Introduction to Python

Strings

>> **mkdir python**

>> **cd/python**

>> **leafpad first.py**

SCRIPT

#!/bin/python3

#print something

print("hello kanhaiya")

print("/n")   // this will print the new line

>> **leafpad first.py&** (& is used to open program with text editor)

>> **pluma first.py&** (used to run program in background with open text editor)

Run Methods

>> **python3 first.py** (1ST Run method)

>> **./first.py** (2nd Run Method)

MATH

```
#!/bin/bash

#Math

#!/bin/bash

#math function

print (10 + 10)  #addition
print (10 - 10)  #subtraction
print (10 * 10)  #multiply
print (10 / 10)  #division
print (10 + 10 - 10 * 10 / 10) #Pamdas
print (10 ** 10)  #Exponent
print (10 & 8)     #modulo
print (10 / 8)
print (10 // 10)   #no leftovers
```

Variables and Methods

Variables are like place holder Example: india = "this is india"

Methods are as functions that are available for given object

```
#!bin/bash/python3

#variable and methods

sutra = "good morning beta"
print(sutra.upper()) #uppercase
print(sutra.lower()) #lowercase
```

Example:

want to make one user and print username age and experience

```
name = "kanhaiya"
age = 24
experience = 17.2
print(int(age))
print(int(17.2)) or (experience)

print(' my name is ' + name + ' and i am ' + str(age) + ' years old ' + str(experience) + ' month of experience ')

age += 1    \\ increase age by one
print(age)
```

Functions ( )

the mini program that are organized block of code that define and call it later

SYNTEX:

Function use like below

def function_name():

 TAB   < - we use indentation means Press Tab

Example:

```
name = " this is kanhaiya "
age  = 24
designation = "software engineer"
experience = 18
print(int(age))
print(int(experience))

print(" hello " + name + " i am " + str(age) + " years old. and i am " + designation + " i have " + str(experience) + " month of experience ")
```

Example:

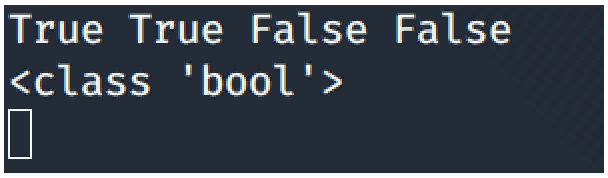
```
#adding the parameter

def adding_the_perameter(num):
	print(num + 100)
adding_the_perameter(200)
```

```
#multiple parameter
def add_perameter(x,y):
        print(x + y)

add_perameter (10,10)
-----------------------------------------------------------------
#mul mul_perameter
def mul(a,b):
        return(a * b)

print(mul(10,5))
--------------------------------------------------------------


#new Line
def new_line():
print("\n")
new_line()
```

Boolean Expressions

Boolean Expression is must be true and false

```
bool1 = True
bool2 = 3*3 == 9                    == \\ means something
bool3 = False
bool4 = 3*3 == 10

print(bool1,bool2,bool3,bool4)
print(type(bool1))
```



```
True True False False
<class 'bool'>
```

Relational and Boolean Expressions

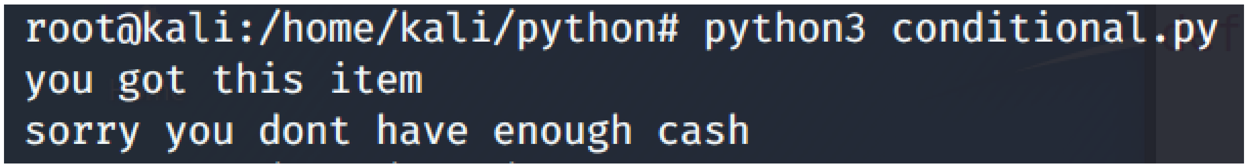
Relational operators and Boolean operators

```
greater then = 7 > 5
less then = 5 > 7
greater_then_equal_to = 7 >= 7
less_then_equal_to = 7 <= 7

test_and = (7 > 5) and (5 < 7) #True
test_and2 = (7 > 5) and (5 > 7) #False
test_or = (7 > 5) or (5 > 7) #True
test_or2 = (7 > 5) or (5 < 7) #True

test_not = not True #False
```

Conditional statement

Example:

