In [1]:

```
class Enrollment:
    Name = ''
    DOB = ''
    Place = ''
    def f1(self,ename,edob,eplace):
        self.Name = ename
        self.DOB = edob
        self.Place=  eplace
        print(f'Emp {self.Name} Enrollment is done')
    def f2(self):
        print(f'Emp name:{self.Name} DOB:{self.DOB} Place:{self.Place}')
    def f3(self,eplace):
        self.Place = eplace
        print('Updated Info:')
        self.f2()
```

In [3]:

```
eobj = Enrollment()
eobj.f1('Arun','1st Jan','City-1') # 1st
eobj.f2()
eobj.f3('Bglore')
```

```
Emp Arun Enrollment is done
Emp name:Arun DOB:1st Jan Place:City-1
Updated Info:
Emp name:Arun DOB:1st Jan Place:Bglore
```

In [5]:

```
eobj1 = Enrollment()
eobj1.f2()
```

```
Emp name: DOB: Place:
```

In [ ]:

```
class DBI:
    def connect(self.....):
        '''estable db connection'''
    def method1(self):
        '''Query1'''
    def method2(self):
        '''Query2'''
Obj = DBI()
Obj.connect()
Obj.method1()
Vs

Obj = DBI()
Obj.method1() # OOPs View ->ValidCall But Operation(DB view) - FailedOperation
```

In [ ]:

```
In python Special attributes (or) magic methods (or) dunder methods
--------------------------------------------------------------------
 |-> pre-defined attributes ==>  __<attributeName>__
                                start and end with double underscores
 |-> these special attributes are called automatically invoked by python
```

In [ ]:

```

```

In [6]:

```
class Enrollment:
    def __init__(self,ename,edob,eplace):
        self.Name = ename
        self.DOB = edob
        self.Place=  eplace
        print(f'Emp {self.Name} Enrollment is done')
    def f2(self):
        print(f'Emp name:{self.Name} DOB:{self.DOB} Place:{self.Place}')
    def f3(self,eplace):
        self.Place = eplace
        print('Updated Info:')
        self.f2()

eobj1 = Enrollment('Arun','1st Jan','City-1')
eobj1.f2()
eobj1.f3('Bglore')
```

```
Emp Arun Enrollment is done
Emp name:Arun DOB:1st Jan Place:City-1
Updated Info:
Emp name:Arun DOB:1st Jan Place:Bglore
```

In [7]:

```
eobj2 = Enrollment('Tom','2nd Feb','City-A')
```

```
Emp Tom Enrollment is done
```

In [8]:

```
eobj2.f2()
```

```
Emp name:Tom DOB:2nd Feb Place:City-A
```

In [9]:

```
eobj2.f3('Pune')
```

```
Updated Info:
Emp name:Tom DOB:2nd Feb Place:Pune
```

In [ ]:

```
>>> class Box:
...     def f1(self):
...             print("This is non-constructor block")
...
>>> Box()
<__main__.Box object at 0x0000026DF77E2350>
>>> obj= Box()
>>>
>>> obj.f1()
This is non-constructor block
>>>
>>> class Box:
...     def __init__(self):
...             print("This is construtor block")
...
>>> Box()
This is construtor block
<__main__.Box object at 0x0000026DF77E1E70>
>>>
>>> obj = Box()
This is construtor block
>>>
>>>
>>> class Box:
...     def __init__(self,a1,a2=0,*a3,**a4):
...             print(a1,a2,a3,a4)
...             print('initialization is done')
...
>>> objX = Box()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
```

```
TypeError: Box.__init__() missing 1 required positional argument: 'a1'
>>>
>>> objX
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'objX' is not defined. Did you mean: 'obj'?
>>>
>>> objX = Box(10)
10 0 () {}
initialization is done
>>>
>>> objX = Box(10,20)
10 20 () {}
initialization is done
>>> objX = Box(10,20,30,40)
10 20 (30, 40) {}
initialization is done
>>> objX = Box(10,20,30,40,db='mysql',port=3303)
10 20 (30, 40) {'db': 'mysql', 'port': 3303}
initialization is done
>>>
>>>
>>>
>>>
>>>
>>> i=10
>>> type(i)
<class 'int'>
>>>
>>> int()
0
>>> # class int:
>>> #    def __init__(self,a=0):
>>> #          self.a=a
>>> #
>>>
>>> int(10)
10
>>> i=10 # int(10)
>>> j=1.34  # float(1.34)
>>>
>>> s1=str('Hello')
>>> s1.upper()
'HELLO'
>>> s1.isupper()
False
>>> # class str:
>>> #    def __init__(self,a=''):
>>> #        self.a=a
>>> #        ..
>>> #    def upper(self):
>>> #        return self.a
>>> #
>>> #    def isupper(self):
>>> #          ...
>>> #          retrun True/False
>>>
>>> L=[] # empty list
>>>
>>> L1 = list() # empty list
>>> L1
[]
>>> i=5
>>> print(i)
5
>>> del(i)
>>>
>>> i=5
>>> print(i)
5
>>> class myclass:
...     def __init__(self):
...             print('Welcome')
...     def __del__(self):
...             print('Thank you')
...
>>> obj = myclass()
Welcome
>>>
>>> del(obj)
Thank you
>>>
>>> obj = myclass()
Welcome
>>> exit()
Thank you
```

In [11]:

```
class Emp:
    def __init__(self,ename,edob):
        self.ename = ename
        self.edob = edob
        print('Initialization is done')
    def __str__(self):
        return str(self.ename)

obj = Emp('Arun','1st Jan')
str(obj)
```

```
Initialization is done
```

Out[11]: 'Arun'

In [ ]:

```
object callable  ==> obj() <== Not obj.MethodCAll()
```

In [15]:

```
class box:
    def __init__(self):
        self.a = 10
    def __str__(self):
        return str(self.a)
    def __del__(self):
        del(self)

objX = box()
# objX() TypeError: 'box' object is not callable
```

In [16]:

```
callable(objX)
```

Out[16]: False

In [18]:

```
def fx():
    print("OK")

print(type(fx))
callable(fx)
```

```
<class 'function'>
```

Out[18]:

```
True
```

In [ ]:

```
fx - class
fx() - object_Callable
```

In [19]:

```
fx.__call__() # same as fx()
```

```
OK
```

In [20]:

```
class box:
    def __init__(self):
        self.a = 10
    def __str__(self):
        return str(self.a)
    def __del__(self):
        del(self)
    def __call__(self):
        pass

objX = box()
callable(objX)
```

Out[20]:

```
True
```

In [21]:

```
class Emp:
    def __init__(self,ename):
        self.ename = ename
    def __str__(self):
        return self.ename
    def __call__(self):
        return self.ename.upper()

obj = Emp('bibu')
obj() # callable
```

Out[21]:

```
'BIBU'
```

In [22]:

```
str(obj)
```

Out[22]:

```
'bibu'
```

In [ ]:

```
__add__ +
__sub__ -
__mul__ *
__truediv__  /
__floordiv__ //
__mod__  %
__pow__ **
```

In [23]:

```
va = 10
vb = 20
```

```
va.__add__(vb) # va+vb
```

```
Out[23]:  30
```

```
In [24]:  s1 = 'python'
          s2 = 'programming'
          s1.__add__(s2) # s1+s2
```

```
Out[24]:  'pythonprogramming'
```

```
In [26]:  va = 150
          va.__gt__(100) # va > 100
```

```
Out[26]:  True
```

```
In [27]:  va.__lt__(100) # va < 100
```

```
Out[27]:  False
```

```
In [28]:  class box:
              user_name = 'root'
              __password = 'abc@123'  # Inside the class any attribute starts with doubleu
                                      # attributes are user defined private attribute

          obj = box()
          obj.user_name
```

```
Out[28]:  'root'
```

```
In [29]:  obj.__password
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[29], line 1
----> 1 obj.__password

AttributeError: 'box' object has no attribute '__password'
```

```
In [30]:  class box:
              def __init__(self):
                  self.user_name = 'root'
                  self.__password = 'abc@123'
              def method1(self):
                  print(self.user_name,self.__password)

          obj = box()
          obj.method1()
```

```
root abc@123
```

```
In [31]:  obj.user_name
```

```
Out[31]:  'root'
```

```
In [32]:  obj.__password
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[32], line 1
----> 1 obj.__password

AttributeError: 'box' object has no attribute '__password'
```

