In [1]:
```
d={}

d['K1']=[]
d['K1'].append({'url':['google.com','python.org','abc.com']})
print(d)
```

```
{'K1': [{'url': ['google.com', 'python.org', 'abc.com']}]}
```

In [2]:
```
d['K1'].append({'url':['oracle.com','ab.org','xyz.com']})
```

In [4]:
```
d['K2']={'Kx':[{'K1':[1,2,3,4]}]}
print(d)
```

```
{'K1': [{'url': ['google.com', 'python.org', 'abc.com']}, {'url': ['oracle.co
m', 'ab.org', 'xyz.com']}], 'K2': {'Kx': [{'K1': [1, 2, 3, 4]}]}}
```

In [5]:
```
import pprint
pprint.pprint(d)
```

```
{'K1': [{'url': ['google.com', 'python.org', 'abc.com']},
        {'url': ['oracle.com', 'ab.org', 'xyz.com']}],
 'K2': {'Kx': [{'K1': [1, 2, 3, 4]}]}}
```

In [6]:
```
pprint.pprint(d['K1'])
```

```
[{'url': ['google.com', 'python.org', 'abc.com']},
 {'url': ['oracle.com', 'ab.org', 'xyz.com']}]
```

In [7]:
```
L=['D1','D2','D3','D4','D5']
d={'K1':'V1','K2':'V2','K3':'V3'}
pprint.pprint(L)
pprint.pprint(d)
```

```
['D1', 'D2', 'D3', 'D4', 'D5']
{'K1': 'V1', 'K2': 'V2', 'K3': 'V3'}
```

In [ ]:
```
# DataBase
#------------
1. import <DB_Module_NAME>
2. <DB_Module_NAME>.connect() ->dbconnectionObject
3. dbconnectionObject.cursor() ->cursor_object
4. cursor_object.execute('Query')
   ..
   ..
   dbconnectionObject.commit()
5. cursor_object.close()
6. dbconnectionObject.close()
```

In [ ]:

```
C:\Users\Raja>mkdir mydb

C:\Users\Raja>cd mydb

C:\Users\Raja\mydb>dir
 Volume in drive C has no label.
 Volume Serial Number is C08D-3DB3

 Directory of C:\Users\Raja\mydb

27-11-2024  10:10    <DIR>          .
27-11-2024  10:10    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  80,677,830,656 bytes free

C:\Users\Raja\mydb>python
Python 3.13.0 (tags/v3.13.0:60403a5, Oct  7 2024, 09:38:07) [MSC v.1941 64 bit
Type "help", "copyright", "credits" or "license" for more information.
>>> import sqlite3
>>> sqlite3.connect
<built-in function connect>
>>>
>>> sqlite3.connect("products.db")
<sqlite3.Connection object at 0x0000017C9896B010>
>>> dbh = sqlite3.connect("products.db")
>>> sth = dbh.cursor()
>>> sth.execute("create table prod(id int,pname Text,pcost int)")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>> sth.execute("insert into prod values(101,'prodA',1000)")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>> sth.execute("insert into prod values(102,'prodB',2000)")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>> sth.execute("insert into prod values(103,'prodC',3000)")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>> sth.execute("insert into prod values(104,'prodD',4000)")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>> va=105
>>> vb='prodE'
>>> vc=5000
>>>
>>> sth.execute("insert into prod values(?,?,?)",(va,vb,vc))
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> sth.execute("select *from prod")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> sth.fetchone()
(101, 'prodA', 1000)
>>> sth.fetchone()
(102, 'prodB', 2000)
>>> sth.fetchone()
(103, 'prodC', 3000)
>>> sth.fetchone()
(104, 'prodD', 4000)
>>> sth.fetchone()
(105, 'prodE', 5000)
>>> sth.fetchone()
```

```
>>> sth.fetchone()
>>>
>>> sth.execute("select *from prod")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> sth.fetchall()
[(101, 'prodA', 1000), (102, 'prodB', 2000), (103, 'prodC', 3000), (104, 'prodD
>>>
>>> sth.execute("select *from prod")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> for var in sth:
...     print(var)
...
(101, 'prodA', 1000)
(102, 'prodB', 2000)
(103, 'prodC', 3000)
(104, 'prodD', 4000)
(105, 'prodE', 5000)
>>>
>>> sth.execute("select *from prod")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> list(sth) # generator
[(101, 'prodA', 1000), (102, 'prodB', 2000), (103, 'prodC', 3000), (104, 'prodD
>>>
>>> sth.execute("select *from prod")
<sqlite3.Cursor object at 0x0000017C9843B240>
>>>
>>> records = list(sth) # generator
>>>
>>> import json
>>> jd = json.dumps(records) # from python to json
>>> jd
'[[101, "prodA", 1000], [102, "prodB", 2000], [103, "prodC", 3000], [104, "proc
>>>
>>> pd = json.loads(jd) # from json to python
>>> pd
[[101, 'prodA', 1000], [102, 'prodB', 2000], [103, 'prodC', 3000], [104, 'prodD
>>>
>>> dbh.commit()
>>> dbh.close()
>>>
```

In [8]:
```
import sqlite3
sqlite3.connect('mydb\products.db')
```

Out[8]:
```
<sqlite3.Connection at 0x12ecb6232e0>
```

In [9]:
```
conn = sqlite3.connect('mydb\products.db')
sth = conn.cursor()
sth.execute("select *from prod")
list(sth) # generator
```

Out[9]:
```
[(101, 'prodA', 1000),
 (102, 'prodB', 2000),
 (103, 'prodC', 3000),
 (104, 'prodD', 4000),
 (105, 'prodE', 5000)]
```

In [21]:
```
conn = sqlite3.connect('mydb\emp.db')
sth = conn.cursor()
sth.execute('create table emp_table(eid INT,ename TEXT,edept TEXT,eplace TEXT,e
```

Out[21]:
```
<sqlite3.Cursor at 0x12ecb68c0c0>
```

In [22]:
```
for var in open('E:\\emp.csv'): # open a inputFile and iterate it
    var=var.strip() # remove \n chars
    va,vb,vc,vd,ve = var.split(",") # split single line into multiple values
    sth.execute('insert into emp_table values(?,?,?,?,?)',(va,vb,vc,vd,ve))
```