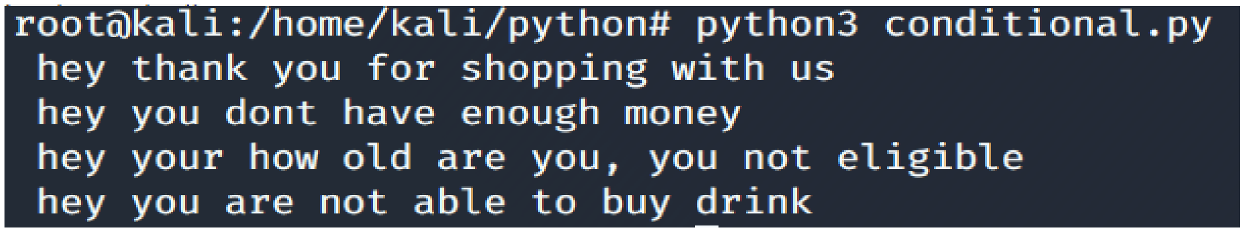
```
def drink(money):
    if money >= 5:
        return "you got this item"
    else:
        return "sorry you don’t have enough cash"


print(drink(8))
print(drink(2))
```



```
root@kali:/home/kali/python# python3 conditional.py
you got this item
sorry you dont have enough cash
```

Example:

```
def alcohol(age, money):
    if (age >= 18) and (money >= 5):
            return " hey thank you for shopping with us"
    elif (age >= 18) and (money < 5):
            return " hey you don’t have enough money "
    elif (age < 18) and (money >= 5):
            return " hey your how old are you, you not eligible"
    else:
            return " hey you are not able to buy drink "


print(alcohol(20,5))
print(alcohol(20, 4))
print(alcohol(1,5))
print(alcohol(1,2))
```

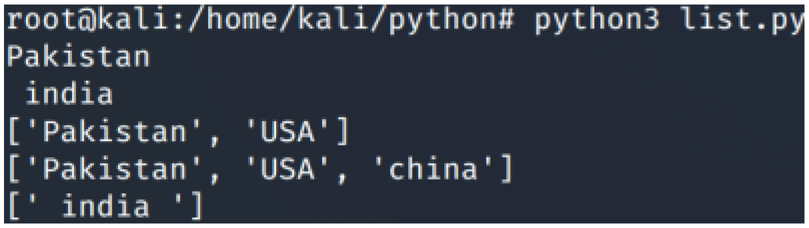

```
root@kali:/home/kali/python# python3 conditional.py
 hey thank you for shopping with us
 hey you dont have enough money
 hey your how old are you, you not eligible
 hey you are not able to buy drink
```

List [ ]

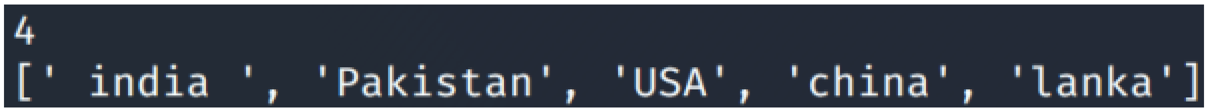
List are the data structure, are the changeable and we can put them list

List use [ ] this box brackets

```
#list
country = [" India " , "Pakistan" , "USA", "china" ]
print(country[1])  \\ this will print second item of list
print(country[0])  \\ this will print first item of list
print(country[1:3]) \\ this will print one number of item and third
print(country[1:])  \\ this will leave first item of list print other items.
print(country[:1])  \\ this will print the very first item of list
```

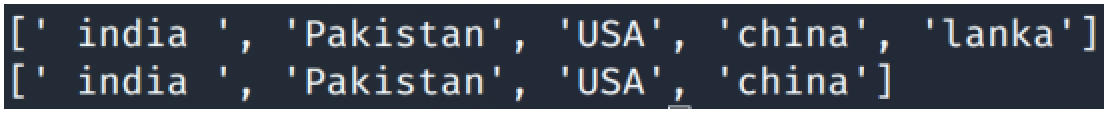

```
root@kali:/home/kali/python# python3 list.py
Pakistan
 india
['Pakistan', 'USA']
['Pakistan', 'USA', 'china']
[' india ']
```

```
print(len(country))
country.append("lanka")
print(country)
```


```
4
[' india ', 'Pakistan', 'USA', 'china', 'lanka']
```

country.pop()

print(country)



```
[' india ', 'Pakistan', 'USA', 'china', 'lanka']
[' india ', 'Pakistan', 'USA', 'china']
```

