In [ ]:
```
# set operation
# function
# module & package
# regx

# dict - Collection of unordered elements - key:value - mutable - keybased

# set  - Collection of unordered elements - key only  - not keybased ;not index b
# set is not like list,tuple,dict
# ---

# set operations - union,intersection,difference,symmetric_difference
# ----------------
#  |_ 1. method()
#  |_ 2. operators
```

In [6]:
```
s={"K1",10,20,10,20,10,20,10,20,"K2"} # Vs d={"Key":"Value"}
print(type(s))
print(s) # set not allowed duplicate elements
# set is not index based; set is not key:value
for var in s:
    print(var)

10 in s
```

```
<class 'set'>
{'K1', 10, 20, 'K2'}
K1
10
20
K2
```

Out[6]:
```
True
```

In [10]:
```
L=["D1","D2","D3"]
L.append("D1")
L.append("D2")
L.append("D1")
L.append("D2")
L
s=set(L) # typecast to set
L=list(s) # typecast to list
L
```

Out[10]:
```
['D2', 'D3', 'D1']
```

In [12]:
```
# set - collection of keys
#                       --->unique element
s={"D1","D2"}
print(len(s))

s.add("D3")
s.add(100)
s.add("D1")
s
```

```
2
```

Out[12]:
```
{100, 'D1', 'D2', 'D3'}
```

In [14]:
```
L=["Dx","Dy","Dz"]
# s.add(L) Error
s.update(L)
s
```

Out[14]:
```
{100, 'D1', 'D2', 'D3', 'Dx', 'Dy', 'Dz'}
```

In [17]:
```
s={10,20,30,40,50}
s.remove(10) # we can delete set element
print(s)
s.discard(20) # we can delete set element
s
```

```
{40, 50, 20, 30}
```

Out[17]:
```
{30, 40, 50}
```

In [19]:
```
s={"p1.log","p2.log"}
# s.remove("p4.log") ->Error
s.discard("p4.log") # ->None
```

In [22]:
```
L1=["d1","d2","d3","d4","d5"]
L2=["test1","test2","d3","test3","d4","test5"]

s1=set(L1)
s2=set(L2)
#                     method()  <or>   operator
# ------------------------------------------
# union           --> union()           |
# intersection    --> intersection()     &
# difference      --> difference()       -
# symmetric_difference  -->symmetric_difference()  ^
#

print(s1|s2) # union operation
print(s1.union(s2)) # union operation
```

```
{'test5', 'test3', 'd4', 'test2', 'd2', 'd5', 'test1', 'd3', 'd1'}
{'test5', 'test3', 'd4', 'test2', 'd2', 'd5', 'test1', 'd3', 'd1'}
```

In [23]:
```
# filter common data
print(s1&s2)
print(s1.intersection(s2))
```

```
{'d3', 'd4'}
{'d3', 'd4'}
```

In [25]:
```
# difference
print(s1-s2)
print(s2-s1)
```

```
{'d2', 'd5', 'd1'}
{'test5', 'test2', 'test3', 'test1'}
```

In [26]:
```
print(s1&s2) # common
print("")
print(s1^s2) # symmetric_difference
```

```
{'d3', 'd4'}

{'test5', 'd2', 'test3', 'd5', 'test1', 'test2', 'd1'}
```

In [27]:
```
D1=["p1.log","p2.log","p3.log","p1.sh","ab.txt","index.html"]
D2=["ab.c","p1.py","p2.log","p2.sh","test.json","ab.txt"]
s1=set(D1)
s2=set(D2)

UNION=s1|s2
COMM=s1&s2
Diff1=s1-s2
Diff2=s2-s1
SYM=s1^s2
print(UNION)
print("")
print(COMM)
print("")
print(Diff1)
print(Diff2)
print("")
print(SYM)
```

```
{'p1.py', 'p1.sh', 'ab.c', 'p1.log', 'p2.log', 'p2.sh', 'p3.log', 'index.html',
'ab.txt', 'test.json'}

{'p2.log', 'ab.txt'}

{'p1.log', 'p3.log', 'index.html', 'p1.sh'}
{'ab.c', 'p1.py', 'test.json', 'p2.sh'}

{'p1.sh', 'test.json', 'p1.py', 'ab.c', 'p1.log', 'p2.sh', 'p3.log', 'index.htm
l'}
```

In [ ]:

```
C:\Users\Karthikeyan>python
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 18 2019, 23:46:00) [MSC
(Intel)] on win32
Type "help", "copyright", "credits" or "license" for more informat
>>> import os
>>> os.listdir("D:\\")
['$RECYCLE.BIN', '2018-05-08_15-45-04.mp4', '2018-05-09_20-51-08(0
05-09_20-51-08.mp4', '4__original.mp4', 'Adv_PDFS', 'Anaconda3-202
86_64.exe', 'Ansible_Notes', 'B.py', 'BankStatements__1st_April_20
h2019', 'BankStatements__1st_April_2018_To31st_March2019.zip', 'C'
copy.mp4', 'D88168GC20_ag1.pdf', 'D88168GC20_ag2.pdf', 'D88168GC20
DB-1', 'Demo', 'emp.csv', 'emp.json', 'Fedora', 'Flask_work', 'Fre
 Joiner', 'Hot-Dell-Desktop', 'interface.log', 'joined-all(0).mp4'
mp4', 'KROSUM', 'mastermindSession', 'mongo', 'msdia80.dll', 'myfl
info.log', 'OL7', 'OOPs_Examples.tar', 'Oracle_Ruby', 'original.mp
 'PowerShell_Scripts__Salem_Nov_2020', 'Project', 'property.txt',
es', 'Python__Day5__docs', 'Python__Day5__docs.zip', 'r1.txt', 'r2
.txt', 'SHAREX', 'ShareX-12.2.0-setup.exe', 'System Volume Informa
 'test.html', 'test.txt', 'test1.png', 'thiruppavai', 'TILL_NOV_18
 'Ubuntu', 'Ubuntu.zip', 'VendorInfo.txt', 'Vendor_info.log', 'Won
 Converter Ultimate', 'zipfiles']
>>>
>>> os.listdir("C:\\")
['$Recycle.Bin', 'Dell', 'DELL LATITUDE E6420 WIN 7 PRO 64 BIT', '
Settings', 'freefallprotection.log', 'hiberfil.sys', 'Intel', 'M11
_Full_Solution', 'MSOCache', 'msys64', 'pagefile.sys', 'PerfLogs',
s', 'Program Files (x86)', 'ProgramData', 'Python27', 'Recovery',
'Ruby24-x64', 'service.csv', 'System Volume Information', 'Users',
>>>
>>>
>>> s={}
>>> type(s)
<class 'dict'>
>>> s=set()
>>> len(s)
0
>>> type(s)
<class 'set'>
>>>
>>> L1=os.listdir(".")
>>> L2=os.listdir("D:\\")
>>>
>>> s1=set(L1)
>>> s2=set(L2)
>>>
>>> common=s1&s2
>>> len(common)
0
>>> with open("emp.csv") as F:
...     pass
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'emp.csv'
>>> with open("emp.csv","w") as W:
...     W.write("test1,test2\n")
```

```
...
12
>>> L1=os.listdir(".")
>>> L2=os.listdir("D:\\")
>>> s1=set(L1)
>>> s2=set(L2)
>>> common=s1&s2
>>> len(common)
1
>>> common
{'emp.csv'}
>>>
>>>
```

