Data Types

Numbers

There are three numeric types in Python:

- int
- float
- Complex : a + bJ, The real part of the number is a, and the imaginary part is b. Complex numbers are not used much in Python programming

Examples:

```
x = 1    # int
y = 2.8  # float
z = 1j   # complex
```

Type Conversion

```
x=1
y=2.8
z=1j
a=float(x)
b=int(y)
c=complex(x)
print(a)
print(b)
print(c)

print(type(a))
print(type(b))
print(type(c))
```

int(x)
long(x)
float(x)
complex(x)

Mathematical Functions

- abs()
- ceil()
- max(x1,x2,.....)
- min(x1,x2,.....)
- pow(x,y)
- sqrt()
- round()

Random Number

Example:

```
import random

print(random.randrange(1,10))
```

string

- Example: var1= 'Hello World!‘
- **Accessing Values in Strings:**

```
var1 = 'Hello World!'
var2 = "Python Programming"
print ("var1[0]: ", var1[0])
print ("var2[1:5]: ", var2[1:5])
```

Updating String

```
var1 = 'Hello World!'
print ("Updated String :- ", var1[:6] +
   'Python')
```

- String Special Operators

Operator	Description	Example
+	Concatenation - Adds values on either side of the operator	a + b will give HelloPython
*	Repetition - Creates new strings, concatenating multiple copies of the same string	a*2 will give - HelloHello
[]	Slice - Gives the character from the given index	a[1] will give e
[:]	Range Slice - Gives the characters from the given range	a[1:4] will give ell
in	Membership - Returns true if a character exists in the given string	H in a will give 1
not in	Membership - Returns true if a character does not exist in the given string	M not in a will give 1

- **String Formatting Operator**

Example:

```
print ("My name is %s and weight is
   %d kg!" % ('Zara', 21))
```

```
My name is Zara and weight is 21 kg!
```

Format Symbol	Conversion
%c	character
%s	string conversion via str() prior to formatting
%i	signed decimal integer
%d	signed decimal integer
%u	unsigned decimal integer
%o	octal integer
%x	hexadecimal integer (lowercase letters)
%X	hexadecimal integer (UPPERcase letters)
%e	exponential notation (with lowercase 'e')
%f	floating point real number

String Function

- **capitalize() -** str.capitalize()

Example

```
str = "this is string example..!!!"
print (str.capitalize())
```

Output

```
This is string example..!!!
```

- **center() :** str.center(width[, fillchar])

Example:

```
str = "this is string example....wow!!!"
print (str.center(40, 'a'))
```

Output

```
aaaathis is string example....wow!!!
   aaaa
```

- **count() -** str.count(sub, start= 0,end=len(string))
- Example:

```
str="this is string example....wow!!!"
sub='i'
print ("str.count('i') : ", str.count(sub))
sub='exam'
print ("str.count('exam', 10, 40) : ",
  str.count(sub,10,40))
```

Output:

```
str.count('i') : 3
str.count('exam', 4, 40) :
```

- **find() -** str.find(str)

Example:

```
str1 = "this is string
   example....wow!!!"
str2 = "exam";
print (str1.find(str2))
```

Output:

15

- **isalnum() –** str.isalnum()

- Example:

```
str = "this2016" # No space in this string
print (str.isalnum())
str = "this is string example....wow!!!"
print (str.isalnum())
```

- Output

```
True
False
```

- **isalpha() -** str.isalpha()

- Example:

```
str = "this"; # No space & digit in this string
print (str.isalpha())
str = "this is string example....wow!!!"
print (str.isalpha())
```

- Output:

```
True
False
```

- **isdigit() -** str.isdigit()

- Example:

```
str = "123456"; # Only digit in this string
print (str.isdigit())
str = "this is string example....wow!!!"
print (str.isdigit())
```

- Output

```
True
False
```

- **islower() –** str.islower()

Example:

```
str = "THIS is string example....wow!!!"
print (str.islower())
str = "this is string example....wow!!!"
print (str.islower())
```

Output:

```
False
True
```

- **isupper()**

- **isnumeric() –** str.isnumeric()

```
str = "this2016"
print (str.isnumeric())
str = "23443434“
print (str.isnumeric())
```

- Output

```
False
True
```

- **isspace() –** str.isspace()

```
str = " "
print (str.isspace())
str = "This is string example....wow!!!"
print (str.isspace())
```

Output:

True

False

- **join() –** str.join(sequence)
- Example:

```
s = "-"
seq = ("a", "b", "c") # This is sequence of
   strings.
print (s.join( seq ))
```