In [33]:

```
class box:
    var = 120
    def fx(self):
        pass
box.__dict__
```

Out[33]:

```
mappingproxy({'__module__': '__main__',
              '__firstlineno__': 1,
              'var': 120,
              'fx': <function __main__.box.fx(self)>,
              '__static_attributes__': (),
              '__dict__': <attribute '__dict__' of 'box' objects>,
              '__weakref__': <attribute '__weakref__' of 'box' objects>,
              '__doc__': None})
```

In [35]:

```
class box:
    '''simple class docs'''
    var = 120
    __port=4567
    def fx(self):
        pass

box.__dict__
```

Out[35]:

```
mappingproxy({'__module__': '__main__',
              '__firstlineno__': 1,
              '__doc__': 'simple class docs',
              'var': 120,
              '_box__port': 4567,
              'fx': <function __main__.box.fx(self)>,
              '__static_attributes__': (),
              '__dict__': <attribute '__dict__' of 'box' objects>,
              '__weakref__': <attribute '__weakref__' of 'box' objects>})
```

In [37]:

```
class box:
        user_name = 'root'
        __password = 'abc@123'

box.__dict__
```

Out[37]:

```
mappingproxy({'__module__': '__main__',
              '__firstlineno__': 1,
              'user_name': 'root',
              '_box__password': 'abc@123',
              '__static_attributes__': (),
              '__dict__': <attribute '__dict__' of 'box' objects>,
              '__weakref__': <attribute '__weakref__' of 'box' objects>,
              '__doc__': None})
```

In [38]:

```
obj = box()
obj._box__password
```

Out[38]:

```
'abc@123'
```

In [ ]:

```
# Inheritance
# parent <--> child relationship

class product:
    prod_id = 101
    prod_name = 'pA'

class customer:
    cname = 'cusA'

obj = customer()
obj.cname # OK
obj.prod_id # AttributeError

# ISA relationship
#
# parent
#   |
# child - we can invoke/access parent attributes

class parent:
    ...
class child(<parentclassName>): <== Single Inheritance
    ...
obj = child()
obj.<parentclass-attribute>
```

In [39]:

```
class product:
    prod_id = 101
    prod_name = 'pA'

class customer(product): # Inheritance  Vs  def functionName(argument):
    cname = 'cusA'

obj = customer()
print(obj.prod_id,obj.prod_name,obj.cname)
```

```
101 pA cusA
```

In [ ]:

```
File: fA.py               File: fB.py                       file: p1.py
==============            =============                     =============
class product:             import fA                          import fB
    prod_id = 101          class customer(fA.product):        obj = fB.customer()
    def fx(self):                def f1(self):                obj.fx()
        ...                         ...                       obj.fy()
    def fy(self):                def f2(self):                obj.f1()
        ...                          ...                      obj.f2()
===================      ===============                  ====================
```

In [44]:

```
class product:
    def __init__(self):
        print('initialization from productClass')
    def method1(self):
        self.pid = 101
        return self.pid

class customer(product):
    def method2(self):
        self.cusName = 'cusA'
        return self.cusName

obj = customer()
pID = obj.method1()
cus_name = obj.method2()
print(f'product ID:{pID} customer Name:{cus_name}')
```

```
initialization from productClass
product ID:101 customer Name:cusA
```

In [45]:

```
class product:
    def display(self):
        print('display product details')

class customer(product):
    def display(self):
        print('display customer records')

obj = customer()
obj.display()
obj.display()
```

```
display customer records
display customer records
```

In [46]:

```
class product:
    def display(self):
        print('display product details')

class customer(product):
    def display(self):
        print('display customer records')
        super().display() # invoke parent class display()

obj = customer()
obj.display()
```

```
display customer records
display product details
```

In [47]:

```
class product:
    def display(self):
        print('display product details')

class customer(product):
    def display(self):
        print('display customer records')
        product.display(self) # invoke parent class display() anotherway

obj = customer()
obj.display()
```

```
display customer records
display product details
```

In [ ]:

```
class A:
    pass
class B:
    pass
class C(A,B): # multiple inheritance
```

```
    pass
cObj = C()
#----------------------------
class A:
    pass
class B(A):
    pass
class C(B):   # multi-level inheritance
    pass
..
```

In [49]:

```
# python <= 2.6
# object parent of native class
# ------
class Box(object):
    bid = 101

Box.bid
obj = Box()
obj.bid

Box.__mro__
```

Out[49]:

```
(__main__.Box, object)
```

In [50]:

```
class mybox:
    pass
mybox.__mro__
```

Out[50]:

```
(__main__.mybox, object)
```

In [51]:

```
class product:
    pass

class customer(product):
    pass

customer.__mro__
```

Out[51]:

```
(__main__.customer, __main__.product, object)
```

In [53]:

```
import bs4
print(bs4.BeautifulSoup)
```

```
<class 'bs4.BeautifulSoup'>
```

In [54]:

```
print(bs4.BeautifulSoup.__doc__)
```

```
A data structure representing a parsed HTML or XML document.

Most of the methods you'll call on a BeautifulSoup object are inherited from
PageElement or Tag.

Internally, this class defines the basic interface called by the
tree builders when converting an HTML/XML document into a data
structure. The interface abstracts away the differences between
parsers. To write a new tree builder, you'll need to understand
these methods as a whole.

These methods will be called by the BeautifulSoup constructor:
  * reset()
  * feed(markup)

The tree builder may call these methods from its feed() implementation:
  * handle_starttag(name, attrs) # See note about return value
  * handle_endtag(name)
  * handle_data(data) # Appends to the current data node
  * endData(containerClass) # Ends the current data node

No matter how complicated the underlying parser is, you should be
able to build a tree using 'start tag' events, 'end tag' events,
'data' events, and "done with data" events.

If you encounter an empty-element tag (aka a self-closing tag,
like HTML's <br> tag), call handle_starttag and then
handle_endtag.
```

In [55]:

```python
bs4.BeautifulSoup.__mro__
```

Out[55]:

```
(bs4.BeautifulSoup, bs4.element.Tag, bs4.element.PageElement, object)
```

In [57]:

```python
print(bs4.element.Tag.__doc__)
```

```
Represents an HTML or XML tag that is part of a parse tree, along
with its attributes and contents.

When Beautiful Soup parses the markup <b>penguin</b>, it will
create a Tag object representing the <b> tag.
```

In [59]:

```python
import re

# re.search('pattern_string','input_string') -> ack_obj / None
re.search('sales','101,raj,sales,pune')
```

Out[59]:

```
<re.Match object; span=(8, 13), match='sales'>
```

In [60]:

```python
re.search('sales','101,raj,QA,pune')
```

In [62]:

```python
import sqlite3
#print(sqlite3.connect)
sqlite3.connect('t1.db')
```

Out[62]:

```
<sqlite3.Connection at 0x29bb1361f30>
```

In [63]:

```
def f1(a1):
    class myclass:
        def __init__(self,a1):
            self.a1 = a1
        def method1(self):
            return self.a1+100
        def method2(self):
            print('method2 block')
    obj = myclass(a1)
    return obj
```