In [34]:

```
# function
# function definition - codeblock - action block
# function call - to invoke a definition

# function definition
# def functionname():
#     ---------------
#     defintion section (or) operation section


# functionname() - simple function call
#

def display():
    print("List of mounted filesystem details:-")
    print("-"*45)
    print("df -Th")
    print("-"*45)
    print("Exit from display block")


#display()
```

In [ ]:
```
>>> L=[]
>>>
>>> L.append("D1")
>>> L.append(["D2","d3"])
>>>
>>> L.append()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: append() takes exactly one argument (0 given)
>>>
>>> L.append("D11","D12")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: append() takes exactly one argument (2 given)
>>>
>>> # def append(a1):
>>>
>>> def f1(a1,a2):
...     print(a1,a2)
...
>>> f1()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() missing 2 required positional arguments: 'a1' and 'a2'
>>> f1(10)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() missing 1 required positional argument: 'a2'
>>> f1(10,20)
10 20
>>> f1(10,20,30)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() takes 2 positional arguments but 3 were given
>>>
>>>
```

In [36]:
```
L=["D1","D2","D3","D4","D5","D6"]
# L.pop()
# L.pop(1)

# def pop(a1=-1)
```

Out[36]: 'D2'

In [41]:
```
def f1(a1,a2):
    print("Hello")
    print(a1,a2)

#f1(10,20)
#f1() -Error
#f1(100) - Error
#f1(10,20,30) - Error
```

In [46]:
```
def f2(a1=10,a2=20): # default arguments
    print(a1,a2)

f2()
f2(100)
f2(150,234)
#f2(10,20,30,40,50) # Error
```

```
10 20
100 20
150 234
```

In [51]:
```
def f1(a1,a2,a3=0,a4=0):
    print(a1,a2,a3,a4)
# f1() # Error
f1("AB","Test")
f1("AB","Test1","Test2","Test3")
#f1("AB","Test1","Test2","T3","T4","T5") - Error
```

```
AB Test 0 0
AB Test1 Test2 Test3
```

In [52]:
```
def f1(a=10,b):
    print("Hello")
```

```
  File "<ipython-input-52-7fcab63e999f>", line 1
    def f1(a=10,b):
           ^
SyntaxError: non-default argument follows default argument
```

In [58]:
```
d={}
d.setdefault("K1","V1")
d.setdefault("K2")
d
# d.setdefault() # Error
# d.setdefault("K3","V1","V2") # Error

# def setdefault(a1,a2=None):
```

Out[58]:
```
{'K1': 'V1', 'K2': None}
```

In [62]:
```
def connect(a1,user="root",db="mysql",password=None): # Required and Defaultargs
    print("connected")

connect("DEMO")
connect("DEMO","admin")
connect("DEMO","admin","sqlite3","PASSWORD")
```

```
connected
connected
connected
```

In [65]:
```
L=['bash','zsh','csh','tcsh','psh','ash','perl']
L.sort(reverse=True)
L
```

Out[65]:
```
['zsh', 'tcsh', 'psh', 'perl', 'csh', 'bash', 'ash']
```

In [66]:
```
L=[]
def f1(a1):
    if(a1>100):
        return a1+100

for var in [50,40,30,120]:
    rv=f1(var)
    L.append(rv)

L
```