- Output

```
a-b-c
```

- **len() -** len(str)
- Example:

```
str = "this is string example....wow!!!"
print ("Length of the string: ", len(str))
```

- Output:

```
Length of the string: 32
```

- **lower() -** str.lower()

Example:

```
str = "THIS IS STRING EXAMPLE....WOW!!!"
print (str.lower())
```

- Output

```
this is string example....wow!!!
```

- **upper()**

- **replace() –** str.replace(old, new[, max])
- Example:

```
str = "this is string example....wow!!! this is really
   string"
print (str.replace("is", "was"))
print (str.replace("is", "was", 3))
```

- Output:

```
thwas was string example....wow!!! thwas was really
   string
thwas was string example....wow!!! thwas is really
   string
```

- **split() –** str.split()
- Example:

```
str = "this is string example....wow!!!"
print (str.split( ))
```

- Output:

```
['this', 'is', 'string', 'example....wow!!!']
```

- **swapcase() -** str.swapcase();
- Example:

```
str = "this is string example....wow!!!"
print (str.swapcase())
str = "This Is String Example....WOW!!!"
print (str.swapcase())
```

- Output

```
THIS IS STRING EXAMPLE....WOW!!!
tHIS iS sTRING eXAMPLE....wow!!!
```

List

- List in python is implemented to store the sequence of various type of data
- A list can be defined as a collection of values or items of different types
- The items in the list are separated with the comma (,) and enclosed with the square brackets []

List

- Example:

List1=[1,2,3,4,5]

list2= ['physics', 'chemistry', 1997, 2000]

list3 = [1, 2, 3, 4, 5]

list4 = ["a", "b", "c", "d"]

List5=[1,”a”,2,”abc”,3]

- **Accessing Values in Lists:**

Example:

```
list1 = ['physics', 'chemistry', 1997, 2000]
list2 = [1, 2, 3, 4, 5, 6, 7 ]
print ("list1[0]: ", list1[0])
print ("list2[1:5]: ", list2[1:5])
```

- **Output:**

```
list1[0]: physics
list2[1:5]: [2, 3, 4, 5]
```

User Input :

```
list=[]
number=int(input("how many value you want in
    a list: "))
for i in range(0,number):
    numbers = int(input("enter your choice
    number:"))
    list.append(numbers)
print(list)
```

- **Updating Lists**

```
list = ['physics', 'chemistry', 1997,
   2000]
print ("Value available at index 2 : ",
   list[2])
list[2] = 2001
print ("New value available at index 2 :
   ", list[2])
```

- **Delete List Elements**

```
list = ['physics', 'chemistry', 1997,
   2000]
print (list)
del list[2]
print ("After deleting value at index 2 :
   ", list)
```

Python Expression	Results	Description
len([1, 2, 3])	3	Length
[1, 2, 3] + [4, 5, 6]	[1, 2, 3, 4, 5, 6]	Concatenation
['Hi!'] * 4	['Hi!', 'Hi!', 'Hi!', 'Hi!']	Repetition
3 in [1, 2, 3]	True	Membership
for x in [1,2,3] : print (x,end=' ')	1 2 3	Iteration

Built-in List Functions

- **len() –** len(list)

- Example

```
list1 = ['physics', 'chemistry', 'maths']
print (len(list1))
```

- Output

```
3
```

- min(list)
- max(list)

Example:

```
list = [1,2,3,4,5,6,7]

print(max(list))
print(min(list))
```

- **append() –** list.append(obj)

- **Example:**

```
list1 = ['C++', 'Java', 'Python']
list1.append('C#')
print("updated list : ", list1)
```

- **Output:**

```
updated list : ['C++', 'Java', 'Python', 'C#']
```

- You can append another list also:

```
list1=[1,2,3,4]
list2=[6,”abc”]
list1.append(list2)
print(list1)
```

- **count() -** list.count(obj)

- Example:

```
aList = [123, 'xyz', 'zara', 'abc', 123];
print("Count for 123 : ", aList.count(123))
print("Count for zara : ", aList.count('zara'))
```

- Output:

```
Count for 123 : 2
Count for zara : 1
```

- **insert() –** list.insert(index, obj)

- Example:

```
list1 = ['physics', 'chemistry', 'maths']
list1.insert(1, 'Biology')
print('Final list : ', list1)
```

Output:

```
Final list : ['physics', 'Biology', 'chemistry',
   'maths']
```

- **pop() –** list.pop()

- **Example:**

```
list1 = ['physics', 'Biology', 'chemistry', 'maths']
list1.pop()
print(list1)
list1.pop(1)   # index
print(list1)
```