In [64]:

```
f1(10)
```

Out[64]:

```
<__main__.f1.<locals>.myclass at 0x29bafca6f90>
```

In [65]:

```
myobj = f1(10)
myobj.method1()
```

Out[65]:

```
110
```

In [66]:

```
myobj.method2()
```

```
method2 block
```

In [ ]:

```
Functional style programming
------------------------------
```

In [67]:

```
L = list()

def f1(a):
    return a+100

for var in [10,20,30,40,50]:
    r = f1(var)
    L.append(r)

print(L)
```

```
[110, 120, 130, 140, 150]
```

In [68]:

```
list(map(lambda a:a+100,[10,20,30,40,50]))
```

Out[68]:

```
[110, 120, 130, 140, 150]
```

In [ ]:

```
1. lambda
2. list comprehension
3. generator
|
4. map,filter
------------------------
1. lambda - unnamed function
   -------
     |-> function Call with arguments and return some value
     |-> lambda <list of arguments>:<basic Operation>
                                    ------------------
```

In [69]:

```
def f1(a1,a2):
    return a1+a2
```

```
f1(10,20)
```

```
Out[69]: 30
```

```
In [71]: f2 = lambda a1,a2:a1+a2
         f2(10,20)
```

```
Out[71]: 30
```

```
In [72]: def f3(a1):
             return a1.upper()

         f3('abc')
```

```
Out[72]: 'ABC'
```

```
In [73]: f4 = lambda a1:a1.upper()
         f4('abc')
```

```
Out[73]: 'ABC'
```

```
In [74]: f5 = lambda a1,a2: a1 >a2
         f5(100,20)
```

```
Out[74]: True
```

```
In [75]: f5(100,120)
```

```
Out[75]: False
```

```
In [76]: def fx(a1):
             return 'sales' in a1
         fx('101,raj,sales,pune')
```

```
Out[76]: True
```

```
In [77]: fy = lambda a1: 'sales' in a1
         fy('101,raj,sales,pune')
```

```
Out[77]: True
```

```
In [78]: fy('101,raj,QA,pune')
```

```
Out[78]: False
```

```
In [79]: def fx(a):
             if(a >501 and a < 600):
                 return a+100
             elif(a >100 and a < 200):
                 return a+500
             else:
                 return a+1000
```

```
In [80]: fx(550)
```

```
Out[80]: 650
```

In [81]:
```
fx(150)
```

Out[81]:
```
650
```

In [82]:
```
fx(10)
```

Out[82]:
```
1010
```

In [83]:
```
f6 = lambda a:fx(a)
f6(100)
```

Out[83]:
```
1100
```

In [84]:
```
f6(550)
```

Out[84]:
```
650
```

In [85]:
```
f6(150)
```

Out[85]:
```
650
```

In [88]:
```
# 2. list comprehension - list append operation
#    ==================
#      [final_value for iterable]
#                     -----(1)---
#      ----(2)-----
#    --------------------------------------

L = list()
for var in [10,20,30,40,50]:
    r = var+100
    L.append(r)
print(L,"\n")

[var+100 for var in [10,20,30,40,50]] # list comprehension
```

```
[110, 120, 130, 140, 150]
```

Out[88]:
```
[110, 120, 130, 140, 150]
```

In [90]:
```
# 2. list comprehension - list append operation
#    ==================
#      [final_value for iterable]
#                     -----(1)---
#      ----(2)-----
#    --------------------------------------

L = list()
for var in [10,20,30,40,50]:
    if(var >30):
        r = var+100
        L.append(r)
    else:
        r = var+500
        L.append(r)

print(L,"\n")
```

```
[var+100 if(var >30) else var+500 for var in [10,20,30,40,50]] # List comprehens
```

```
[510, 520, 530, 140, 150]
```

Out[90]:
```
[510, 520, 530, 140, 150]
```

In [91]:
```
d={}
d['counts'] = [var+100 if(var >30) else var+500 for var in [10,20,30,40,50]]
d
```

Out[91]:
```
{'counts': [510, 520, 530, 140, 150]}
```

In [ ]:
```
In Python - class - object design model
int - class
 |-> 10 - object -> 0x1234
 |-> 11 - object -> 0x3334
 |-> 12 - object -> 0x3413

str - class
 |-> 'a' - object -> 0x312
 |-> 'b' - object -> 0x315
...
...
```

In [95]:
```
a = 10
b = 10
c = 2+8
print(hex(id(a)))
print(hex(id(b)))
print(hex(id(c)))
print(hex(id(10)))
print('')
a=a+1
print(hex(id(a)))
print(hex(id(11)))
```

```
0x7ffde15cb4c8
0x7ffde15cb4c8
0x7ffde15cb4c8
0x7ffde15cb4c8

0x7ffde15cb4e8
0x7ffde15cb4e8
```

In [96]:
```
s = 'abcab'    #   | a | b | c | a | b |
               #   | 0 | 1 | 2 | 3 | 4 | <== index
               #   |0x1 0x4 0x6 0x1 0x4
print(hex(id(s[0])))
print(hex(id(s[3])))
print(hex(id('a')))
```

```
0x7ffde15dafe0
0x7ffde15dafe0
0x7ffde15dafe0
```

In [97]:
```
iter(s)
 |->iterate this object
 |-> 1. next() - iterated one element at a time....StopIteration
 |-> 2. for loop
 |-> 3. typecast to list
```

Out[97]:
```
<str_ascii_iterator at 0x29bb13c3df0>
```

In [98]:
```
s='abcd'
obj = iter(s)
print(next(obj))
print(next(obj))
print(next(obj))
print(next(obj))
print(next(obj))
```

```
a
b
c
d
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[98], line 7
      5 print(next(obj))
      6 print(next(obj))
----> 7 print(next(obj))

StopIteration: 
```

In [99]:
```
s='abcd'
obj = iter(s)
for var in obj:
    print(var)
```

```
a
b
c
d
```

In [100…
```
s='abcd'
obj = iter(s)
list(obj)
```

Out[100…
```
['a', 'b', 'c', 'd']
```

In [ ]:
```
# Generator
# ------------
#  |->function returns an address(iterator)
#                ----------------------------//generator
#                   yield <Value>/Expression
```

In [103…
```
def f1():
    return 10
def f2():
    yield 10

print(type(f1),type(f2))
print(type(f1()),type(f2()))
```

```
<class 'function'> <class 'function'>
<class 'int'> <class 'generator'>
```

In [105…

```
def f1():
    return 10
    print('this line never execute')

def f2():
    yield 10
    print('this line will execute')
    yield 20
    print('this line also will execute')
    yield 'd1','d2'

f1()
f2()
```

Out[105…

```
<generator object f2 at 0x0000029BB130B640>
```

In [109…

```
gobj = f2()
print(next(gobj))
print(next(gobj))
print(next(gobj))
print(next(gobj))
```

```
10
this line will execute
20
this line also will execute
('d1', 'd2')
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[109], line 5
      3 print(next(gobj))
      4 print(next(gobj))
----> 5 print(next(gobj))

StopIteration: 
```

In [110…

```
def f2():
    yield 10
    print('this line will execute')
    yield 20
    print('this line also will execute')
    yield 'd1','d2'

gobj = f2()
for var in gobj:
    print(var)
```

```
10
this line will execute
20
this line also will execute
('d1', 'd2')
```

In [111…

```
gobj = f2()
list(gobj)
```

```
this line will execute
this line also will execute
```

Out[111…

```
[10, 20, ('d1', 'd2')]
```

In [ ]:

```
i=10   # mapping to an existing object - initialization -> i - label (or) placeh
j=10   # mapping to an existing object - initialization -> j - label (or) placeh
```