Out[66]:
```
[None, None, None, 220]
```

In [ ]:

```
>>> L=[]
>>>
>>> L.append("D1")
>>> L.append(["D2","d3"])
>>>
>>> L.append()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: append() takes exactly one argument (0 given)
>>>
>>> L.append("D11","D12")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: append() takes exactly one argument (2 given)
>>>
>>> # def append(a1):
>>>
>>> def f1(a1,a2):
...     print(a1,a2)
...
>>> f1()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() missing 2 required positional arguments: 'a1' and 'a2'
>>> f1(10)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() missing 1 required positional argument: 'a2'
>>> f1(10,20)
10 20
>>> f1(10,20,30)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() takes 2 positional arguments but 3 were given
>>>
>>> def f1(a1=10,b):
...     print("Hello")
...
  File "<stdin>", line 1
SyntaxError: non-default argument follows default argument
>>> def f1(a1,a2,a3,a4,a5=0,a6=0,a7=0):
...     pass
...
>>> def f1(a1=10,a2=20):
...     print("Hello")
...
>>> f1()
Hello
>>> f1(100)
Hello
>>> f1(10,20)
Hello
>>> f1(1,2,3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() takes from 0 to 2 positional arguments but 3 were given
```

```
>>>
>>> def fx(*args):
...     print("Hello")
...     print(args)
...     print(type(args))
...
>>> fx(10,20,30,40)
Hello
(10, 20, 30, 40)
<class 'tuple'>
>>>
>>> fx()
Hello
()
<class 'tuple'>
>>>
>>> def fx(*a1):
...     print(a1)
...
>>> fx()
()
>>> def f1(*a):
...     print(a)
...
>>> f1(10,20,30,40)
(10, 20, 30, 40)
>>>
>>> f1(["D1","D2"],("T1","T2"),{"K1":"V1"})
(['D1', 'D2'], ('T1', 'T2'), {'K1': 'V1'})
>>>
>>> def f1(**a):
...     print(a)
...
>>> f1(var=100,db='oracle',user='root',ip='10.20.30.40') # call
{'var': 100, 'db': 'oracle', 'user': 'root', 'ip': '10.20.30.40'}
>>>
>>>
>>>
>>> def f1(a1=10): # default args
...     print(a1)
...
>>> f1()
10
>>> # f1(a1=10) <== keyword arguments
>>>
>>> def f1(*args):
...     print(args)
...
>>> f1({"a":10})
({'a': 10},)
>>>
>>> f1(a=10)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f1() got an unexpected keyword argument 'a'
>>> #Threading.Thread(target=threadname,args=(a1,a2))
>>>                   ------------------============ //keyword arguments
```

```
  File "<stdin>", line 1
    ------------------============ //keyword arguments
    ^
IndentationError: unexpected indent
>>>
>>> def f1(*args,**kwargs):
...     print("Hello")
...
>>> f1()
Hello
>>> f1(12,23,23,2,323,2,31,3234)
Hello
>>> f1(Var=10,v2=34)
Hello
>>>
>>> def f1(**a,*b):
  File "<stdin>", line 1
    def f1(**a,*b):
               ^
SyntaxError: invalid syntax
>>>
>>> def f1(*a1,**a2):
...     print(a1,a2)
...
>>> f1(123,23,3,31,,12,12,12,12,322)
  File "<stdin>", line 1
    f1(123,23,3,31,,12,12,12,12,322)
                   ^
SyntaxError: invalid syntax
>>>
>>> f1(123,23,3,31,12,12,12,12,322)
(123, 23, 3, 31, 12, 12, 12, 12, 322) {}
>>>
>>> f1(user="root",db="mysql",port=32443)
() {'user': 'root', 'db': 'mysql', 'port': 32443}
>>>
>>> f1(123,23,3,31,12,12,12,12,322,user="root")
(123, 23, 3, 31, 12, 12, 12, 12, 322) {'user': 'root'}
>>>
>>> def f1():
...     global v1,v2
...     v1=100
...     v2=200
...     v3=300 # local variable
...     v4='data' # local variable
...     print(v1,v2,v3,v4)
...
>>> f1()
100 200 300 data
>>> v1
100
>>> v2
200
>>> v3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'v3' is not defined
```

```
>>> v4
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'v4' is not defined
>>> def f2():
...     print(v1,v2)
...
>>> f2()
100 200
>>>
>>> def f1():
...     var=100
...     return var
...
>>> f1()
100
>>> def f2():
...     var=234
...
>>> f2()
>>> f2() == None
True
>>>
>>>
>>> rv=f1()
>>> rv
100
>>> print(f1())
100
>>> rv=f1()
>>> print(rv)
100
>>>
>>> f1()
```

In [69]:
```
def f1():
    return 10

#f1()
def f2():
    return 10,
f2() # tuple

v1=10,
type(v1)
```

Out[69]:
```
tuple
```

In [ ]:
```
# module
# |__ existing python file(.py)
#

D:\>file:ab.py              D:\>file:p1.py          D:\>file:p2.py
=================           ===============         ===============
port=80                 fname="p1.log"          import ab
def f1():               print(fname)            print(ab.port)
    print("list of files")      def fx():           ================
====================                print("Display block")  D:\>python p2.py
                    fx()                80
D:\>python ab.py{Enter}         ==========================
Empty                   D:\>python p1.py{Enter}
                    p1.log
                    Display block

     |
 loadable(or) runnable file     running file (or) script file
 ..........................

import <filename>

filename.member
    (variable,function,class etc.,)

import os
os.system("command")
os.listdir(".")
os.getcwd()



import file1,file2,file3

import os,json,pprint,re
  (or)
import os
import json
import pprint
import re

file: D:\>ab.py         file: E:\>p1.py     file: D:\>p2.py
------------------      --------------------    -----------------
port=80             import ab       import ab
------------------      print(ab.port)      print(ab.port)
D:\>python ab.py        ----------------------  -----------------
Empty               E:\>python p1.py    D:\>python p2.py
                Error           80
                |
                ModuleNotFound (or) Import Error

import sys

os - os commands
pprint - display complex ds - dumper format
json - jsonparsing /jsondata
```

```
sys - python info - version,modules,modulepath etc.,

import sys
sys.path ->[  ]

PYTHONPATH

import modulename
modulename.member

import module
module.member


import os
help(os)

import sys
help(sys)


import filename

1. search the path -->refer sys.path
2. pvm ->filename.py -->filename.pyc
            ------------
             bytecode

import filename
filename.member
>>> import sys
>>> import openpyxl
>>>
>>> openpyxl in sys.modules
False
>>> openpyxl
<module 'openpyxl' from 'C:\\Users\\Karthikeyan\\AppData\\Local\\Programs\\Pytho
n\\Python37-32\\lib\\site-packages\\openpyxl\\__init__.py'>
>>>
>>> openpyxl in sys.modules
False
>>>
>>> sys.modules['openpyxl']
<module 'openpyxl' from 'C:\\Users\\Karthikeyan\\AppData\\Local\\Programs\\Pytho
n\\Python37-32\\lib\\site-packages\\openpyxl\\__init__.py'>
>>>
>>> import openpyxl
>>>
>>>

>>> os.mkdir("Dx")
>>>
>>> os.chdir('Dx')
>>> os.getcwd()
'C:\\Users\\Karthikeyan\\Dx'
>>> os.listdir(".")
```

```
[]
>>> os.chdir("..")
>>> os.getcwd()
'C:\\Users\\Karthikeyan'
>>> os.chdir("Dx")
>>> os.mkdir("D1")
>>> os.listdir(".")
['D1']
>>> os.chdir("D1")
>>> os.getcwd()
'C:\\Users\\Karthikeyan\\Dx\\D1'
>>>
>>> os.chdir("..")
>>> os.rmdir("D1")
>>>
>>> os.system("powershell get-process python")

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
    624      88    63476      74500      10.73    872   1 python
    221      74    45040      49440       1.00   1536   1 python
     61      11     5160       8192       0.11   4148   1 python
     66      12     5408       8560       0.11   6376   1 python
    225      90    82816      87864       6.49   7440   1 python


0
>>>
```

In [71]:
```
d={}
s='interface=eth0'
K,V=s.split("=")
d[K]=V
d
```

Out[71]:
```
{'interface': 'eth0'}
```

In [72]:
```
d={} # empty dict
print(d,len(d))

with open("D:\\property.txt") as FH: # open an existing property.txt file
    for var in FH.readlines():  # reading datafrom file - line by line
        var=var.strip() # remove \n char
        K,V=var.split("=") # split eachline into multiplevalues
        d[K]=V # adding data to dict

print(d,len(d))
```

```
{} 0
{'interface': 'eth0', 'IP': '10.20.30.40', 'Subnet': '24', 'onboot': 'None', 'I
PADD': 'IPV4', 'domain': 'example.com', 'prefix': 'no', 'DNS1': '134.565.423.45
6'} 8
```