In [23]:
```
sth.execute('select *from emp_table')
list(sth) # generator
```

Out[23]:
```
[('eid', 'ename', 'edept', 'eplace', 'ecost'),
 (101, 'raj', 'sales', 'pune', 1000),
 (102, 'leo', 'prod', 'bglore', 2000),
 (103, 'paul', 'HR', 'chennai', 3000),
 (104, 'anu', 'hr', 'hyderabad', 4000),
 (456, 'kumar', 'sales', 'bglore', 3000),
 (105, 'zion', 'Hr', 'mumbai', 5000),
 (106, 'bibu', 'sales', 'bglore', 1450),
 (107, 'theeb', 'sales', 'noida', 4590),
 (108, 'bibu', 'sales', 'bglore', 5000),
 (113, 'kumar', 'prod', 'hyderabad', 5423)]
```

In [24]:
```
sth.execute("select *from emp_table where edept = 'sales'")
list(sth) # generator
```

Out[24]:
```
[(101, 'raj', 'sales', 'pune', 1000),
 (456, 'kumar', 'sales', 'bglore', 3000),
 (106, 'bibu', 'sales', 'bglore', 1450),
 (107, 'theeb', 'sales', 'noida', 4590),
 (108, 'bibu', 'sales', 'bglore', 5000)]
```

In [25]:
```
conn.commit()
```

In [26]:
```
conn.close()
```

In [27]:
```
conn = sqlite3.connect('mydb\emp.db')
cur = conn.cursor()
cur.execute('select *from emp_table')
records = cur.fetchall()
conn.close()
```

In [28]:
```
print(records)
```

```
[('eid', 'ename', 'edept', 'eplace', 'ecost'), (101, 'raj', 'sales', 'pune',
1000), (102, 'leo', 'prod', 'bglore', 2000), (103, 'paul', 'HR', 'chennai', 3
000), (104, 'anu', 'hr', 'hyderabad', 4000), (456, 'kumar', 'sales', 'bglor
e', 3000), (105, 'zion', 'Hr', 'mumbai', 5000), (106, 'bibu', 'sales', 'bglor
e', 1450), (107, 'theeb', 'sales', 'noida', 4590), (108, 'bibu', 'sales', 'bg
lore', 5000), (113, 'kumar', 'prod', 'hyderabad', 5423)]
```

In [30]:
```
wobj = open('mydb\emp.log','w')
for var in records:
    wobj.write(f'{var[0]}\t{var[1]}\t{var[-1]}\n')

wobj.close()
```

In [31]:
```
print(open('mydb\emp.log').read())
```

```
eid     ename   ecost
101     raj     1000
102     leo     2000
103     paul    3000
104     anu     4000
456     kumar   3000
105     zion    5000
106     bibu    1450
107     theeb   4590
108     bibu    5000
113     kumar   5423
```

In [32]:
```
import re
re.search
```

Out[32]:
```
<function re.search(pattern, string, flags=0)>
```

In [33]:
```
re.search('sales','sales details')
```

Out[33]:
```
<re.Match object; span=(0, 5), match='sales'>
```

In [34]:
```
sqlite3.connect
```

Out[34]:
```
<function _sqlite3.connect>
```

In [35]:
```
sqlite3.connect('test.db')
```

Out[35]:
```
<sqlite3.Connection at 0x12ecc3d3100>
```

In [36]:
```
def f1(a):
    class box:
        def __init__(self,a=0):
            self.a=a
        def method1(self):
            return self.a+100
    obj = box(a)
    return obj
```

In [37]:
```
f1(10)
```

Out[37]:
```
<__main__.f1.<locals>.box at 0x12ecc400290>
```

In [38]:
```
myobj = f1(10)
myobj.method1()
```

Out[38]:
```
110
```

In [ ]:
```
numpy    - computation
pandas   - dataframe - analysis
matplotlib - visualization
```

In [39]:
```
L=[10,20,30,40,50]
L+100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[39], line 2
      1 L=[10,20,30,40,50]
----> 2 L+100

TypeError: can only concatenate list (not "int") to list
```

In [40]:
```
L.append('data')
L
```

Out[40]:
```
[10, 20, 30, 40, 50, 'data']
```

In [42]:
```
import array
#help(array)

arr = array.array('i',)
arr.append(10)
arr.append(20)
arr
```

Out[42]:
```
array('i', [10, 20])
```

In [43]:
```
arr = array.array('i',[10,20,30,40,50])
arr[0]
```

Out[43]:
```
10
```

In [44]:
```
arr[1]
```

Out[44]:
```
20
```

In [45]:
```
arr.append(56)
arr
```

Out[45]:
```
array('i', [10, 20, 30, 40, 50, 56])
```

In [47]:
```
arr.append(56.0) # Vs L.append('data')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[47], line 1
----> 1 arr.append(56.0)

TypeError: 'float' object cannot be interpreted as an integer
```

In [46]:
```
arr+100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[46], line 1
----> 1 arr+100

TypeError: can only append array (not "int") to array
```

In [ ]:
```
numpy
 - numerical computation
 - universal mathematical functions
 |
 - like array - index based
 - supports - slicing
 - we can perform multidimensional array
                     |->reshape
```

In [ ]:
```
file:ab.py
     -----   =======> import ab
                      ab.<variable>
project/                                  -> import m1
        |__ m1.py m2.py ... m100.py            ...
        |__ Sub1/ma.py mb.py ...mz.py
    |_____________________________________   import project
          |                                    project.<member>
          |
           __init__.py <==
           from project.m1 import *
           from project.Sub1.mb import *
```

In [48]:
```
import numpy
numpy.__file__
```

Out[48]:
```
'C:\\ProgramData\\anaconda3\\Lib\\site-packages\\numpy\\__init__.py'
```

In [49]:
```
import re
re.__file__
```

Out[49]:
```
'C:\\ProgramData\\anaconda3\\Lib\\re\\__init__.py'
```

In [50]:
```
import os
os.__file__
```

Out[50]:
```
'C:\\ProgramData\\anaconda3\\Lib\\os.py'
```

In [51]:
```
import numpy
print(numpy.__file__)
numpy.__version__
```

```
C:\ProgramData\anaconda3\Lib\site-packages\numpy\__init__.py
```

Out[51]:
```
'1.24.3'
```

In [52]:
```
import numpy
numpy.array([10,20,30,40,50])
```

Out[52]:
```
array([10, 20, 30, 40, 50])
```

In [ ]:
```
L=[10,20,30,40,50]
L+100 =>Error

arr = array.array('i',[10,20,30,40,50])
arr+100 =>Error
```