In [ ]:

```
karth@paka MINGW64 ~
$ ls Demo1
D1/  p1.log  p2.log

karth@paka MINGW64 ~
$ ls -R Demo1
Demo1:
D1/  p1.log  p2.log

Demo1/D1:
D2/  t1.txt

Demo1/D1/D2:
D3/

Demo1/D1/D2/D3:
r1.log  r2.log

karth@paka MINGW64 ~
$
C:\Users\karth>python
Python 3.10.0 (tags/v3.10.0:b494f59, Oct  4 2021, 19:00:18) [MSC v.1929 64 bit (
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>>
>>> os.listdir("Demo1")
['D1', 'p1.log', 'p2.log']
>>>
>>> os.walk("Demo1")
<generator object _walk at 0x000001C877539A80>
>>> gobj = os.walk("Demo1")
>>> for var in gobj:
...     print(var)
...
('Demo1', ['D1'], ['p1.log', 'p2.log'])
('Demo1\\D1', ['D2'], ['t1.txt'])
('Demo1\\D1\\D2', ['D3'], [])
('Demo1\\D1\\D2\\D3', [], ['r1.log', 'r2.log'])
>>>
>>> def mywalk():
...     yield "Demo1",["D1"],["p1.log","p2.log"]
...     yield "Demo1\D1",["D2"],["t1.txt"]
...     yield "Demo1\D1\D2",["D3"],[]
...     yield "Demo1\D1\D2\D3",[],['r1.log','r2.log']
...
>>> mywalk()
<generator object mywalk at 0x000001C877626030>
>>>
>>> gobj = mywalk()
>>> for var in gobj:
...     print(var)
...
('Demo1', ['D1'], ['p1.log', 'p2.log'])
('Demo1\\D1', ['D2'], ['t1.txt'])
('Demo1\\D1\\D2', ['D3'], [])
```

```
('Demo1\\D1\\D2\\D3', [], ['r1.log', 'r2.log'])
>>>
```

In [ ]:

```
map() => map(function,<collection/iterable>) --> generator
                |               |                ============
              lambda        list comprehension/collection
```

In [ ]:

```
L = list()
--------------------
def f1(a):
    return a+100
-----------------------  // lambda
for var in [10,20,30,40,50]:
    r = f1(var)
    L.append(r)

print(L)
```

In [112…

```
map(lambda a:a+100,[10,20,30,40,50])
```

Out[112…

```
<map at 0x29bb14111e0>
```

In [114…

```
map_obj = map(lambda a:a+100,[10,20,30,40,50])
print(next(map_obj))
print(next(map_obj))
print(next(map_obj))
print(next(map_obj))
print(next(map_obj))
print(next(map_obj))
```

```
110
120
130
140
150
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[114], line 7
      5 print(next(map_obj))
      6 print(next(map_obj))
----> 7 print(next(map_obj))

StopIteration:
```

In [115…

```
map_obj = map(lambda a:a+100,[10,20,30,40,50])
for var in map_obj:
    print(var)
```

```
110
120
130
140
150
```

In [116…

```
map_obj = map(lambda a:a+100,[10,20,30,40,50])
list(map_obj)
```

Out[116…

```
[110, 120, 130, 140, 150]
```

In [117…

```
open('emp.csv','r')
```

Out[117…

```
<_io.TextIOWrapper name='emp.csv' mode='r' encoding='cp1252'>
```

In [118…

```
fobj = open('emp.csv','r')
print(next(fobj))
print(next(fobj))
```

```
eid,ename,edept,ecity,ecost

101,raj,sales,pune,1000

```

In [120…

```
fobj = open('emp.csv','r')
for var in fobj:
    print(var.strip())
```

```
eid,ename,edept,ecity,ecost
101,raj,sales,pune,1000
102,leo,prod,bglore,2301
230,raj,prod,pune,2300
450,shan,sales,bglore,3401
542,anu,HR,mumbai,4590
321,bibu,sales,hyd,5419
651,ram,hr,bglore,3130
541,leo,admin,chennai,4913
652,karthik,sales,bglore,3405
```

In [121…

```
fobj = open('emp.csv','r')
list(fobj)
```

Out[121…

```
['eid,ename,edept,ecity,ecost\n',
 '101,raj,sales,pune,1000\n',
 '102,leo,prod,bglore,2301\n',
 '230,raj,prod,pune,2300\n',
 '450,shan,sales,bglore,3401\n',
 '542,anu,HR,mumbai,4590\n',
 '321,bibu,sales,hyd,5419\n',
 '651,ram,hr,bglore,3130\n',
 '541,leo,admin,chennai,4913\n',
 '652,karthik,sales,bglore,3405\n']
```

In [122…

```
list(map(lambda a:a.upper(),open('emp.csv')))
```

Out[122…

```
['EID,ENAME,EDEPT,ECITY,ECOST\n',
 '101,RAJ,SALES,PUNE,1000\n',
 '102,LEO,PROD,BGLORE,2301\n',
 '230,RAJ,PROD,PUNE,2300\n',
 '450,SHAN,SALES,BGLORE,3401\n',
 '542,ANU,HR,MUMBAI,4590\n',
 '321,BIBU,SALES,HYD,5419\n',
 '651,RAM,HR,BGLORE,3130\n',
 '541,LEO,ADMIN,CHENNAI,4913\n',
 '652,KARTHIK,SALES,BGLORE,3405\n']
```

In [123…

```
list(map(lambda a:a.title(),open('emp.csv')))
```

Out[123…

```
['Eid,Ename,Edept,Ecity,Ecost\n',
 '101,Raj,Sales,Pune,1000\n',
 '102,Leo,Prod,Bglore,2301\n',
 '230,Raj,Prod,Pune,2300\n',
 '450,Shan,Sales,Bglore,3401\n',
 '542,Anu,Hr,Mumbai,4590\n',
 '321,Bibu,Sales,Hyd,5419\n',
 '651,Ram,Hr,Bglore,3130\n',
 '541,Leo,Admin,Chennai,4913\n',
 '652,Karthik,Sales,Bglore,3405\n']
```

In [124…

```
d={}
d['CSV'] = list(map(lambda a:a.title(),open('emp.csv')))
d
```

Out[124…

```
{'CSV': ['Eid,Ename,Edept,Ecity,Ecost\n',
  '101,Raj,Sales,Pune,1000\n',
  '102,Leo,Prod,Bglore,2301\n',
  '230,Raj,Prod,Pune,2300\n',
  '450,Shan,Sales,Bglore,3401\n',
  '542,Anu,Hr,Mumbai,4590\n',
  '321,Bibu,Sales,Hyd,5419\n',
  '651,Ram,Hr,Bglore,3130\n',
  '541,Leo,Admin,Chennai,4913\n',
  '652,Karthik,Sales,Bglore,3405\n']}
```

In [125…

```
import json
jd = json.dumps(d,indent=2)
```

In [126…

```
json.loads(jd) # convert to python
```

Out[126…

```
{'CSV': ['Eid,Ename,Edept,Ecity,Ecost\n',
  '101,Raj,Sales,Pune,1000\n',
  '102,Leo,Prod,Bglore,2301\n',
  '230,Raj,Prod,Pune,2300\n',
  '450,Shan,Sales,Bglore,3401\n',
  '542,Anu,Hr,Mumbai,4590\n',
  '321,Bibu,Sales,Hyd,5419\n',
  '651,Ram,Hr,Bglore,3130\n',
  '541,Leo,Admin,Chennai,4913\n',
  '652,Karthik,Sales,Bglore,3405\n']}
```

In [127…

```
def f1(a):
    if(a >30):
        return True
    else:
        return False

L = list()
for var in [10,20,30,40,50,60,25,15]:
    r = f1(var)
    L.append(r)
L
```

Out[127…

```
[False, False, False, True, True, True, False, False]
```

In [128…

```
list(map(lambda a: a>30,[10,20,30,40,50,60,25,15]))
```

Out[128…

```
[False, False, False, True, True, True, False, False]
```

In [129…

```
list(filter(lambda a: a>30,[10,20,30,40,50,60,25,15]))
```

Out[129…

```
[40, 50, 60]
```

In [ ]:

```
map(function,collection) ->Generator
filter(function,collection) ->Generator
```