In [75]:
```
with open("D:\\property.txt") as FH: # open an existing property.txt file
    for var in FH.readlines():  # reading datafrom file - line by line
        var=var.strip() # remove \n char
        K,V=var.split("=") # split eachline into multiplevalues
        d[K]=V # adding data to dict

print("Key/Value details:-")

for var in d:
    print("{}\t{}".format(var,d[var]))

d['interface']='eth1'  # dict modification
d['onboot']='dhcp'
d['prefix']='yes'
d['DNS2']='134.553.342.442' # adding newdata to existing dict
print("\nUpdated dict details:-")

for var in d:
    print("{}\t{}".format(var,d[var])) # display updated dict key/value

with open("D:\\newproperty.txt","w") as WH:  # create a new file
    for var in d:
        WH.write("{}={}\n".format(var,d[var])) # Write data to newFILE
```

```
Key/Value details:-
interface       eth0
IP      10.20.30.40
Subnet  24
onboot  None
IPADD   IPV4
domain  example.com
prefix  no
DNS1    134.565.423.456
DNS2    134.553.342.442

Updated dict details:-
interface       eth1
IP      10.20.30.40
Subnet  24
onboot  dhcp
IPADD   IPV4
domain  example.com
prefix  yes
DNS1    134.565.423.456
DNS2    134.553.342.442
```

In [ ]:
```
# module - existing python file(filename.py (or) filename.pyc)
# -----
# project/p1.py  p2.py  p3.py  .. p50.py
#
# import p1,p2,p3,p4,p5,...p50
# (or)
# import p1
# import p2
#  ..
# import p50

# package - Collection of modules ( collection of .py files)
# ----------------------------------------------------------
# OS view -> module - reg.file
# ------------------------------->Package  ->directory (or) Folder file

# commandline steps

1. create a folder(or)directory
2. collect list of .py files into  folder(or)directory
3. create a package initialization(specialfile) file __init__.py
4. import all external symbols to __init__file
   from module import *

5. test your package -> import <directoryname>
```

In [ ]:
```
:ab.py                  file:p1.py
---------                   --------------
=80                     import ab
f1():                   fname="p1.log"
print("display block")      print(ab.port); print(port)->Error
-----------------------------   print(fname)
                    ab.f1()
                    -----------------
                        |
ol(or)dict table                |
-----------------           __main__.fname|p1.log
                    ------------------------
ain__.port | 80          ab.port|80
---------------------       ---------------
ain__.f1   | 0x12345        ab.f1|0x12345
---------------------       -------------------------

'K1":"V1"}

@host~]# ls
 b.py
@host~]# cat a.py
ess
@host~]# cat /etc/passwd    -----------  import module
ess                         module.member
@host~]# cat passwd
r
@host~]# cp /etc/passwd .   ------- from module import <member>
@host~]# cat passwd         from ab import port
ess                     print(port)

                    from tkinter import *


 ERP.sales import f1
 ERP.CRM.customer import display

/
_CRM/
   |_customer.py
-------------
def display():
    ....


================================================================================
```

In [76]:
```
import pprint
pprint.pprint(d)
```

```
{'DNS1': '134.565.423.456',
 'DNS2': '134.553.342.442',
 'IP': '10.20.30.40',
 'IPADD': 'IPV4',
 'Subnet': '24',
 'domain': 'example.com',
 'interface': 'eth1',
 'onboot': 'dhcp',
 'prefix': 'yes'}
```

In [77]:
```
import pprint as p
p.pprint(d)
```

```
{'DNS1': '134.565.423.456',
 'DNS2': '134.553.342.442',
 'IP': '10.20.30.40',
 'IPADD': 'IPV4',
 'Subnet': '24',
 'domain': 'example.com',
 'interface': 'eth1',
 'onboot': 'dhcp',
 'prefix': 'yes'}
```

In [ ]:

```
file:ab.py                file:p1.py
-------------             --------------
port=80                   import ab
def f1():                 fname="p1.log"
    print("display block")    print(ab.port); print(port)->Error
-------------------------------   print(fname)
 |                  ab.f1()
 |                  -----------------
 |                      |
symbol(or)dict table                |
--------------------          __main__.fname|p1.log
                   ------------------------
__main__.port | 80          ab.port|80
-------------------------     ---------------
__main__.f1   | 0x12345       ab.f1|0x12345
-------------------------     -------------------------

d={"K1":"V1"}

root@host~]# ls
a.py b.py
root@host~]# cat a.py
Success
root@host~]# cat /etc/passwd    -----------  import module
Success                      module.member
root@host~]# cat passwd
Error
root@host~]# cp /etc/passwd .   ------- from module import <member>
root@host~]# cat passwd         from ab import port
Success                  print(port)

                   from tkinter import *


from ERP.sales import f1
from ERP.CRM.customer import display

ERP/
 |__CRM/
      |_customer.py
    -------------
    def display():
        ....


================================================================================
```