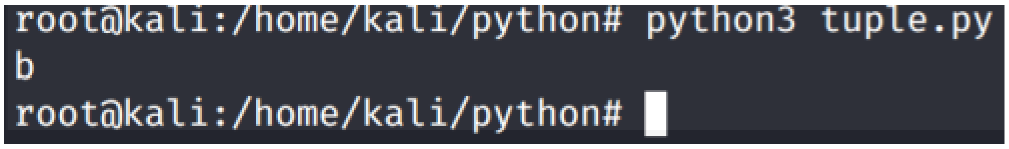
Tuples

tuples that can not be changed

(Immutable)

Grades = ('a', 'b', 'c','d', 'e', 'f')

print(Grades[1])



```
root@kali:/home/kali/python# python3 tuple.py
b
root@kali:/home/kali/python# 
```

Looping

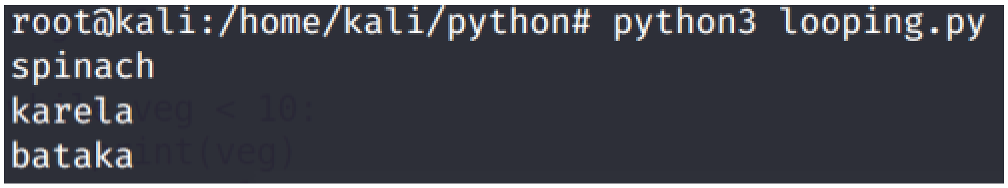
While loop is executing as long as something is TRUE

#tuples

vagetables = ['spinach', 'karela', 'bataka']

for vagges in vagetables:

print (vagges)



```
root@kali:/home/kali/python# python3 looping.py
spinach
karela
bataka
```

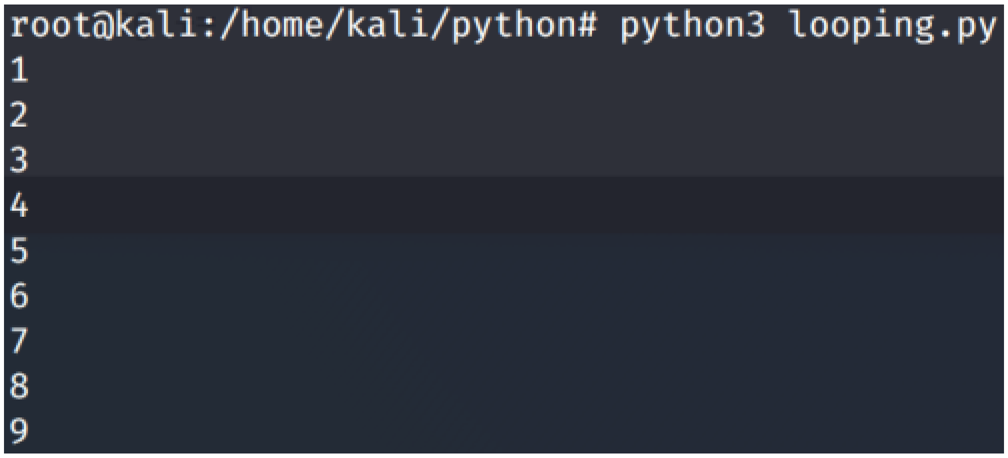
#while loop = is executing as long as TRUE

veg = 1

while veg < 10:

print(veg)

veg += 1



```
root@kali:/home/kali/python# python3 looping.py
1
2
3
4
5
6
7
8
9
```

Importing Modules

Most common import is

import sys

import os

import datetime from datetime

sys.exit()

>> **sys** (is to related with system function and parameters.)

>> Example:

import sys

import datetime from datetime

print("sys.version")

print(datetime.now())

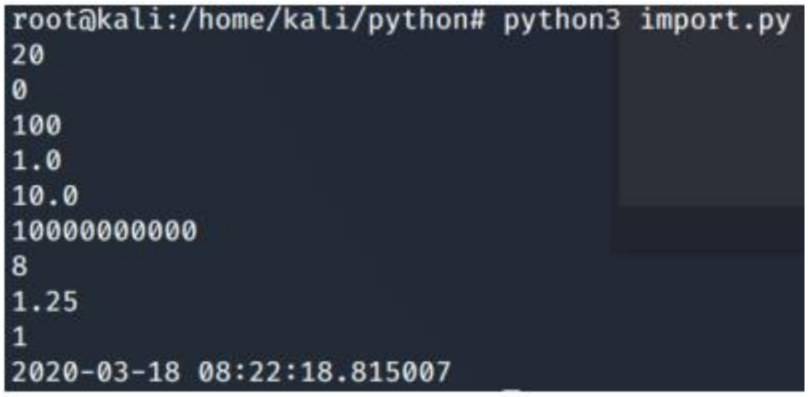
>> Example:

import sys

import datetime from datetime as dt

print("sys.version")

print(dt.now())



```
root@kali:/home/kali/python# python3 import.py
20
0
100
1.0
10.0
10000000000
8
1.25
1
2020-03-18 08:22:18.815007
```