In [53]:
```
narr = numpy.array([10,20,30,40,50])
narr + 100
```

Out[53]:
```
array([110, 120, 130, 140, 150])
```

In [54]:
```
narr[0]
```

Out[54]:
```
10
```

In [55]:
```
narr[-1]
```

Out[55]:
```
50
```

In [56]:
```
narr[1:]
```

Out[56]:
```
array([20, 30, 40, 50])
```

In [57]:
```
narr[-2:]
```

Out[57]:
```
array([40, 50])
```

In [58]:
```
narr + 0.34
```

Out[58]:
```
array([10.34, 20.34, 30.34, 40.34, 50.34])
```

In [59]:
```
narr * 2.42
```

Out[59]:
```
array([ 24.2,  48.4,  72.6,  96.8, 121. ])
```

In [60]:
```
narr / 5
```

Out[60]:
```
array([ 2.,  4.,  6.,  8., 10.])
```

In [61]:
```
narr
```

Out[61]:
```
array([10, 20, 30, 40, 50])
```

In [62]:
```
narr >30
```

Out[62]:
```
array([False, False, False,  True,  True])
```

In [63]:
```
narr > 24.32
```

Out[63]:
```
array([False, False,  True,  True,  True])
```

In [64]:
```
numpy.arange(5) # like range(5)  <or> list(range(5)) ->[0,1,2,3,4]
```

Out[64]:
```
array([0, 1, 2, 3, 4])
```

In [65]:
```
numpy.random.rand(10)
```

Out[65]:
```
array([0.05281847, 0.43360102, 0.22973833, 0.69683384, 0.20139467,
       0.74182012, 0.43539863, 0.12976905, 0.98761256, 0.33877046])
```

In [66]:
```
import numpy as np
narr = np.array([10,20,30,40,50])
```

In [68]:
```
narr.ndim # ->1 <== one - index
```

Out[68]:
```
1
```

In [69]:
```
narr.shape # 1D => no.of items (5,) -> total no.of items are 5
```

Out[69]:
```
(5,)
```

In [71]:
```
narr = np.array([[10,20,30,40,50]])
narr.ndim
```

Out[71]:
```
2
```

In [72]:
```
narr.shape # 2D => row X columns  (1,5) ->1Row X 5Cols
#        0    1    2    3    4
#    ----------------------------
#  0  | 10 | 20 | 30 | 40 | 50 |
#    --------------------------
#
```

Out[72]:
```
(1, 5)
```

In [75]:
```
narr = np.array([[[10,20,30,40,50]]])
narr.ndim,narr.shape
```

Out[75]:
```
(3, (1, 1, 5))
```

In [76]:
```
narr = np.array([[10,20,30,40,50]])
narr[0]
```

Out[76]:
```
array([10, 20, 30, 40, 50])
```

In [77]:
```
narr[0][0]
```

Out[77]:
```
10
```

In [78]:
```
narr[0][1]
```

Out[78]:
```
20
```

In [79]:
```
narr[0,1] # same as narr[0][1]
```

Out[79]:
```
20
```

In [80]:
```
np.array([10,20,30]).dtype
```

Out[80]:
```
dtype('int32')
```

In [81]:
```
np.array([10,20,30.0]).dtype
```

Out[81]:
```
dtype('float64')
```

In [82]:
```
np.array([10,20,30.0,True]).dtype
```

Out[82]:
```
dtype('float64')
```

In [83]:
```
np.array([10,20,30.0,'data']).dtype
```

Out[83]:
```
dtype('<U32')
```

In [84]:
```
np.array([True,False]).dtype
```

Out[84]:
```
dtype('bool')
```

In [85]:
```
a1 = np.array([[1,2,3,4,5],[6,7,8,9,10]])
a1[1]
```

Out[85]:
```
array([ 6,  7,  8,  9, 10])
```

In [86]:
```
a1[1,1:3]
```

Out[86]:
```
array([7, 8])
```

In [87]:
```
a2 = np.array([10,20,30,40,50,60,70,80])
a2.ndim
```

Out[87]:
```
1
```

In [88]:
```
np.reshape(a2,(2,4))
```

Out[88]:
```
array([[10, 20, 30, 40],
       [50, 60, 70, 80]])
```

In [89]:
```
a2 = np.array([10,20,30,40,50,60,70,80,90])
a2.ndim
```

Out[89]:
```
1
```

In [91]:
```
#np.reshape(a2,(2,4))
#ValueError: cannot reshape array of size 9 into shape (2,4)
```

In [93]:
```
a2 = np.array([1,2,3,4,5,6,7,8])
a2.ndim
```

Out[93]:
```
1
```

In [95]:
```
np.reshape(a2,(2,2,2))
```

Out[95]:
```
array([[[1, 2],
        [3, 4]],

       [[5, 6],
        [7, 8]]])
```

In [96]:
```
a3 = np.reshape(a2,(2,2,2))
a3[0]
```

Out[96]:
```
array([[1, 2],
       [3, 4]])
```

In [97]:
```
a3[0][0]
```

Out[97]:
```
array([1, 2])
```

In [98]:
```
a3[0][0][0]
```

Out[98]:
```
1
```

In [99]:
```
a1 = np.array([1,2,3,4])
a2 = np.array([2,4,6,8])

a1+a2
```

Out[99]:
```
array([ 3,  6,  9, 12])
```

In [100]:
```
np.add(a1,a2)
```

Out[100]:
```
array([ 3,  6,  9, 12])
```

In [101]:
```
np.divide(a1,a2)
```

Out[101]:
```
array([0.5, 0.5, 0.5, 0.5])
```

In [102]:
```
a1**a2
```

Out[102]:
```
array([    1,    16,   729, 65536])
```

In [103]:
```
np.power(a1,a2)
```

Out[103]:
```
array([    1,    16,   729, 65536])
```

In [104]:
```
np.power(a1,2)
```

Out[104]:
```
array([ 1,  4,  9, 16], dtype=int32)
```

In [105]:
```
np.mod(a1,a2)
```

Out[105]:
```
array([1, 2, 3, 4])
```

In [ ]:
```
< less()
> greater()
<= less_equal()
>= greater_equal()
!= not_equal()
```