In [ ]:

```
apelix@krosumlabs:~/Temp$ mkdir ERP
apelix@krosumlabs:~/Temp$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import ERP
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named ERP
>>> exit()
apelix@krosumlabs:~/Temp$ cd ERP
apelix@krosumlabs:~/Temp/ERP$ vi sales.py
apelix@krosumlabs:~/Temp/ERP$ vi prod.py
apelix@krosumlabs:~/Temp/ERP$ vi fi.py
apelix@krosumlabs:~/Temp/ERP$
```

```
apelix@krosumlabs:~/Temp/ERP$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> class Box:
...     var=100
...     def f1():
...             print("Hello")
...
>>> Box.__dict__
{'var': 100, 'f1': <function f1 at 0xb77713e4>, '__module__': '__main__', '__doc_
>>>
apelix@krosumlabs:~/Temp/ERP$
```

```
apelix@krosumlabs:~/Temp/ERP$ ls
```

```
fi.py  prod.py  sales.py
apelix@krosumlabs:~/Temp/ERP$ cat sales.py
def f1():
    print("sales count:123")
apelix@krosumlabs:~/Temp/ERP$ cat prod.py
def f2():
    print("production details:")
apelix@krosumlabs:~/Temp/ERP$ cat fi.py
def f3():
    return 1000
apelix@krosumlabs:~/Temp/ERP$ cat >__init__.py
from sales import f1
from prod import *
from fi import *
apelix@krosumlabs:~/Temp/ERP$ ls
fi.py  __init__.py  prod.py  sales.py
apelix@krosumlabs:~/Temp/ERP$ cd ..
apelix@krosumlabs:~/Temp$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import ERP
>>> ERP.f1()
sales count:123
>>> ERP.f2()
production details:
>>> ERP.f3()
1000
>>> ERP
<module 'ERP' from 'ERP/__init__.py'>
>>>
apelix@krosumlabs:~/Temp$ ls ERP
fi.py  fi.pyc  __init__.py  __init__.pyc  prod.py  prod.pyc  sales.py  sales.pyc
apelix@krosumlabs:~/Temp$
apelix@krosumlabs:~/Temp$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import ERP
>>> help(ERP)

>>>
apelix@krosumlabs:~/Temp$ ls ERP
fi.py  fi.pyc  __init__.py  __init__.pyc  prod.py  prod.pyc  sales.py  sales.pyc
apelix@krosumlabs:~/Temp$
apelix@krosumlabs:~/Temp$
apelix@krosumlabs:~/Temp$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import ERP
>>> ERP.f1()
sales count:123
>>>
>>> from ERP import f1
>>> f1()
sales count:123
```

```
>>>
apelix@krosumlabs:~/Temp$ ls ERP
fi.py  fi.pyc  __init__.py  __init__.pyc  prod.py  prod.pyc  sales.py  sales.pyc
apelix@krosumlabs:~/Temp$
```

In [ ]:
```
# Regular Expression (Regx)
# --------------------------
# search,substitute
# input validation ex: n=input("Enter any two digits:")
#                      Enter any two digits:___

# string+list+conditional+loop+filehandling+function+module //programming style

# shellscript            vs   python
# command                     program = filehandling+conditional+loop+function()
# grep ;sed;awk

import re
re.search() --> re.search("Pattern","input_string") -><Ack>/None # validation
re.findall() --> re.findall("Pattern","input_string") ->[matched_result]/[]




grep/findstr

open a existing file  ==============> FH=open("inputfile")
read the contents line by line        L=FH.readlines()
search the pattern from input         if(re.search("pattern","input")):
print - matched pattern lines                    print()



>>> import re
>>> with open('D:\\emp.csv') as FH:
...     for var in FH.readlines():
...             if(re.search('sales',var)):
...                     print(var.strip())
...
ram,sales,pune,1000
xerox,sales,chennai,45900
theeb,sales,hyd,5678
>>>
>>> with open('D:\\emp.csv') as FH:
...     for var in FH.readlines():
...             if(re.search('sales',var)):
...                     print(var)
...
ram,sales,pune,1000

xerox,sales,chennai,45900

theeb,sales,hyd,5678

apelix@krosumlabs:~/Temp$ grep bash /etc/passwd
root:x:0:0:root:/root:/bin/bash
apelix:x:1000:1000:karthikeyan,,,:/home/apelix:/bin/bash
apelix@krosumlabs:~/Temp$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
```

```
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import re
>>>
>>> with open("/etc/passwd") as FH:
...     for var in FH.readlines():
...             var=var.strip()
...             if(re.search("bash",var)):
...                     print(var)
...
root:x:0:0:root:/root:/bin/bash
apelix:x:1000:1000:karthikeyan,,,:/home/apelix:/bin/bash
>>>
```

```
BRE                                     ERE
---Single                               ----Multiple pattern

^                                       | ( ) + {}
$
^pattern$
.
.*
[]
^[]
[]$
[^]
^$

^pattern - line starts with pattern

\s - space

pattern$ - line ends with pattern

\s$ - line ends with space

^\s - line starts with space

^pattern$ - pattern only


>>> import re
>>>
>>> re.search("sales","ram,sales,pune")
<re.Match object; span=(4, 9), match='sales'>
>>>
>>> re.search("^sales","ram,sales,pune")
>>>
>>> re.search("^sales","sales,pune")
<re.Match object; span=(0, 5), match='sales'>
>>>
>>> re.search("^5","5sales,pune")
<re.Match object; span=(0, 1), match='5'>
>>>
>>> re.search("^5","sales5,pune")
>>>
>>> re.search("^5"," 5sales,pune")
```

```
>>> re.search("^\s"," 5sales,pune")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("sales$","sdfadsfasddsf sales")
<re.Match object; span=(14, 19), match='sales'>
>>>
>>> re.search("sales$","sdfadsfasddsf sales ")
>>>
>>> re.search("\s$","sdfadsfasddsf sales ")
<re.Match object; span=(19, 20), match=' '>
>>>
>>> s1="sales,kumar"
>>> s2="sales,"
>>> s3="sales"
>>> s4="sales "
>>> s5="arun,sales"
>>>
>>> re.search("^sales$",s1)
>>> re.search("^sales$",s2)
>>> re.search("^sales$",s3)
<re.Match object; span=(0, 5), match='sales'>
>>>
>>> re.search("^sales$",s4)
>>> re.search("^sales$",s5)
>>> re.search("^sales$","saleskumarsales")
>>>
>>> re.search("^sales$","salessales")
>>>
>>> # re.search("^Pattern","inputstring")
>>> # re.search("Pattern$","inputstring")
>>> # re.search("^Pattern$","inputstring")
>>> #


>>> with open("/etc/passwd") as FH:
...     for var in FH.readlines():
...             if(re.search("^a",var)):
...                     print(var.strip())
...
avahi-autoipd:x:105:112:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:106:113:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
apelix:x:1000:1000:karthikeyan,,,:/home/apelix:/bin/bash
>>>
```