In [133…

```
# help(enumerate)
for var in enumerate('abcd'):
    print(var)
```

```
(0, 'a')
(1, 'b')
(2, 'c')
(3, 'd')
```

In [134…

```
for var in enumerate('abcd',5):
    print(var)
```

```
(5, 'a')
(6, 'b')
(7, 'c')
(8, 'd')
```

In [135…

```
for var in enumerate(open('emp.csv'),1):
    print(var)
```

```
(1, 'eid,ename,edept,ecity,ecost\n')
(2, '101,raj,sales,pune,1000\n')
(3, '102,leo,prod,bglore,2301\n')
(4, '230,raj,prod,pune,2300\n')
(5, '450,shan,sales,bglore,3401\n')
(6, '542,anu,HR,mumbai,4590\n')
(7, '321,bibu,sales,hyd,5419\n')
(8, '651,ram,hr,bglore,3130\n')
(9, '541,leo,admin,chennai,4913\n')
(10, '652,karthik,sales,bglore,3405\n')
```

In [136…

```
for line,data in enumerate(open('emp.csv'),1):
    print(f'{line} - {data.strip()}')
```

```
1 - eid,ename,edept,ecity,ecost
2 - 101,raj,sales,pune,1000
3 - 102,leo,prod,bglore,2301
4 - 230,raj,prod,pune,2300
5 - 450,shan,sales,bglore,3401
6 - 542,anu,HR,mumbai,4590
7 - 321,bibu,sales,hyd,5419
8 - 651,ram,hr,bglore,3130
9 - 541,leo,admin,chennai,4913
10 - 652,karthik,sales,bglore,3405
```

In [138…

```
for line,data in enumerate(open('emp.csv'),1):
    if(line >=5 and line <=8):
        print(f'{line} - {data.strip()}')
```

```
5 - 450,shan,sales,bglore,3401
6 - 542,anu,HR,mumbai,4590
7 - 321,bibu,sales,hyd,5419
8 - 651,ram,hr,bglore,3130
```

In [139…

```
for line,data in enumerate(open('emp.csv'),1):
    if not(line >=5 and line <=8):
        print(f'{line} - {data.strip()}')
```

```
1 - eid,ename,edept,ecity,ecost
2 - 101,raj,sales,pune,1000
3 - 102,leo,prod,bglore,2301
4 - 230,raj,prod,pune,2300
9 - 541,leo,admin,chennai,4913
10 - 652,karthik,sales,bglore,3405
```

In [146…

```
[[line,data.strip()] for line,data in enumerate(open('emp.csv'),1)]
```

Out[146…

```
[[1, 'eid,ename,edept,ecity,ecost'],
 [2, '101,raj,sales,pune,1000'],
 [3, '102,leo,prod,bglore,2301'],
 [4, '230,raj,prod,pune,2300'],
 [5, '450,shan,sales,bglore,3401'],
 [6, '542,anu,HR,mumbai,4590'],
 [7, '321,bibu,sales,hyd,5419'],
 [8, '651,ram,hr,bglore,3130'],
 [9, '541,leo,admin,chennai,4913'],
 [10, '652,karthik,sales,bglore,3405']]
```

In [147…

```
[[line,data.strip()] for line,data in enumerate(open('emp.csv'),1) if(line >=5 a
```

Out[147…

```
[[5, '450,shan,sales,bglore,3401'],
 [6, '542,anu,HR,mumbai,4590'],
 [7, '321,bibu,sales,hyd,5419']]
```

In [ ]:

```
Iterator - Address - object ->traverse through elements one at a time

s = 'abcd'     | [a] [b] [c] [d] | <--
               | 0x1 0x2 0x3 0x4     |
               |-----------------|   |
               |     0x6         | <==
```

In [148…

```
s='abcd'
iter(s) # convert string into iterator
```

Out[148…

```
<str_ascii_iterator at 0x29babcf2500>
```

In [149…

```
obj = iter(s)
for var in obj:
    print(var)
```

```
a
b
c
d
```

In [ ]:

```
__iter__()
__next__()
for loop --> internally use iter() and next() to loop
```

In [150…

```
'''custom iterator '''
class myclass:
    def __init__(self,n):
        self.n = n
        self.num = 1
    def __iter__(self):
        return self
    def __next__(self):
        if self.num <= self.n:
            val = self.num
            self.num += 1
            return val
        else:
            raise StopIteration
```

In [151…

```
myclass(5)
```

Out[151…

```
<__main__.myclass at 0x29bafca74d0>
```

In [152…

```
for var in myclass(5):
    print(var)
```

```
1
2
3
4
5
```

In [ ]:

```
Thread
Regx
Database
Webscraping
json
...
```

In [153…

```
fobj = open('emp.csv','r')
s = fobj.read()
fobj.close()

with open('emp.csv','r') as fobj:
    s = fobj.read()
    print(s)

# no need to write fobj.close()

with open('r1.log','w') as wobj:
    wobj.write('test-1\n')
    wobj.write('test-2\n')
```

```
eid,ename,edept,ecity,ecost
101,raj,sales,pune,1000
102,leo,prod,bglore,2301
230,raj,prod,pune,2300
450,shan,sales,bglore,3401
542,anu,HR,mumbai,4590
321,bibu,sales,hyd,5419
651,ram,hr,bglore,3130
541,leo,admin,chennai,4913
652,karthik,sales,bglore,3405
```

In [154…

```
open('r1.log').read()
```

```
Out[154…  'test-1\ntest-2\n'
```

```
In [ ]: File,Process
File - Data - Under the store device(HD)
Process - Data - Under the CPU
|
|-> [Data] + [Address] //process - new process - ProcessPID (import os ; os.getp
                                                 ParentProcessPID( os.getppid())

P1 - parent process - 101 - Wait  - 0x123
|
P2 - new process - 102 - Child Process (R+) <== PID: 102  PPID:101  - 0x345
                                        |__Exit ---->....

Thread - [Data] //execution - smallest unit of data(execution)
P1 [ t1 t2 t3 ]
      |__|__|
        0x123

In Python - GIL (Global Interpreter Lock)
CPython - GIL enabled
t1 - r+ -IO
t2 -r+
t1 t2
r+ r+

IPython GIL is disabled
Jython GIL is disabled
pip install ipython
ipython
[] ..
ipython ...
python.org -->Cpython - we can do python programming
              |-> MLlibs - no available by default => pip install numpy pip inst
                            pip install matplotlib  pip install scikitlearn
anaconda python ==> we can do python programming + MLlibs installed
GIL is disabled
--------------------------------------------------------
import threading
threading.Thread <== className
           |->constructor - keyword arguments (var=value)
                            |->target=<threadName>,args=(tuple_type)

threadObject = threading.Thread(target=f1,args=(10,20,30))
threadObject.start() # start thread

    [  ]  <== t1 t2 .. tn
     |---------------------
Synchronization

threading.Thread - class - creation
threading.Lock - class - synchronization
           |->obj.acquire()
           |->obj.release()
  <or>
  with Lock_object:      # only one thread update counter value at a time
      <critical section>

Lock_object = threading.Lock()
```

```
def f1():
    with Lock_object:
        Code block operation # critical section

for var in range(1,1000000):
    threading.Thread(target=f1).start()
-------------------------------------------------------
```

In [ ]:

```
Exception Handling
- Exception is logicalError - Exit state

try:
    <code-initialization>
except Exception as eobj:
    Handle the logical
else:
    There is no error
finally:
    <Always running>

Python
    'select *from table' //str
        |------------------------------------------------->DB(SQL) ....
                                                                   |
                   -------------<-----------------------------------------
                             python understanding format
```

In [155…

```
import sqlite3
sqlite3.connect('test1.db')
```

Out[155…

```
<sqlite3.Connection at 0x29babd1b1f0>
```

In [157…

```
conn = sqlite3.connect('test1.db')
sth = conn.cursor()
sth.execute('create table product(pid int,pname Text,pcost int)')
```