In [106]:
```
np.sin(50)
```

Out[106]:
```
-0.26237485370392877
```

In [108]:
```
np.zeros(5)
```

Out[108]:
```
array([0., 0., 0., 0., 0.])
```

In [109]:
```
np.ones(5)
```

Out[109]:
```
array([1., 1., 1., 1., 1.])
```

In [110]:
```
np.random.randn(5)
```

Out[110]:
```
array([ 0.81097122, -0.76450369, -1.20327516, -1.98411229,  0.69072728])
```

In [116]:
```
a4 = np.random.randn(5)
print(a4)
print("max=",np.max(a4))
print("min=",np.min(a4))
print("mean=",np.mean(a4))
```

```
[-0.13366042 -0.31866182 -0.74737308 -0.31478422 -0.6445233 ]
max= -0.133660423757907
min= -0.7473730774569033
mean= -0.431800567149233
```

In [118]:
```
# matrix
a5 = np.array([[1,2],[3,4]])
print(a5.ndim,a5.shape)
```

```
2 (2, 2)
```

In [122]:
```
a6 = np.matrix([[1,2],[3,4]]) # same as 2D array - np.array([[],[]])
print(a6.ndim,a6.shape)
a6
```

```
2 (2, 2)
```

Out[122]:
```
matrix([[1, 2],
        [3, 4]])
```

In [123]:
```
a5+a6 #martix addition
```

Out[123]:
```
matrix([[2, 4],
        [6, 8]])
```

In [125]:
```
a5
```

Out[125]:
```
array([[1, 2],
       [3, 4]])
```

In [124]:
```
np.transpose(a5)
```

Out[124]:
```
array([[1, 3],
       [2, 4]])
```

In [126]:
```
a5 * a6 # matrix multiplication
```

Out[126]:
```
matrix([[ 7, 10],
        [15, 22]])
```

In [127]:
```
a5 @ a6 # matrix multiplication
```

Out[127]:
```
matrix([[ 7, 10],
        [15, 22]])
```

In [128]:
```
a1=np.array([10,20,30])
a2=np.array([10,20,40])
np.union1d(a1,a2) # set(a1)|set(a2) <or> a1.union(a2)
```

Out[128]:
```
array([10, 20, 30, 40])
```

In [129]:
```
np.intersect1d(a1,a2)
```

Out[129]:
```
array([10, 20])
```

In [130]:
```
L=[10,20,30,40,50]

t=0
for var in L:
    t=t+var
print(t)
```

```
150
```

In [131]:
```
np.sum([10,20,30,40,50])
```

Out[131]:
```
150
```

In [132]:
```
np.sum(L)
```

Out[132]:
```
150
```

In [133]:
```
np.min(L)
```

Out[133]:
```
10
```

In [134]:
```
np.median(L)
```

Out[134]:
```
30.0
```

In [213]:
```
# numpy can hold different types of value
numpy.array([10,20.0,'data',True])
```

Out[213]:
```
array(['10', '20.0', 'data', 'True'], dtype='<U32')
```

In [215]:
```
a = numpy.array([10,20,30,'abc'])
print(a)
a+100
```

```
['10' '20' '30' 'abc']

---------------------------------------------------------------------------
UFuncTypeError                            Traceback (most recent call last)
Cell In[215], line 3
      1 a = numpy.array([10,20,30,'abc'])
      2 print(a)
----> 3 a+100

UFuncTypeError: ufunc 'add' did not contain a loop with signature matching ty
pes (dtype('<U11'), dtype('int32')) -> None
```

In [216]:
```
a1 = numpy.array([10,20,30])
a1[1]
```

Out[216]:
```
20
```

In [217]:
```
a1[1]=100 # we can modify
a1
```

Out[217]:
```
array([ 10, 100,  30])
```

In [218]:
```
a1[1]=200.0
```

In [219]:
```
a1
```

Out[219]:
```
array([ 10, 200,  30])
```

In [220]:
```
a1[1]='data'  # Vs List_name[1]='data'
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[220], line 1
----> 1 a1[1]='data'

ValueError: invalid literal for int() with base 10: 'data'
```

In [223]:
```
a2 = numpy.array([10,20,30,'data'],dtype='object')
a2[1]='data' # -------------------------------
a2
```

Out[223]:
```
array([10, 'data', 30, 'data'], dtype=object)
```

In [135]:
```
OL={}
OL['OL8']=[]
OL['OL9']=[]
OL
```

Out[135]:
```
{'OL8': [], 'OL9': []}
```

In [ ]:
```
Download list of rpm files from
https://yum.oracle.com/repo/OracleLinux/OL8/baseos/latest/x86_64/index.html
|
Append list of all the rpm files to list_
|
add rpm list to an existing dict
|
convert to json # OL.json
```

In [137]:
```
import requests
r=requests.get('https://yum.oracle.com/repo/OracleLinux/OL8/baseos/latest/x86_6
if(r.status_code != 200):
    print('url download is failed')
```

In [138]:
```
ol_web = r.text
```

In [139]:
```
import bs4
ol_obj = bs4.BeautifulSoup(ol_web)
```

In [141]:
```
ol_obj.find('title')
```

Out[141]:
```
<title>Oracle Linux 8 (x86_64)  BaseOS Latest | Oracle, Software. Hardware. C
omplete.</title>
```

In [145]:
```
import re
```

In [148]:
```
for var in ol_obj.find_all('a'):
    if(re.search('rpm$',var.get('href'))):
        print(var.string)
```

```
NetworkManager-adsl-1.32.10-4.0.1.el8.x86_64.rpm
NetworkManager-adsl-1.32.10-5.0.1.el8_5.x86_64.rpm
NetworkManager-adsl-1.36.0-4.0.1.el8.x86_64.rpm
NetworkManager-adsl-1.36.0-7.0.1.el8_6.x86_64.rpm
NetworkManager-adsl-1.36.0-9.0.1.el8_6.x86_64.rpm
NetworkManager-adsl-1.40.0-1.0.2.el8.x86_64.rpm
NetworkManager-adsl-1.40.0-2.0.1.el8_7.x86_64.rpm
NetworkManager-adsl-1.40.0-5.0.1.el8_7.x86_64.rpm
NetworkManager-adsl-1.40.0-5.0.2.el8_7.x86_64.rpm
NetworkManager-adsl-1.40.0-5.0.3.el8_7.x86_64.rpm
NetworkManager-adsl-1.40.0-6.0.1.el8_7.x86_64.rpm
NetworkManager-adsl-1.40.16-1.0.1.el8.x86_64.rpm
NetworkManager-adsl-1.40.16-13.0.1.el8_9.x86_64.rpm
NetworkManager-adsl-1.40.16-15.0.1.el8.x86_64.rpm
NetworkManager-adsl-1.40.16-15.0.1.el8_9.x86_64.rpm
NetworkManager-adsl-1.40.16-3.0.1.el8_8.x86_64.rpm
NetworkManager-adsl-1.40.16-4.0.1.el8_8.x86_64.rpm
NetworkManager-adsl-1.40.16-9.0.1.el8.x86_64.rpm
NetworkManager-bluetooth-1.14.0-14.el8.x86_64.rpm
NetworkManager-bluetooth-1.20.0-3.0.1.el8.x86_64.rpm
```