In [ ]:
```
# .(dot) -> it matching any single character , except \n char

# re.search("^..","input") - line starts with any two chars

character based search
character class []

[Aa]run
---------> Arun  arun //matched

[Aav][Rr]un
--------------->Arun arun vrun
                 ARun aRun vRun

[A-Z] - any single uppercase chars
[a-z] - any single lowercase chars

^[A-Z] - line starts with any single uppercase chars
^[a-z] - line starts with any single lowercase chars

^[A-Za-z] - line starts with any alpha


[a-z]$ - line ends with any lowercase chars
[A-Za-z]$ - line ends with any alpha

\s$ - line ends with space
^\s - line starts with space
\*

[0-9] - any digits

^[0-9] - line starts with digits
[0-9]$ - line ends with digits

[A-Za-z0-9] - match alpha and number
aix
OL5
Temp
aT
re.search("[aT5]","input") - any where char 'a' ->True
                                          'T' ->True
                                          '5' ->True

re.search("^[aT5]","input") -> aix
                               -
                               Temp
                               -
                               aT
                               -
re.search("[aT5]$","input") - line ends with a T 5
                        OL5
                          -
                         aT
                          -
```

```
re.search("[^aT5]","input") - NOT matching 'a' 'T' '5'
           ======                          ------------
            aix
             --
            OL5
            --
            Temp
             ---

re.search("[^A-Za-z0-9\s]") - specialchars  <=== re.search("[^\w\s]","input")
re.search("[^\s]") - not matching space

[0-9] -->\d
[A-Za-z0-9] -->\w



>>> import re
>>>
>>> re.search("sales","ram,sales,pune")
<re.Match object; span=(4, 9), match='sales'>
>>>
>>> re.search("^sales","ram,sales,pune")
>>>
>>> re.search("^sales","sales,pune")
<re.Match object; span=(0, 5), match='sales'>
>>>
>>> re.search("^5","5sales,pune")
<re.Match object; span=(0, 1), match='5'>
>>>
>>> re.search("^5","sales5,pune")
>>>
>>> re.search("^5"," 5sales,pune")
>>> re.search("^\s"," 5sales,pune")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("sales$","sdfadsfasddsf sales")
<re.Match object; span=(14, 19), match='sales'>
>>>
>>> re.search("sales$","sdfadsfasddsf sales ")
>>>
>>> re.search("\s$","sdfadsfasddsf sales ")
<re.Match object; span=(19, 20), match=' '>
>>>
>>> s1="sales,kumar"
>>> s2="sales,"
>>> s3="sales"
>>> s4="sales "
>>> s5="arun,sales"
>>>
>>> re.search("^sales$",s1)
>>> re.search("^sales$",s2)
>>> re.search("^sales$",s3)
<re.Match object; span=(0, 5), match='sales'>
>>>
>>> re.search("^sales$",s4)
>>> re.search("^sales$",s5)
```

```
>>> re.search("^sales$","saleskumarsales")
>>>
>>> re.search("^sales$","salessales")
>>>
>>> # re.search("^Pattern","inputstring")
>>> # re.search("Pattern$","inputstring")
>>> # re.search("^Pattern$","inputstring")
>>> #
>>>
>>>
>>> s2
'sales,'
>>> s3
'sales'
>>> s4
'sales '
>>>
>>>
>>> re.search("^sales.$",s2)
<re.Match object; span=(0, 6), match='sales,'>
>>> re.search("^sales.$",s3)
>>>
>>> re.search("^sales.$",s4)
<re.Match object; span=(0, 6), match='sales '>
>>>
>>> n=input("Enter any two digits:")
Enter any two digits:5
>>> n=input("Enter any two digits:")
Enter any two digits:sadfsdaf
>>> n=input("Enter any two digits:")
Enter any two digits:3232
>>>
>>> re.search("^[0-9][0-9]$","5")
>>> re.search("^[0-9][0-9]$",str(5))
>>>
>>> re.search("^[0-9][0-9]$",str(56))
<re.Match object; span=(0, 2), match='56'>
>>> re.search("^[0-9][0-9]$",str(567))
>>> re.search("^[0-9][0-9]$","abc")
>>>
>>> re.search("^\d\d$","45")
<re.Match object; span=(0, 2), match='45'>
>>>
>>> # ^[A-Z][0-9][0-9][0-9]$
>>> #
>>> # ^[A-Z]\d\d\d$
>>>
>>> # ^$ - empty line
>>>
>>> re.search("sales","kumar,sales,pune,1234")
<re.Match object; span=(6, 11), match='sales'>
>>>
>>> re.findall("sales","kumar,sales,pune,1234")
['sales']
>>>
>>> re.findall("^[0-9][0-9]","123-code-sales")
['12']
```

```
>>>
>>> re.search("SALES","sales")
>>>
>>> re.search("SALES","sales",re.I)
<re.Match object; span=(0, 5), match='sales'>
>>>
>>> re.findall("SALES","sales",re.I)
['sales']
>>> help(re.search)
Help on function search in module re:

search(pattern, string, flags=0)
    Scan through string looking for a match to the pattern, returning
    a Match object, or None if no match was found.

>>>

>>> import re
>>>
>>> # | () + {}
>>>
>>> # | - alternate pattern - like logical or
>>> #
>>> # pattern1|pattern2 - any one pattern is matched -True
>>>
>>> re.search("^\s"," sfasdf")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("[adf]$","sdfd")
<re.Match object; span=(3, 4), match='d'>
>>>
>>> re.search("^\s|[adf]$","4sdfds")
>>>
>>> re.search("^\s|[adf]$"," 4sdfds")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("^\s|[adf]$","4sdfdsf")
<re.Match object; span=(6, 7), match='f'>
>>>
>>> # (pattern1)(pattern2) - both pattern shouldmatch - same order
>>> #                          like logical and (P1)(P2)
>>> #
>>> re.search("(sales)(prod)","emp sales and prod depts")
>>>
>>> # salesprod
>>>
>>> re.search("(sales).*(prod)","emp sales and prod depts")
<re.Match object; span=(4, 18), match='sales and prod'>
>>> re.findall("(sales).*(prod)","emp sales and prod depts")
[('sales', 'prod')]
>>> L=re.findall("(sales).*(prod)","emp sales and prod depts")
>>> L[0]
('sales', 'prod')
>>> L[0][0]
'sales'
>>>
>>>
```

```
>>>
>>> re.findall("sales.*prod","emp sales and prod depts")
['sales and prod']
>>>
>>> var="SAMPLE data on 15th sep 2020 yes code134"
>>>
>>> re.findall("[0-9]",var)
['1', '5', '2', '0', '2', '0', '1', '3', '4']