Out[157…

```
<sqlite3.Cursor at 0x29bb12d5140>
```

In [158…

```
sth.execute("insert into product(pid,pname,pcost) values(101,'pA',1000)")
```

Out[158…

```
<sqlite3.Cursor at 0x29bb12d5140>
```

In [159…

```
sth.execute("insert into product(pid,pname,pcost) values(102,'pB',2000)")
sth.execute("insert into product(pid,pname,pcost) values(103,'pC',3000)")
sth.execute("insert into product(pid,pname,pcost) values(104,'pD',4000)")
sth.execute("insert into product(pid,pname,pcost) values(105,'pE',5000)")
```

Out[159…

```
<sqlite3.Cursor at 0x29bb12d5140>
```

In [160…

```
va = 106
vb = 'pF'
vc = 3490
sth.execute("insert into product(pid,pname,pcost) values(?,?,?)",(va,vb,vc))
```

Out[160…

```
<sqlite3.Cursor at 0x29bb12d5140>
```

In [161…

```
sth.execute('select *from product')
```

Out[161…

```
<sqlite3.Cursor at 0x29bb12d5140>
```

In [162…

```
sth.fetchone()
```

Out[162…

```
(101, 'pA', 1000)
```

In [163…

```
sth.fetchone()
```

Out[163…

```
(102, 'pB', 2000)
```

In [164…

```
sth.fetchone()
```

Out[164…

```
(103, 'pC', 3000)
```

In [165…

```
sth.fetchone()
```

Out[165…

```
(104, 'pD', 4000)
```

In [166…

```
sth.fetchone()
```

Out[166…

```
(105, 'pE', 5000)
```

In [167…

```
sth.execute('select *from product')
sth.fetchall()
```

Out[167…

```
[(101, 'pA', 1000),
 (102, 'pB', 2000),
 (103, 'pC', 3000),
 (104, 'pD', 4000),
 (105, 'pE', 5000),
 (106, 'pF', 3490)]
```

In [168…

```
sth.execute('select *from product')
# apply generator
for var in sth:
    print(var)
```

```
(101, 'pA', 1000)
(102, 'pB', 2000)
(103, 'pC', 3000)
(104, 'pD', 4000)
(105, 'pE', 5000)
(106, 'pF', 3490)
```

In [169…

```
sth.execute('select *from product')
list(sth) # generator
```

Out[169…

```
[(101, 'pA', 1000),
 (102, 'pB', 2000),
 (103, 'pC', 3000),
 (104, 'pD', 4000),
 (105, 'pE', 5000),
 (106, 'pF', 3490)]
```

In [170…

```
conn.commit() # commit changes
```

In [171…

```
conn.close() # close the connection
```

In [172…

```
conn = sqlite3.connect('test1.db')
sth = conn.cursor()
sth.execute('select *from product')
list(sth)
```

Out[172…

```
[(101, 'pA', 1000),
 (102, 'pB', 2000),
 (103, 'pC', 3000),
 (104, 'pD', 4000),
 (105, 'pE', 5000),
 (106, 'pF', 3490)]
```

In [ ]:

```
# Requests module
# -->download webcontent
#                   |->response as webpage = <html>+{{data}}
#                   |->response as data <--- json
import requests
requests.get(URL) -> 200 # OK
r = requests.get(URL)
r.status_code -> 200
|
r.headers -> webHeader info - dict//
|
r.text --> Get the content
```

In [173…

```
URL1 = 'https://www.google.com'
URL2 = 'https://api.github.com/users/hadley/orgs'
```

In [174…

```
import requests
requests.get(URL1)
```

Out[174…

```
<Response [200]>
```

In [175…

```
requests.get(URL2)
```

Out[175…

```
<Response [200]>
```

In [177…

```
r1 = requests.get(URL1)
r2 = requests.get(URL2)
print(r1.status_code,r2.status_code)
```

```
200 200
```

In [179…

```
r1.headers
r1.headers['Content-Type']
```

Out[179…

```
'text/html; charset=ISO-8859-1'
```

In [180…

```
r2.headers['Content-Type']
```

Out[180…

```
'application/json; charset=utf-8'
```

In [181…

```
google_web_page = r1.text
print(type(google_web_page),len(google_web_page))
```

```
<class 'str'> 19196
```

In [182…

```
with open('test1.html','w') as wobj:
    wobj.write(google_web_page)

''' testing purpose '''
```

In [184…

```
#print(google_web_page)
```

In [ ]:

```
bs4 - module
 |-> web scraping
 |-> parse html / xml document and extract the data

bs4.BeautifulSoup - className
     |->Constructor - webpage  - default parser is html.parser

soupObject = bs4.BeautifulSoup(webpage)
soupObject.<htmlTagName> -> Value (ex: soupObject.p ->paragraphTag ;
                                       soupObject.h1 ->header1Tag ; )
 <or>
soupObject.find(<htmlTagName>) //same as soupObject.<htmlTagName>
  Vs
soupObject.find_all(<htmlTagName>) -> //list of all inputTag data
soupObject.get_text() - Extract all the text
```

In [186…

```
import requests
URL1 = 'https://www.google.com'
r1 = requests.get(URL1)
if(r1.status_code != 200):
    print(f'URL - {URL1} download is failed')
    exit()
r1.headers['Content-Type']
gpage = r1.text
```

In [187…

```
import bs4
soupObj = bs4.BeautifulSoup(gpage)
```

In [188…

```
soupObj.title
```

Out[188…

```
<title>Google</title>
```

In [189…

```
soupObj.p
```

Out[189…

```
<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/priv
acy/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>
```

In [190…

```
soupObj.h1
```

In [192…

```
head_info = soupObj.head
```

In [193…

```
soupObj.find('title') # soupObj.title
```

Out[193…

```
<title>Google</title>
```

In [194…

```
soupObj.find('p') # soupObj.p
```

Out[194…

```
<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/priv
acy/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>
```

In [ ]:

```
# soupObj.find('p') # like a dict   <Tag>=<Value>
#                                      |->dictKey = Value
```

In [197…

```
d={'K1':'V1'}
d['K1']
# d['Kx']  -> KeyError: 'Kx'
```

Out[197…

```
'V1'
```

In [199…

```
# dictName.get('inputKey') ->Value / None  Vs  dictName['InputKey'] ->Value/KeyE
print(d.get('K1'))
print(d.get('Kx'))
```

```
V1
None
```

In [200…

```
soupObj.find('p')['style']
```

Out[200…

```
'font-size:8pt;color:#636363'
```

In [201…

```
soupObj.find('p').get('style')
```

Out[201…

```
'font-size:8pt;color:#636363'
```

In [203…

```
soupObj.find('p').text
```

Out[203…

```
'© 2025 - Privacy - Terms'
```

In [205…

```
soupObj.title.text
```

Out[205…

```
'Google'
```

In [206…

```
soupObj.body.text
```

Out[206…

```
'Search Images Maps Play YouTube News Gmail Drive More »Web History | Settings
| Sign in\xa0Advanced searchGoogle offered in:  हिन्दी বাংলা తెలుగు मराठी தமிழ் ગુજ
રાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ AdvertisingBusiness SolutionsAbout GoogleGoogle.co.in
© 2025 - Privacy - Terms'
```

In [207…

```
soupObj.find('p')
```

Out[207…

```
<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/priv
acy/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>
```

In [208…

```
soupObj.find_all('p')
```

Out[208…

```
[<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/pri
vacy/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>]
```

In [209…

```
soupObj.get_text() # soupObj.body.text
```

Out[209…

```
'GoogleSearch Images Maps Play YouTube News Gmail Drive More »Web History | Set
tings | Sign in\xa0Advanced searchGoogle offered in:  हिन्दी বাংলা తెలుగు मराठी த
மிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ AdvertisingBusiness SolutionsAbout GoogleGoogl
e.co.in© 2025 - Privacy - Terms'
```