In [149]:
```
OL8=[] # empt list
```

In [150]:
```
for var in ol_obj.find_all('a'):
    if(re.search('rpm$',var.get('href'))):
        OL8.append(var.string)
```

In [151]:
```
ol={} # empty dict
```

In [152]:
```
ol['OL8']=OL8 # adding new data to an existing dict
```

In [153]:
```
pprint.pprint(ol)
```

```
         'NetworkManager-adsl-1.40.16-4.0.1.el8_8.x86_64.rpm',
         'NetworkManager-adsl-1.40.16-9.0.1.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.14.0-14.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.20.0-3.0.1.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.20.0-5.0.1.el8_1.x86_64.rpm',
         'NetworkManager-bluetooth-1.22.8-4.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.22.8-5.el8_2.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-12.0.1.el8_3.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-13.0.1.el8_3.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-14.0.1.el8_3.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-8.0.1.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-9.0.1.el8_3.x86_64.rpm',
         'NetworkManager-bluetooth-1.26.0-9.0.2.el8_3.x86_64.rpm',
         'NetworkManager-bluetooth-1.30.0-10.0.1.el8_4.x86_64.rpm',
         'NetworkManager-bluetooth-1.30.0-13.0.1.el8_4.x86_64.rpm',
         'NetworkManager-bluetooth-1.30.0-7.0.1.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.30.0-9.0.1.el8_4.x86_64.rpm',
         'NetworkManager-bluetooth-1.32.10-4.0.1.el8.x86_64.rpm',
         'NetworkManager-bluetooth-1.32.10-5.0.1.el8_5.x86_64.rpm',
         'NetworkManager-bluetooth-1.36.0-4.0.1.el8.x86_64.rpm',
```

In [155]:
```
import json
```

In [156]:
```
wobj = open('E:\\ol.json','w')
json.dump(ol,wobj)
wobj.close()
```

In [180]:
```
url='https://yum.oracle.com/repo/OracleLinux/OL8/baseos/latest/x86_64/index.htm
```

In [181]:
```
def download_url(url):
    r=requests.get(url)
    if(r.status_code != 200):
        print('url download is failed')
        return False
    if('text/html' in r.headers['Content-Type']):
        ol_web = r.text
        webparsing(ol_web)
    else:
        return 'Given url is not a webpage content'
```

In [182]:
```
Linux_rpm=[]
def webparsing(webpage):
    print(webpage)
    ol_obj = bs4.BeautifulSoup(webpage)
    for var in ol_obj.find_all('a'):
        if(re.search('rpm$',var.get('href'))):
            Linux_rpm.append(var.string)
```

In [184]:
```
download_url(url)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

In [185]:
```
ol_rpm={}
```

In [186]:
```
ol_rpm['OL8']=Linux_rpm
```

In [188]:
```
#ol_rpm
```

In [189]:
```
wobj = open('E:\\ol.json','w')
json.dump(ol_rpm,wobj)
wobj.close()
```

In [ ]:
```

```

In [199]:
```
url='https://yum.oracle.com/repo/OracleLinux/OL9/baseos/latest/x86_64/index.htm
```

In [195]:
```
def download_url(url):
    r=requests.get(url)
    if(r.status_code != 200):
        print('url download is failed')
        return False
    if('text/html' in r.headers['Content-Type']):
        ol_web = r.text
        webparsing(ol_web)
    else:
        return 'Given url is not a webpage content'
```

In [200]:
```
Linux_rpm=[]
def webparsing(webpage):
    print(webpage)
    ol_obj = bs4.BeautifulSoup(webpage)
    for var in ol_obj.find_all('a'):
        if(re.search('rpm$',var.get('href'))):
            Linux_rpm.append(var.string)
```

In [202]:
```
download_url(url)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

In [203]:
```
ol_rpm['OL9']=Linux_rpm
```

In [204]:
```
ol_rpm['OL9']
```

```
 'expat-2.5.0-1.el9_3.1.x86_64.rpm',
 'expat-2.5.0-2.el9_4.1.i686.rpm',
 'expat-2.5.0-2.el9_4.1.x86_64.rpm',
 'expat-2.5.0-2.el9_4.i686.rpm',
 'expat-2.5.0-2.el9_4.x86_64.rpm',
 'expat-2.5.0-3.el9_5.1.i686.rpm',
 'expat-2.5.0-3.el9_5.1.x86_64.rpm',
 'fcoe-utils-1.0.34-0.git14ef0d2.el9.x86_64.rpm',
 'file-5.39-10.el9.x86_64.rpm',
 'file-5.39-12.1.el9_2.x86_64.rpm',
 'file-5.39-12.el9.x86_64.rpm',
 'file-5.39-14.el9.x86_64.rpm',
 'file-5.39-16.el9.x86_64.rpm',
 'file-5.39-8.el9.x86_64.rpm',
 'file-libs-5.39-10.el9.i686.rpm',
 'file-libs-5.39-10.el9.x86_64.rpm',
 'file-libs-5.39-12.1.el9_2.i686.rpm',
 'file-libs-5.39-12.1.el9_2.x86_64.rpm',
 'file-libs-5.39-12.el9.i686.rpm',
```

In [205]:
```
wobj = open('E:\\ol.json','w')
json.dump(ol_rpm,wobj)
wobj.close()
```

In [208]:
```
# reading json data ->python
# pprint.pprint(json.load(open('E:\\ol.json')))
```

In [210]:
```
ol_rpm.keys()
```

Out[210]:
```
dict_keys(['OL8', 'OL9'])
```

In [212]:
```
print(len(ol_rpm['OL8']),len(ol_rpm['OL9']))
```

```
20955 9249
```

In [224]:
```
## pandas

import pandas
pandas.__file__
```

Out[224]:
```
'C:\\ProgramData\\anaconda3\\Lib\\site-packages\\pandas\\__init__.py'
```