>>>
>>> re.findall("[0-9].*",var)
['15th sep 2020 yes code134']
>>>
>>> re.findall("[0-9].*[0-9]",var)
['15th sep 2020 yes code134']
>>>
>>> var="SAMPLE data on 15th sep 2020 yes code134rs"
>>> re.findall("[0-9].*[0-9]",var)
['15th sep 2020 yes code134']
>>>
>>> # <Pattern>+  - 1 or more
>>> # -----------
>>>
>>> # ab+c -->abc abbbbbbbbbbbbbbbbbbc //matched
>>>
>>> # abbbbbbbbbb,bbbbbbbbbbc //not-matched
>>>
>>> re.findall("[0-9]",var)
['1', '5', '2', '0', '2', '0', '1', '3', '4']
>>>
>>> re.findall("[0-9]+",var)
['15', '2020', '134']
>>> # ^\s+
>>>
>>> re.search("^[A-Z]\d+[a-z]$","D5s")
<re.Match object; span=(0, 3), match='D5s'>
>>>
>>> re.search("^[A-Z]\d+[a-z]$","D323432432324325s")
<re.Match object; span=(0, 17), match='D323432432324325s'>
>>> re.search("^[A-Z]\d+[a-z]$","D32343243232,4325s")
>>>
>>>
>>> # <Pattern>{n} - n times
>>> # ab{2}c ---------------------->abbc //matched
>>> #          abc abbbc //not-matched
>>>
>>> re.findall("^[A-Z]\d\d\d[a-z][a-z]$","F345rg")
['F345rg']
>>>
>>> re.findall("^[A-Z]\d{3}[a-z]{2}$","F345rg")
['F345rg']
>>>
>>>
>>> # <Pattern>{n,} - minimum 'n' times - maximum nolimit
>>> # ..............
>>>
>>> # ab{2,}c --> abbc abbbbbbbbbbbbbbbbbbbbbbc //matched
>>>
```

```
>>> # ab+c --same as -- ab{1,}c
>>>
>>> re.findall("\d{3,}","sadfsa34dfs4sdfs56788fd")
['56788']
>>>
>>> # <Pattern>{n,m} - minimum 'n' times - maximum 'm' times
>>>
>>> # ab{2,4}c --> abbc abbbc abbbbc //matched
>>> #
>>> # ^[A-Z]\d{3,4}[a-z]$
>>> # ^.*\s{2,5}
>>>




>>> import re
>>>
>>> # | () + {}
>>>
>>> # | - alternate pattern - like logical or
>>> #
>>> # pattern1|pattern2 - any one pattern is matched -True
>>>
>>> re.search("^\s"," sfasdf")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("[adf]$","sdfd")
<re.Match object; span=(3, 4), match='d'>
>>>
>>> re.search("^\s|[adf]$","4sdfds")
>>>
>>> re.search("^\s|[adf]$"," 4sdfds")
<re.Match object; span=(0, 1), match=' '>
>>>
>>> re.search("^\s|[adf]$","4sdfdsf")
<re.Match object; span=(6, 7), match='f'>
>>>
>>> # (pattern1)(pattern2) - both pattern shouldmatch - same order
>>> #                          like logical and (P1)(P2)
>>> #
>>> re.search("(sales)(prod)","emp sales and prod depts")
>>>
>>> # salesprod
>>>
>>> re.search("(sales).*(prod)","emp sales and prod depts")
<re.Match object; span=(4, 18), match='sales and prod'>
>>> re.findall("(sales).*(prod)","emp sales and prod depts")
[('sales', 'prod')]
>>> L=re.findall("(sales).*(prod)","emp sales and prod depts")
>>> L[0]
('sales', 'prod')
>>> L[0][0]
'sales'
>>>
>>>
>>>
```

```
>>> re.findall("sales.*prod","emp sales and prod depts")
['sales and prod']
>>>
>>> var="SAMPLE data on 15th sep 2020 yes code134"
>>>
>>> re.findall("[0-9]",var)
['1', '5', '2', '0', '2', '0', '1', '3', '4']
>>>
>>> re.findall("[0-9].*",var)
['15th sep 2020 yes code134']
>>>
>>> re.findall("[0-9].*[0-9]",var)
['15th sep 2020 yes code134']
>>>
>>> var="SAMPLE data on 15th sep 2020 yes code134rs"
>>> re.findall("[0-9].*[0-9]",var)
['15th sep 2020 yes code134']
>>>
>>> # <Pattern>+  - 1 or more
>>> # -----------
>>>
>>> # ab+c -->abc abbbbbbbbbbbbbbbbbbc //matched
>>>
>>> # abbbbbbbbbb,bbbbbbbbbbc //not-matched
>>>
>>> re.findall("[0-9]",var)
['1', '5', '2', '0', '2', '0', '1', '3', '4']
>>>
>>> re.findall("[0-9]+",var)
['15', '2020', '134']
>>> # ^\s+
>>>
>>> re.search("^[A-Z]\d+[a-z]$","D5s")
<re.Match object; span=(0, 3), match='D5s'>
>>>
>>> re.search("^[A-Z]\d+[a-z]$","D323432432324325s")
<re.Match object; span=(0, 17), match='D323432432324325s'>
>>> re.search("^[A-Z]\d+[a-z]$","D32343243232,4325s")
>>>
>>>
>>> # <Pattern>{n} - n times
>>> # ab{2}c ---------------------->abbc //matched
>>> #          abc abbbc //not-matched
>>>
>>> re.findall("^[A-Z]\d\d\d[a-z][a-z]$","F345rg")
['F345rg']
>>>
>>> re.findall("^[A-Z]\d{3}[a-z]{2}$","F345rg")
['F345rg']
>>>
>>>
>>> # <Pattern>{n,} - minimum 'n' times - maximum nolimit
>>> # ..............
>>>
>>> # ab{2,}c --> abbc abbbbbbbbbbbbbbbbbbbbbbc //matched
>>>
>>> # ab+c --same as -- ab{1,}c
```

```
>>>
>>> re.findall("\d{3,}","sadfsa34dfs4sdfs56788fd")
['56788']
>>>
>>> # <Pattern>{n,m} - minimum 'n' times - maximum 'm' times
>>>
>>> # ab{2,4}c --> abbc abbbc abbbbc //matched
>>> #
>>> # ^[A-Z]\d{3,4}[a-z]$
>>> # ^.*\s{2,5}
>>>
>>>
>>> for v in os.listdir("D:\\"):
...     if(re.search(".html$|.py$",v)):
...             print(v)
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'os' is not defined
>>> import os
>>> for v in os.listdir("D:\\"):
...     if(re.search(".html$|.py$",v)):
...             print(v)
...
B.py
test.html
test1.py
>>>
>>> for v in os.popen('powershell get-service').readlines():
...     if(re.search("^Running",v)):
...             print(v.strip())
...
Running  AdobeARMservice    Adobe Acrobat Update Service
Running  AeLookupSvc        Application Experience
Running  AESTFilters        Andrea ST Filters Service
Running  AGMService         Adobe Genuine Monitor Service
Running  AGSService         Adobe Genuine Software Integrity Se...
Running  AnyDesk            AnyDesk Service
Running  Appinfo            Application Information
Running  AudioEndpointBu... Windows Audio Endpoint Builder
Running  AudioSrv           Windows Audio
Running  BFE                Base Filtering Engine
Running  BITS               Background Intelligent Transfer Ser...
Running  Browser            Computer Browser
Running  bthserv            Bluetooth Support Service
Running  btwdins            Bluetooth Service
Running  CertPropSvc        Certificate Propagation
Running  Credential Vaul... Credential Vault Host Control Service
Running  Credential Vaul... Credential Vault Host Storage