In [210…

```
soupObj.find_all('p')
```

Out[210…

```
[<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/pri
vacy/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>]
```

In [211…

```
soupObj.find('a')
```

Out[211…

```
<a class="gb1" href="https://www.google.com/imghp?hl=en&amp;tab=wi">Images</a>
```

In [212…

```
soupObj.find_all('a')
```

Out[212…

```
[<a class="gb1" href="https://www.google.com/imghp?hl=en&amp;tab=wi">Images</a
>,
 <a class="gb1" href="https://maps.google.co.in/maps?hl=en&amp;tab=wl">Maps</a
>,
 <a class="gb1" href="https://play.google.com/?hl=en&amp;tab=w8">Play</a>,
 <a class="gb1" href="https://www.youtube.com/?tab=w1">YouTube</a>,
 <a class="gb1" href="https://news.google.com/?tab=wn">News</a>,
 <a class="gb1" href="https://mail.google.com/mail/?tab=wm">Gmail</a>,
 <a class="gb1" href="https://drive.google.com/?tab=wo">Drive</a>,
 <a class="gb1" href="https://www.google.co.in/intl/en/about/products?tab=wh" s
tyle="text-decoration:none"><u>More</u> »</a>,
 <a class="gb4" href="http://www.google.co.in/history/optout?hl=en">Web History
</a>,
 <a class="gb4" href="/preferences?hl=en">Settings</a>,
 <a class="gb4" href="https://accounts.google.com/ServiceLogin?hl=en&amp;passiv
e=true&amp;continue=https://www.google.com/&amp;ec=GAZAAQ" id="gb_70" target="_
top">Sign in</a>,
 <a href="/advanced_search?hl=en-IN&amp;authuser=0">Advanced search</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=hi&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAY">हिन्दी</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=bn&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAc">বাংলা</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=te&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAg">తెలుగు</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=mr&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAk">मराठी</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=ta&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAo">தமிழ்</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=gu&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAs">ગુજરાતી</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=kn&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCAw">ಕನ್ನಡ</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=ml&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCA0">മലയാളം</a>,
 <a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&
amp;hl=pa&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8
Q2ZgBCA4">ਪੰਜਾਬੀ</a>,
 <a href="/intl/en/ads/">Advertising</a>,
 <a href="http://www.google.co.in/services/">Business Solutions</a>,
 <a href="/intl/en/about.html">About Google</a>,
 <a href="https://www.google.com/setprefdomain?prefdom=IN&amp;prev=https://www.
google.co.in/&amp;sig=K_1ibo2GagxxYN4HnbCWS0x7bAki4%3D">Google.co.in</a>,
 <a href="/intl/en/policies/privacy/">Privacy</a>,
 <a href="/intl/en/policies/terms/">Terms</a>]
```

In [213…

```
soupObj.find_all('a')[0]
```

Out[213…

```
<a class="gb1" href="https://www.google.com/imghp?hl=en&amp;tab=wi">Images</a>
```

In [214…

```
soupObj.find_all('a')[0]['href']
```

Out[214…

```
'https://www.google.com/imghp?hl=en&tab=wi'
```

In [217…

```python
soupObj.find_all('a')[0].text
```

Out[217…

```
'Images'
```

In [218…

```python
for var in soupObj.find_all('a'):
    print(var.get('href'))
```

```
https://www.google.com/imghp?hl=en&tab=wi
https://maps.google.co.in/maps?hl=en&tab=wl
https://play.google.com/?hl=en&tab=w8
https://www.youtube.com/?tab=w1
https://news.google.com/?tab=wn
https://mail.google.com/mail/?tab=wm
https://drive.google.com/?tab=wo
https://www.google.co.in/intl/en/about/products?tab=wh
http://www.google.co.in/history/optout?hl=en
/preferences?hl=en
https://accounts.google.com/ServiceLogin?hl=en&passive=true&continue=https://www.
google.com/&ec=GAZAAQ
/advanced_search?hl=en-IN&authuser=0
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=hi&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAY
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=bn&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAc
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=te&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAg
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=mr&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAk
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=ta&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAo
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=gu&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAs
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=kn&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAw
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=ml&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCA0
https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=pa&source
=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCA4
/intl/en/ads/
http://www.google.co.in/services/
/intl/en/about.html
https://www.google.com/setprefdomain?prefdom=IN&prev=https://www.google.co.in/&si
g=K_1ibo2GagxxYN4HnbCWS0x7bAki4%3D
/intl/en/policies/privacy/
/intl/en/policies/terms/
```

In [219…

```python
gURLs=[]
for var in soupObj.find_all('a'):
    gURLs.append(var.get('href'))
```

In [220…

```python
len(gURLs)
```

Out[220…

```
27
```

In [221…

```python
d={}
d['Google'] = gURLs
# d['Python'] = pURLs
# d['aws'] = aURLs
d
```

Out[221…

```
{'Google': ['https://www.google.com/imghp?hl=en&tab=wi',
  'https://maps.google.co.in/maps?hl=en&tab=wl',
  'https://play.google.com/?hl=en&tab=w8',
  'https://www.youtube.com/?tab=w1',
  'https://news.google.com/?tab=wn',
  'https://mail.google.com/mail/?tab=wm',
  'https://drive.google.com/?tab=wo',
  'https://www.google.co.in/intl/en/about/products?tab=wh',
  'http://www.google.co.in/history/optout?hl=en',
  '/preferences?hl=en',
  'https://accounts.google.com/ServiceLogin?hl=en&passive=true&continue=http
s://www.google.com/&ec=GAZAAQ',
  '/advanced_search?hl=en-IN&authuser=0',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=hi&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAY',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=bn&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAc',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=te&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAg',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=mr&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAk',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=ta&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAo',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=gu&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAs',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=kn&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCAw',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=ml&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCA0',
  'https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&hl=pa&s
ource=homepage&sa=X&ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2ZgBCA4',
  '/intl/en/ads/',
  'http://www.google.co.in/services/',
  '/intl/en/about.html',
  'https://www.google.com/setprefdomain?prefdom=IN&prev=https://www.google.co.i
n/&sig=K_1ibo2GagxxYN4HnbCWS0x7bAki4%3D',
  '/intl/en/policies/privacy/',
  '/intl/en/policies/terms/']}
```