In [225]:
```
pandas.__version__
```

Out[225]:
```
'2.0.3'
```

In [226]:
```
print(pandas.Series)
<pandas>
  |_<core>
       |__series.py
             |__ class Series:
                    def __init__(self..):
```

```
<class 'pandas.core.series.Series'>
```

In [227]:
```
import pandas as pd
S1 = pd.Series(['D1','D2','D3'])
S2 = pd.Series({'K1':'V1','K2':'V2','K3':'V3'})
print(S1)
print('')
print(S2)
```

```
0    D1
1    D2
2    D3
dtype: object

K1    V1
K2    V2
K3    V3
dtype: object
```

In [228]:
```
print(pandas.DataFrame)
```

```
<class 'pandas.core.frame.DataFrame'>
```

In [230]:
```
D1 = pd.DataFrame(['D1','D2','D3'])
D2 = pd.DataFrame({'K1':['V1','V2','V3']})
print(D1)
print('')
print(D2)
```

```
    0
0  D1
1  D2
2  D3

   K1
0  V1
1  V2
2  V3
```

In [231]:
```
products={}

products['pid']=['p101','p102','p103']
products['pname']=['pA','pB','pC']
products['pcost']=[1000,2000,3000]
products
```

Out[231]:
```
{'pid': ['p101', 'p102', 'p103'],
 'pname': ['pA', 'pB', 'pC'],
 'pcost': [1000, 2000, 3000]}
```

In [232]:
```
pd.DataFrame(products)
```

Out[232]:

	pid	pname	pcost
0	p101	pA	1000
1	p102	pB	2000
2	p103	pC	3000

In [233]:
```
df1 = pd.DataFrame(products)
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   pid     3 non-null      object
 1   pname   3 non-null      object
 2   pcost   3 non-null      int64 
dtypes: int64(1), object(2)
memory usage: 204.0+ bytes
```

In [235]:
```
df1.index
```

Out[235]:
```
RangeIndex(start=0, stop=3, step=1)
```

In [236]:
```
df1.columns
```

Out[236]:
```
Index(['pid', 'pname', 'pcost'], dtype='object')
```

In [237]:
```
# To display particular columns from dataframe
#
# df['Column']

df1['pid']
```

Out[237]:
```
0    p101
1    p102
2    p103
Name: pid, dtype: object
```

In [238]:
```
df1['pname']
```

Out[238]:
```
0    pA
1    pB
2    pC
Name: pname, dtype: object
```

In [239]:
```
df1['pcost']
```

Out[239]:
```
0    1000
1    2000
2    3000
Name: pcost, dtype: int64
```

In [241]:
```
df1['pcost']>1500
```

Out[241]:
```
0    False
1     True
2     True
Name: pcost, dtype: bool
```

In [243]:
```
df1[df1['pcost']>1500]
```

Out[243]:

	pid	pname	pcost
1	p102	pB	2000
2	p103	pC	3000

In [244]:
```
df1['pname'] == 'pC'
```

Out[244]:
```
0    False
1    False
2     True
Name: pname, dtype: bool
```

In [245]:
```
df1[df1['pname'] == 'pC']
```

Out[245]:

	pid	pname	pcost
2	p103	pC	3000

In [246]:
```
df1
```

Out[246]:

	pid	pname	pcost
0	p101	pA	1000
1	p102	pB	2000
2	p103	pC	3000

In [248]:
```
df1.loc[0]
```

Out[248]:
```
pid      p101
pname      pA
pcost    1000
Name: 0, dtype: object
```

In [249]:
```
df1.iloc[0]
```

Out[249]:
```
pid      p101
pname      pA
pcost    1000
Name: 0, dtype: object
```

In [250]:
```
df1.iloc[1]
```

Out[250]:
```
pid      p102
pname      pB
pcost    2000
Name: 1, dtype: object
```

In [251]:
```
df1.loc[1:3] # row index slicing
```

Out[251]:

	pid	pname	pcost
1	p102	pB	2000
2	p103	pC	3000

In [258]:
```
df1.loc[[0,2]]
```

Out[258]:

	pid	pname	pcost
0	p101	pA	1000
2	p103	pC	3000

In [259]:
```
df1.loc[1:3] ['pid']
```

Out[259]:
```
1    p102
2    p103
Name: pid, dtype: object
```

In [261]:
```
df1.loc[1:3] [['pid','pcost']] # Row X Col
```

Out[261]:

	pid	pcost
1	p102	2000
2	p103	3000

In [263]:
```
pd.DataFrame(products)
```

Out[263]:

	pid	pname	pcost
0	p101	pA	1000
1	p102	pB	2000
2	p103	pC	3000

In [264]:
```
pd.DataFrame(products,index=['p1','p2','p3'])
```

Out[264]:

	pid	pname	pcost
p1	p101	pA	1000
p2	p102	pB	2000
p3	p103	pC	3000

In [266]:
```
df2 = pd.DataFrame(products,index=['p1','p2','p3'])
df2.loc['p1']
```

Out[266]:
```
pid      p101
pname      pA
pcost    1000
Name: p1, dtype: object
```

In [268]:
```
# df2.iloc['p1']
# TypeError: Cannot index by location index with a non-integer key
```

