in Dictionary

get() :dictionary method **get()** returns a value for the given key. If key is not available then returns default value None.

items() : dictionary method **items()** returns a list of dict' s (key, value) tuple pairs

keys() : dictionary method **keys()** returns a list of all the available keys in the dictionary.

values(): dictionary method **values()** returns a list of all the values available in a given dictionary.

Set

Sets are used to store multiple items in a single variable.

A set is a collection which is both *unordered* and *unindexed*.

Sets are written with curly brackets.

Set items are unordered, unchangeable, and do not allow duplicate values.

```
myset = {"apple", "banana", "cherry"}
```

Operators in Python

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

Arithmetic operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

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
&=	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

Comparison operators

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical operators

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)

Identity operators:

is, isnot

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.

```
a = 20
b = 20

if ( a is b ):
   print "Line 1 - a and b have same identity"  else:
   print "Line 1 - a and b do not have same identity"
```

The built-in id() a function is used to get the “identity” of an object.

Membership operators

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

```
x = ["apple", "banana"]

print("banana" in x)

# returns True because a sequence with the value "banana" is in the list
x = ["apple", "banana"]

print("pineapple" not in x)

# returns True because a sequence with the value "pineapple" is not in the
list
```

Bitwise Operators

Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

AND Operator (&)

If both side bit is on result will be On

a	b
0	0
0	1
1	0
1	1

Steps to solve:-

- a = 12 (find binary form:1100)
- b = 25 (find binary form:11001)

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
			1	0	0	0	8

a & b= 01100

(12)

11001 (25)

01000 (8)

Ans.

What will be output?

```
a=20;
b=25;
print((a&b));
```

A. 15

B. 16

C. 20

OR Operator (|)

If any side bit is on result will be On

a	b
0	0
0	1
1	0
1	1

Steps to solve:-

- a = 12 (find binary form:1100)
- b = 25 (find binary form:11001)

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
		1	1	1	0	1	29

a | b= 01100 (12)

11001 (25)

11101 (29)

Ans.

What will be output?

```
a=20;
b=25;
print((a|b));
```

A. 31

B. 32

C. 22

D. 29

XOR Operator (^)

If both side bit is opposite result will be On

a	b
0	0
0	1
1	0
1	1

Steps to solve:-

- a = 12 (find binary form:1100)
- b = 25 (find binary form:11001)

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
		1	0	1	0	1	21

a ^ b= 01100

(12)

11001 (25)

10101 (21)

Ans.

Left Shift Operator(<<)

a=10 (1010)

a<<1 1010.0

10100(20) Ans.

a<<2

1010.00

101000(40) Ans.

Right Shift Operator(>>)

a=10

(1010)

a>>1

1010.

101(5) Ans.

a<<2

1010.

10(2) Ans.

What will be output? a=15;

print((a>>1))

Options:

A. 5

B. 6

C. 7

D. 8

Any Query?