Running  CryptSvc           Cryptographic Services
Running  CscService         Offline Files
Running  DcomLaunch         DCOM Server Process Launcher
Running  Dhcp               DHCP Client
Running  Dnscache           DNS Client
Running  DPS                Diagnostic Policy Service
Running  EapHost            Extensible Authentication Protocol
Running  EMP_UDSA           EMP_UDSA
```

```
Running  eventlog           Windows Event Log
Running  EventSystem        COM+ Event System
Running  EvtEng             Intel(R) PROSet/Wireless Event Log
Running  FontCache          Windows Font Cache Service
Running  gpsvc              Group Policy Client
Running  hidserv            Human Interface Device Access
Running  HP LaserJet Ser... HP LaserJet Service
Running  HPSIService        HP SI Service
Running  ICCS               Intel(R) Integrated Clock Controlle...
Running  iphlpsvc           IP Helper
Running  KeyIso             CNG Key Isolation
Running  LanmanServer       Server
Running  LanmanWorkstation  Workstation
Running  lmhosts            TCP/IP NetBIOS Helper
Running  LMS                Intel(R) Management and Security Ap...
Running  mccspsvc           McAfee CSP Service
Running  MMCSS              Multimedia Class Scheduler
Running  MpsSvc             Windows Firewall
Running  Netman             Network Connections
Running  netprofm           Network List Service
Running  NlaSvc             Network Location Awareness
Running  nsi                Network Store Interface Service
Running  O2FLASH            O2FLASH
Running  O2SDIOAssist       O2SDIOAssist
Running  osppsvc            Office Software Protection Platform
Running  PcaSvc             Program Compatibility Assistant Ser...
Running  PlugPlay           Plug and Play
Running  Power              Power
Running  ProfSvc            User Profile Service
Running  RegSrvc            Intel(R) PROSet/Wireless Registry S...
Running  RelevantKnowledge  RelevantKnowledge
Running  RpcEptMapper       RPC Endpoint Mapper
Running  rpcnet             Remote Procedure Call (RPC) Net
Running  RpcSs              Remote Procedure Call (RPC)
Running  SamSs              Security Accounts Manager
Running  SCardSvr           Smart Card
Running  Schedule           Task Scheduler
Running  SENS               System Event Notification Service
Running  ShellHWDetection   Shell Hardware Detection
Running  Spooler            Print Spooler
Running  SSDPSRV            SSDP Discovery
Running  STacSV             Audio Service
Running  stisvc             Windows Image Acquisition (WIA)
Running  SysMain            Superfetch
Running  TabletInputService Tablet PC Input Service
Running  Themes             Themes
Running  TrkWks             Distributed Link Tracking Client
Running  UNS                Intel(R) Management and Security Ap...
Running  upnphost           UPnP Device Host
Running  UxSms              Desktop Window Manager Session Manager
Running  VMAuthdService     VMware Authorization Service
Running  VMnetDHCP          VMware DHCP Service
Running  VMUSBArbService    VMware USB Arbitration Service
Running  VMware NAT Service VMware NAT Service
Running  WdiServiceHost     Diagnostic Service Host
Running  WinDefend          Windows Defender
Running  WinHttpAutoProx... WinHTTP Web Proxy Auto-Discovery Se...
```

```
Running  Winmgmt            Windows Management Instrumentation
Running  Wlansvc            WLAN AutoConfig
Running  WMPNetworkSvc      Windows Media Player Network Sharin...
Running  wscsvc             Security Center
Running  WSearch            Windows Search
Running  wuauserv           Windows Update
Running  wudfsvc            Windows Driver Foundation - User-mo...
Running  ZeroConfigService  Intel(R) PROSet/Wireless Zero Confi...
>>>
>>> for v in os.popen('powershell get-service').readlines():
...     if(re.search("^Running.*network",v)):
...             print(v.strip())
...
>>>
>>> for v in os.popen('powershell get-service').readlines():
...     if(re.search("^Running.*network",v,re.I)):
...             print(v.strip())
...
Running  Netman             Network Connections
Running  netprofm           Network List Service
Running  NlaSvc             Network Location Awareness
Running  nsi                Network Store Interface Service
Running  WMPNetworkSvc      Windows Media Player Network Sharin...
>>>
>>>
>>> for v in os.popen('powershell get-service').readlines():
...     if(re.search("(^Running|^Stopped).*network",v,re.I)):
...             print(v.strip())
...
Stopped  napagent           Network Access Protection Agent
Running  Netman             Network Connections
Running  netprofm           Network List Service
Running  NlaSvc             Network Location Awareness
Running  nsi                Network Store Interface Service
Stopped  p2pimsvc           Peer Networking Identity Manager
Stopped  p2psvc             Peer Networking Grouping
Running  WMPNetworkSvc      Windows Media Player Network Sharin...

>>> for v in os.popen("ps -e").readlines():
...     if(re.search("[hs\d]$",v)):
...             print(v.strip())
...
3 ?        00:00:00 ksoftirqd/0
6 ?        00:00:00 migration/0
9 ?        00:00:00 netns
10 ?        00:00:00 sync_supers
21 ?        00:00:00 kswapd0
35 ?        00:00:01 kworker/u:2
36 ?        00:00:00 scsi_eh_0
37 ?        00:00:02 scsi_eh_1
38 ?        00:00:00 kworker/u:3
192 ?        00:00:00 mpt_poll_0
194 ?        00:00:00 mpt/0
262 ?        00:00:00 scsi_eh_2
279 ?        00:00:00 jbd2/sda1-8
```

In [ ]:
```
import pdb

pdb.set_trace()
----------------------------

python -m pdb p1.py
(pdb) n
(pdb) l
(pdb) h
```