In [269]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 3 entries, p1 to p3
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   pid     3 non-null      object
 1   pname   3 non-null      object
 2   pcost   3 non-null      int64 
dtypes: int64(1), object(2)
memory usage: 204.0+ bytes
```

In [270]:
```
df2.index
```

Out[270]:
```
Index(['p1', 'p2', 'p3'], dtype='object')
```

In [271]:
```
df2.loc['p2'][['pid','pname']]
```

Out[271]:
```
pid      p102
pname      pB
Name: p2, dtype: object
```

In [274]:
```
pd.DataFrame([10,20,30],index=['Item1','Item2','Item3'],columns=['K1'])
```

Out[274]:

	K1
Item1	10
Item2	20
Item3	30

In [275]:
```
pd.read_csv('E:\\emp.csv')
```

Out[275]:

	eid	ename	edept	eplace	ecost
0	101	raj	sales	pune	1000
1	102	leo	prod	bglore	2000
2	103	paul	HR	chennai	3000
3	104	anu	hr	hyderabad	4000
4	456	kumar	sales	bglore	3000
5	105	zion	Hr	mumbai	5000
6	106	bibu	sales	bglore	1450
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [276]:
```
df3 = pd.read_csv('E:\\emp.csv')
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   eid     10 non-null     int64 
 1   ename   10 non-null     object
 2   edept   10 non-null     object
 3   eplace  10 non-null     object
 4   ecost   10 non-null     int64 
dtypes: int64(2), object(3)
memory usage: 532.0+ bytes
```

In [277]:
```
df3.index
```

Out[277]:
```
RangeIndex(start=0, stop=10, step=1)
```

In [278]:
```
df3.columns
```

Out[278]:
```
Index(['eid', 'ename', 'edept', 'eplace', 'ecost'], dtype='object')
```

In [279]:
```
df3['eid']
```

Out[279]:
```
0    101
1    102
2    103
3    104
4    456
5    105
6    106
7    107
8    108
9    113
Name: eid, dtype: int64
```

In [280]:
```
df3['ename']
```

Out[280]:
```
0      raj
1      leo
2     paul
3      anu
4    kumar
5     zion
6     bibu
7    theeb
8     bibu
9    kumar
Name: ename, dtype: object
```

In [281]:
```
df3.loc[4:]
```

Out[281]:

	eid	ename	edept	eplace	ecost
4	456	kumar	sales	bglore	3000
5	105	zion	Hr	mumbai	5000
6	106	bibu	sales	bglore	1450
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [282]:
```
df3.loc[4:][['ename','eplace']]
```

Out[282]:

	ename	eplace
4	kumar	bglore
5	zion	mumbai
6	bibu	bglore
7	theeb	noida
8	bibu	bglore
9	kumar	hyderabad

In [283]:
```
df3.loc[4:8]
```

Out[283]:

	eid	ename	edept	eplace	ecost
4	456	kumar	sales	bglore	3000
5	105	zion	Hr	mumbai	5000
6	106	bibu	sales	bglore	1450
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000

In [284]:
```
df3.loc[4:8][['eid','ename','eplace']]
```

Out[284]:

	eid	ename	eplace
4	456	kumar	bglore
5	105	zion	mumbai
6	106	bibu	bglore
7	107	theeb	noida
8	108	bibu	bglore

In [287]:
```
df3[df3['ecost'] >4500]
```

Out[287]:

	eid	ename	edept	eplace	ecost
5	105	zion	Hr	mumbai	5000
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [289]:
```
# from 7th row to list of all - emp's ecost is above 4500
df3.loc[7:][df3['ecost']>4500]
```

```
C:\Users\Raja\AppData\Local\Temp\ipykernel_7772\3648241022.py:1: UserWarning:
Boolean Series key will be reindexed to match DataFrame index.
  df3.loc[7:][df3['ecost']>4500]
```

Out[289]:

	eid	ename	edept	eplace	ecost
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [290]:
```
df3['ecost']
```

Out[290]:
```
0    1000
1    2000
2    3000
3    4000
4    3000
5    5000
6    1450
7    4590
8    5000
9    5423
Name: ecost, dtype: int64
```

In [291]:
```
df3['ecost'] * 0.18
```

Out[291]:
```
0    180.00
1    360.00
2    540.00
3    720.00
4    540.00
5    900.00
6    261.00
7    826.20
8    900.00
9    976.14
Name: ecost, dtype: float64
```

In [294]:
```
df3.loc[7:][df3['ecost']>4500]
```

```
C:\Users\Raja\AppData\Local\Temp\ipykernel_7772\3648241022.py:1: UserWarning:
Boolean Series key will be reindexed to match DataFrame index.
  df3.loc[7:][df3['ecost']>4500]
```

Out[294]:

	eid	ename	edept	eplace	ecost
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [295]:
```
df3.loc[7:][df3['ecost']>4500]['ecost']
```

```
C:\Users\Raja\AppData\Local\Temp\ipykernel_7772\3662624420.py:1: UserWarning:
Boolean Series key will be reindexed to match DataFrame index.
  df3.loc[7:][df3['ecost']>4500]['ecost']
```

Out[295]:
```
7    4590
8    5000
9    5423
Name: ecost, dtype: int64
```

In [296]:
```
df3.loc[7:][df3['ecost']>4500]['ecost'] * 0.19
```

```
C:\Users\Raja\AppData\Local\Temp\ipykernel_7772\403829018.py:1: UserWarning:
Boolean Series key will be reindexed to match DataFrame index.
  df3.loc[7:][df3['ecost']>4500]['ecost'] * 0.19
```

Out[296]:
```
7     872.10
8     950.00
9    1030.37
Name: ecost, dtype: float64
```

In [301]:
```
pd.read_csv('netinfo.conf')
```

Out[301]:

	param = value
0	Type = ethernet
1	onboot = yes
2	proto = dhcp
3	ip = 10.20.30.40

In [302]:
```
pd.read_csv('netinfo.conf',sep="=")
```

Out[302]:

	param	value
0	Type	ethernet
1	onboot	yes
2	proto	dhcp
3	ip	10.20.30.40

In [303]:
```
net_df = pd.read_csv('netinfo.conf',sep="=")
net_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   param   4 non-null      object
 1    value  4 non-null      object
dtypes: object(2)
memory usage: 196.0+ bytes
```

In [305]:
```
net_df['param']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3
653, in Index.get_loc(self, key)
   3652 try:
-> 3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:147, i
n pandas._libs.index.IndexEngine.get_loc()

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:176, i
n pandas._libs.index.IndexEngine.get_loc()

File pandas\_libs\hashtable_class_helper.pxi:7080, in pandas._libs.hashtable.
PyObjectHashTable.get_item()

File pandas\_libs\hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.
PyObjectHashTable.get_item()

KeyError: 'param'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[305], line 1
----> 1 net_df['param']

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:3761, in
DataFrame.__getitem__(self, key)
   3759 if self.columns.nlevels > 1:
   3760     return self._getitem_multilevel(key)
-> 3761 indexer = self.columns.get_loc(key)
   3762 if is_integer(indexer):
   3763     indexer = [indexer]

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3
655, in Index.get_loc(self, key)
   3653     return self._engine.get_loc(casted_key)
   3654 except KeyError as err:
-> 3655     raise KeyError(key) from err
   3656 except TypeError:
   3657     # If we have a listlike key, _check_indexing_error will raise
   3658     #  InvalidIndexError. Otherwise we fall through and re-raise
   3659     #  the TypeError.
   3660     self._check_indexing_error(key)