- **Output:**

```
['physics', 'Biology', 'chemistry']
['physics', 'chemistry']
```

- **remove() -** list1.remove(‘str’)

- **Example:**

```
list1 = ['physics', 'Biology', 'chemistry', 'maths']
list1.remove('Biology')
print (list1)
list1.remove('maths')
print (list1)
```

- **Output**

```
['physics', 'chemistry', 'maths']
['physics', 'chemistry']
```

- **reverse() – list.reverse()**

- **Example:**

```
list1 = ['physics', 'Biology', 'chemistry',
   'maths']
list1.reverse()
print (list1)
```

- **Output:**

['maths', 'chemistry', 'Biology', 'physics']

- **sort()**

- **Example:**

```
list1 = ['physics', 'Biology', 'chemistry', 'maths']
list1.sort()
print (list1)
```

- **Output:**

```
['Biology', 'chemistry', 'maths', 'physics']
```

Tuples

- A tuple is a sequence of immutable Python objects
- Tuples are sequences, just like lists
- The main difference between the tuples and the lists is that the tuples cannot be changed unlike lists
- Tuples use parentheses, whereas lists use square brackets

Example

```
tup1= ('physics', 'chemistry', 1997,
   2000)
tup2 = (1, 2, 3, 4, 5 )
tup3 = "a", "b", "c", "d"


print(tup1)
print(tup2)
print(tup3)
```

Accessing Values in Tuples

```
tup1 = ('physics', 'chemistry', 1997, 2000)
tup2 = (1, 2, 3, 4, 5, 6, 7 )


print (tup1[0])

print (tup2[1:5])
```

Updating Tuples

- Tuples are immutable

```
tup1 = (12, 34.56)

tup1[0] = 100

print (tup1)
```

Delete Tuple Elements

```
tup = ('physics', 'chemistry', 1997, 2000);

print(tup)


del tup[3];  # del tup - it will delete tup

print(tup)
```

Basic Tuples Operations

Python Expression	Results	Description
len((1, 2, 3))	3	Length
(1, 2, 3) + (4, 5, 6)	(1, 2, 3, 4, 5, 6)	Concatenation
('Hi!',) * 4	('Hi!', 'Hi!', 'Hi!', 'Hi!')	Repetition
3 in (1, 2, 3)	True	Membership
for x in (1,2,3) : print (x, end=' ')	1 2 3	Iteration

```
tp1=(1,2,3,4,5,6)
print(len(tp1))
print( 3 in tp1)
for x in tp1:
   print(x)
```

- **max()** – max(tuple)

- **Example**

```
tuple1, tuple2 = ('maths', 'che', 'phy', 'bio'), (456, 700,
  200)
print (max(tuple1))
print (max(tuple2))
```

- **Output**

phy
700

- **min() – min(tuple)**

- **Example:**

```
tuple1, tuple2 = ('maths', 'che', 'phy', 'bio'), (456, 700,
   200)
print (min(tuple1))

print (min(tuple2))
```

- **Output**

```
bio
200
```

- count()

- **Example:**

```
t1=(1,2,3,4,5,1,3,1)

print(t1.count(1))
```

- **Output:**

3

tuple() Method

- converts a list of items into tuples
- Syntax – tuple(seq)
- **Example**

```
list1= ['maths', 'che', 'phy', 'bio']
tuple1=tuple(list1)

print (tuple1)
```

- **Output**

```
('maths', 'che', 'phy', 'bio')
```

Dictionary

Dictionary

- A dictionary is a collection which is unordered, changeable and indexed
- In Python dictionaries are written with curly brackets, and they have keys and values.
- Example:

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
print(dict)
```

```
dict = {
   'Name': 'Zara',
   'Age': 7,
   'Class': 'First'
}

print(dict)
```

Accessing Values in Dictionary

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}


print (dict['Name'])
print (dict['Age'])
```

Updating Dictionary

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}

dict['Age'] = 8;

print (dict)
```

Delete Dictionary Elements

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}

del dict['Name']

print (dict)
```

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}

del dict         #delete entire dictionary

print (dict)
```

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}

dict.clear()   # remove all entries in
   dict

print (dict)
```

len()

```
dict = {'Name': 'Zara', 'Age': 7, 'Class':
   'First'}

print(len(dict))
```

copy()

```
dict1 = {'Name': 'Zara', 'Age': 7,
   'Class': 'First'}

dict2=dict1.copy()

print(dict2)
```

Only key

```
dict ={
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
for x in dict:
  print(x)
```

Only values

```
dict ={
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
for x in dict.values():
  print(x)
```

Both key and value

```
dict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
for x, y in dict.items():
  print(x, y)
```