In [223…

```
for var in soupObj.find_all('a'):
    print(var,var.text)
```

```
<a class="gb1" href="https://www.google.com/imghp?hl=en&amp;tab=wi">Images</a> Im
ages
<a class="gb1" href="https://maps.google.co.in/maps?hl=en&amp;tab=wl">Maps</a> Ma
ps
<a class="gb1" href="https://play.google.com/?hl=en&amp;tab=w8">Play</a> Play
<a class="gb1" href="https://www.youtube.com/?tab=w1">YouTube</a> YouTube
<a class="gb1" href="https://news.google.com/?tab=wn">News</a> News
<a class="gb1" href="https://mail.google.com/mail/?tab=wm">Gmail</a> Gmail
<a class="gb1" href="https://drive.google.com/?tab=wo">Drive</a> Drive
<a class="gb1" href="https://www.google.co.in/intl/en/about/products?tab=wh" styl
e="text-decoration:none"><u>More</u> »</a> More »
<a class="gb4" href="http://www.google.co.in/history/optout?hl=en">Web History</a
> Web History
<a class="gb4" href="/preferences?hl=en">Settings</a> Settings
<a class="gb4" href="https://accounts.google.com/ServiceLogin?hl=en&amp;passive=t
rue&amp;continue=https://www.google.com/&amp;ec=GAZAAQ" id="gb_70" target="_top">
Sign in</a> Sign in
<a href="/advanced_search?hl=en-IN&amp;authuser=0">Advanced search</a> Advanced s
earch
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=hi&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAY">हिन्दी</a> हिन्दी
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=bn&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAc">বাংলা</a> বাংলা
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=te&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAg">తెలుగు</a> తెలుగు
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=mr&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAk">मराठी</a> मराठी
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=ta&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAo">தமிழ்</a> தமிழ்
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=gu&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAs">ગુજરાતી</a> ગુજરાતી
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=kn&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCAw">ಕನ್ನಡ</a> ಕನ್ನಡ
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=ml&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCA0">മലയാളം</a> മലയാളം
<a href="https://www.google.com/setprefs?sig=0_gdycJoLm1PbO_kfzhlXgs1ne-VM%3D&am
p;hl=pa&amp;source=homepage&amp;sa=X&amp;ved=0ahUKEwiKofKjudCPAxUmP7kGHTBWHW8Q2Zg
BCA4">ਪੰਜਾਬੀ</a> ਪੰਜਾਬੀ
<a href="/intl/en/ads/">Advertising</a> Advertising
<a href="http://www.google.co.in/services/">Business Solutions</a> Business Solut
ions
<a href="/intl/en/about.html">About Google</a> About Google
<a href="https://www.google.com/setprefdomain?prefdom=IN&amp;prev=https://www.goo
gle.co.in/&amp;sig=K_1ibo2GagxxYN4HnbCWS0x7bAki4%3D">Google.co.in</a> Google.co.i
n
<a href="/intl/en/policies/privacy/">Privacy</a> Privacy
<a href="/intl/en/policies/terms/">Terms</a> Terms
```

In [227…

```
for var in soupObj.find_all(['p','tr','td']):
    print(f'{var}\n{var.text}\n--\n')
```

```
<tr valign="top"><td width="25%"> </td><td align="center" nowrap=""><input name
="ie" type="hidden" value="ISO-8859-1"/><input name="hl" type="hidden" value="en-
IN"/><input name="source" type="hidden" value="hp"/><input name="biw" type="hidde
n"/><input name="bih" type="hidden"/><div class="ds" style="height:32px;margin:4p
x 0"><input autocomplete="off" class="lst" maxlength="2048" name="q" size="57" st
yle="margin:0;padding:5px 8px 0 6px;vertical-align:top;color:#1f1f1f" title="Goog
le Search" value=""/></div><br style="line-height:0"/><span class="ds"><span clas
s="lsbb"><input class="lsb" name="btnG" type="submit" value="Google Search"/></sp
an></span><span class="ds"><span class="lsbb"><input class="lsb" id="tsuid_SZ3CaM
rYBab-5OUPsKz1-AY_1" name="btnI" type="submit" value="I'm Feeling Lucky"/><script
nonce="XPfiKqHYsOQ2Ai9FvoEd_w">(function(){var id='tsuid_SZ3CaMrYBab-5OUPsKz1-AY_
1';document.getElementById(id).onclick = function(){if (this.form.q.value){this.c
hecked = 1;if (this.form.iflsig)this.form.iflsig.disabled = false;}
else top.location='/doodles/';};})();</script><input name="iflsig" type="hidden"
value="AOw8s4IAAAAAaMKrWYqt1vOPqQvcA5TadWVojeWtp7vW"/></span></span></td><td alig
n="left" class="fl sblc" nowrap="" width="25%"><a href="/advanced_search?hl=en-IN
&amp;authuser=0">Advanced search</a></td></tr>
 Advanced search
--

<td width="25%"> </td>

--

<td align="center" nowrap=""><input name="ie" type="hidden" value="ISO-8859-1"/><
input name="hl" type="hidden" value="en-IN"/><input name="source" type="hidden" v
alue="hp"/><input name="biw" type="hidden"/><input name="bih" type="hidden"/><div
class="ds" style="height:32px;margin:4px 0"><input autocomplete="off" class="lst"
maxlength="2048" name="q" size="57" style="margin:0;padding:5px 8px 0 6px;vertica
l-align:top;color:#1f1f1f" title="Google Search" value=""/></div><br style="line-
height:0"/><span class="ds"><span class="lsbb"><input class="lsb" name="btnG" typ
e="submit" value="Google Search"/></span></span><span class="ds"><span class="lsb
b"><input class="lsb" id="tsuid_SZ3CaMrYBab-5OUPsKz1-AY_1" name="btnI" type="subm
it" value="I'm Feeling Lucky"/><script nonce="XPfiKqHYsOQ2Ai9FvoEd_w">(function()
{var id='tsuid_SZ3CaMrYBab-5OUPsKz1-AY_1';document.getElementById(id).onclick = f
unction(){if (this.form.q.value){this.checked = 1;if (this.form.iflsig)this.form.
iflsig.disabled = false;}
else top.location='/doodles/';};})();</script><input name="iflsig" type="hidden"
value="AOw8s4IAAAAAaMKrWYqt1vOPqQvcA5TadWVojeWtp7vW"/></span></span></td>

--

<td align="left" class="fl sblc" nowrap="" width="25%"><a href="/advanced_search?
hl=en-IN&amp;authuser=0">Advanced search</a></td>
Advanced search
--

<p style="font-size:8pt;color:#636363">© 2025 - <a href="/intl/en/policies/privac
y/">Privacy</a> - <a href="/intl/en/policies/terms/">Terms</a></p>
© 2025 - Privacy - Terms
--
```

In [229…

```
requests.get('https://www.python.org').headers['Content-Type']
```

Out[229…

```
'text/html; charset=utf-8'
```

In [230…

```
pSoup = bs4.BeautifulSoup(requests.get('https://www.python.org').text)
pSoup.title
```

Out[230…

```
<title>Welcome to Python.org</title>
```

In [234…

```
for var in pSoup.find_all(['h1','h2','h3']):
    print(var.text)
```

```


Functions Defined
Compound Data Types
Intuitive Interpretation
All the Flow You’d Expect
Quick & Easy to Learn
Get Started
Download
Docs
Jobs
Latest News
Upcoming Events
Success Stories
Use Python for…

>>> Python Software Foundation
```

In [236…

```
headers = []
for var in pSoup.find_all(['h1','h2','h3']):
    headers.append(var.text.strip())

headers
```

Out[236…

```
['',
 'Functions Defined',
 'Compound Data Types',
 'Intuitive Interpretation',
 'All the Flow You’d Expect',
 'Quick & Easy to Learn',
 'Get Started',
 'Download',
 'Docs',
 'Jobs',
 'Latest News',
 'Upcoming Events',
 'Success Stories',
 'Use Python for…',
 '>>> Python Software Foundation']
```

In [248…

```
import sqlite3
conn = sqlite3.connect('python_tags.db')
sth = conn.cursor()
sth.execute("create table if not exists pythonorg(id INT PRIMARY KEY AUTOINCREME
```

Out[248…

```
<sqlite3.Cursor at 0x29bb1c2b6c0>
```

In [249…

```
for var in headers:
    sth.execute("insert into pythonorg(heading) values(?)",(var,))
```

In [250…

```
conn.commit()
```

In [251…

```
sth.execute('select *from pythonorg')
list(sth)
```

Out[251…

```
[(None, ''),
 (None, 'Functions Defined'),
 (None, 'Compound Data Types'),
 (None, 'Intuitive Interpretation'),
 (None, 'All the Flow You’d Expect'),
 (None, 'Quick & Easy to Learn'),
 (None, 'Get Started'),
 (None, 'Download'),
 (None, 'Docs'),
 (None, 'Jobs'),
 (None, 'Latest News'),
 (None, 'Upcoming Events'),
 (None, 'Success Stories'),
 (None, 'Use Python for…'),
 (None, '>>> Python Software Foundation')]
```

In [ ]:

```
##################### end of day3 #######################
```