KeyError: 'param'
```

In [308]:
```
net_df.columns
```

Out[308]:
```
Index(['param ', ' value'], dtype='object')
```

In [309]:
```
net_df['param ']
```

Out[309]:
```
0      Type
1    onboot
2     proto
3        ip
Name: param , dtype: object
```

In [310]:
```
net_df.rename(columns={'param ':'K1'})
```

Out[310]:

	K1	value
0	Type	ethernet
1	onboot	yes
2	proto	dhcp
3	ip	10.20.30.40

In [311]:
```
net_df # Note: There is no changes in the dataframe(net_df)
```

Out[311]:

	param	value
0	Type	ethernet
1	onboot	yes
2	proto	dhcp
3	ip	10.20.30.40

In [312]:
```
df4 = net_df.rename(columns={'param ':'K1'}) # we can new dataframe
df4
```

Out[312]:

	K1	value
0	Type	ethernet
1	onboot	yes
2	proto	dhcp
3	ip	10.20.30.40

In [313]:
```
net_df.rename(columns={'param ':'K1'},inplace=True)
# update the changes in net_df ->won't display results to monitor
```

In [314]:
```
net_df
```

Out[314]:

	K1	value
0	Type	ethernet
1	onboot	yes
2	proto	dhcp
3	ip	10.20.30.40

In [315]:
```
df3
```

Out[315]:

	eid	ename	edept	eplace	ecost
0	101	raj	sales	pune	1000
1	102	leo	prod	bglore	2000
2	103	paul	HR	chennai	3000
3	104	anu	hr	hyderabad	4000
4	456	kumar	sales	bglore	3000
5	105	zion	Hr	mumbai	5000
6	106	bibu	sales	bglore	1450
7	107	theeb	sales	noida	4590
8	108	bibu	sales	bglore	5000
9	113	kumar	prod	hyderabad	5423

In [316]:
```
pd.read_csv('E:\\emp.csv',index_col='edept')
```

Out[316]:

edept	eid	ename	eplace	ecost
sales	101	raj	pune	1000
prod	102	leo	bglore	2000
HR	103	paul	chennai	3000
hr	104	anu	hyderabad	4000
sales	456	kumar	bglore	3000
Hr	105	zion	mumbai	5000
sales	106	bibu	bglore	1450
sales	107	theeb	noida	4590
sales	108	bibu	bglore	5000
prod	113	kumar	hyderabad	5423

In [317]:
```
pd.read_csv('E:\\emp.csv',index_col='eplace')
```

Out[317]:

eplace	eid	ename	edept	ecost
pune	101	raj	sales	1000
bglore	102	leo	prod	2000
chennai	103	paul	HR	3000
hyderabad	104	anu	hr	4000
bglore	456	kumar	sales	3000
mumbai	105	zion	Hr	5000
bglore	106	bibu	sales	1450
noida	107	theeb	sales	4590
bglore	108	bibu	sales	5000
hyderabad	113	kumar	prod	5423

In [318]:
```
pd.read_csv('E:\\emp.csv',index_col='eid')
```

Out[318]:

eid	ename	edept	eplace	ecost
101	raj	sales	pune	1000
102	leo	prod	bglore	2000
103	paul	HR	chennai	3000
104	anu	hr	hyderabad	4000
456	kumar	sales	bglore	3000
105	zion	Hr	mumbai	5000
106	bibu	sales	bglore	1450
107	theeb	sales	noida	4590
108	bibu	sales	bglore	5000
113	kumar	prod	hyderabad	5423

In [319]:
```
df4 = pd.read_csv('E:\\emp.csv',index_col='eid')
```

In [320]:
```
df4.index
```

Out[320]:
```
Index([101, 102, 103, 104, 456, 105, 106, 107, 108, 113], dtype='int64', name
='eid')
```

In [321]:
```
df4.loc[103]
```

Out[321]:
```
ename        paul
edept          HR
eplace    chennai
ecost        3000
Name: 103, dtype: object
```

In [322]:
```
df5 = pd.read_csv('E:\\emp.csv',index_col='edept')
df5
```

Out[322]:

edept	eid	ename	eplace	ecost
sales	101	raj	pune	1000
prod	102	leo	bglore	2000
HR	103	paul	chennai	3000
hr	104	anu	hyderabad	4000
sales	456	kumar	bglore	3000
Hr	105	zion	mumbai	5000
sales	106	bibu	bglore	1450
sales	107	theeb	noida	4590
sales	108	bibu	bglore	5000
prod	113	kumar	hyderabad	5423

In [323]:
```
df5.loc['sales']
```

Out[323]:

edept	eid	ename	eplace	ecost
sales	101	raj	pune	1000
sales	456	kumar	bglore	3000
sales	106	bibu	bglore	1450
sales	107	theeb	noida	4590
sales	108	bibu	bglore	5000

In [324]:
```
df5.loc['sales'][['ename','eplace']]
```

Out[324]:

edept	ename	eplace
sales	raj	pune
sales	kumar	bglore
sales	bibu	bglore
sales	theeb	noida
sales	bibu	bglore

In [326]:
```
pd.read_csv('E:\\emp.csv',index_col=2)
```

Out[326]:

edept	eid	ename	eplace	ecost
sales	101	raj	pune	1000
prod	102	leo	bglore	2000
HR	103	paul	chennai	3000
hr	104	anu	hyderabad	4000
sales	456	kumar	bglore	3000
Hr	105	zion	mumbai	5000
sales	106	bibu	bglore	1450
sales	107	theeb	noida	4590
sales	108	bibu	bglore	5000
prod	113	kumar	hyderabad	5423

In [327]:
```
df5 = pd.read_csv('E:\\emp.csv',index_col=2)
df5.loc['sales']
```

Out[327]:

edept	eid	ename	eplace	ecost
sales	101	raj	pune	1000
sales	456	kumar	bglore	3000
sales	106	bibu	bglore	1450
sales	107	theeb	noida	4590
sales	108	bibu	bglore	5000

In [328]:
```
df5.loc['sales']['ecost'] >4500
```

Out[328]:
```
edept
sales    False
sales    False
sales    False
sales     True
sales     True
Name: ecost, dtype: bool
```

In [339]:
```
df5.loc['sales'][['ename','eplace']]
```

Out[339]:

edept	ename	eplace
sales	raj	pune
sales	kumar	bglore
sales	bibu	bglore
sales	theeb	noida
sales	bibu	bglore