Dictionaries { }

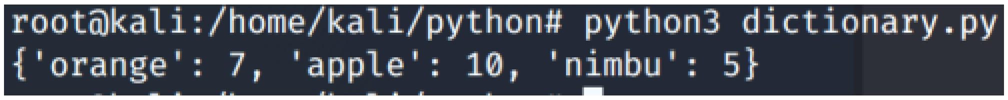
Dictionaries is using key/value pairs

Dictionaries always use { } this brackets if you define something

Examples: With single value and pairs (drink with cost)

drink = {"orange": 7, "apple": 10, "nimbu": 5,}

print(drink)



```
root@kali:/home/kali/python# python3 dictionary.py
{'orange': 7, 'apple': 10, 'nimbu': 5}
```

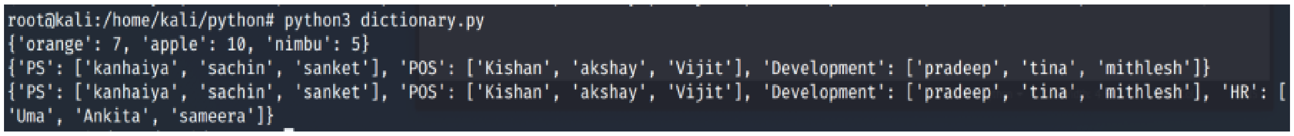
Example: with multiple Value and Pairs

Emp = {"PS": ["kanhaiya", "sachin", "sanket"], "POS":["Kishan", "akshay", "Vijit"], "Development": ["pradeep", "tina", "mithlesh"] }

print(Emp)

Emp["HR"] = ["Uma", "Ankita", "sameera",] \\\adding the items in dictionaries

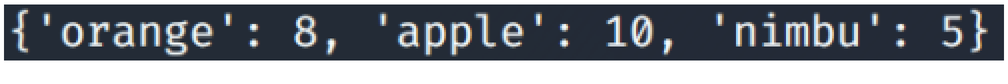
print(Emp)



```
root@kali:/home/kali/python# python3 dictionary.py
{'orange': 7, 'apple': 10, 'nimbu': 5}
{'PS': ['kanhaiya', 'sachin', 'sanket'], 'POS': ['Kishan', 'akshay', 'Vijit'], 'Development': ['pradeep', 'tina', 'mithlesh']}
{'PS': ['kanhaiya', 'sachin', 'sanket'], 'POS': ['Kishan', 'akshay', 'Vijit'], 'Development': ['pradeep', 'tina', 'mithlesh'], 'HR': [
'Uma', 'Ankita', 'sameera']}
```

drink["orange"] = 8 \\ we can change the value with this way

print(drink)



```
{'orange': 8, 'apple': 10, 'nimbu': 5}
```

print(drink.get("oranges"))

Sockets

Sockets are used to connect two nodes together.

socket are used to connect open port and IP address establish connection and sent malicious data (used in port scanner and exploit development)

Simple script Example:

import socket

HOST = "127.0.0.1"

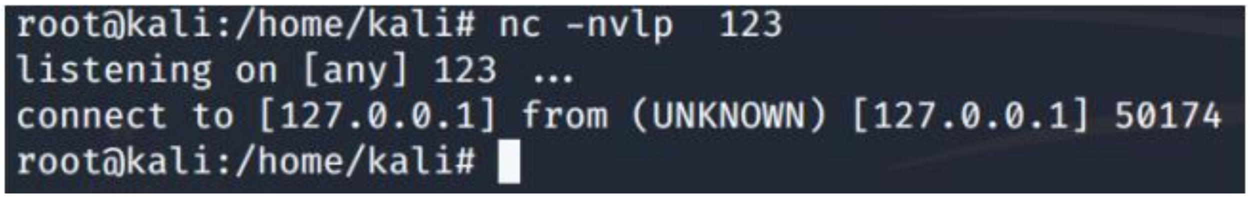
PORT = 123

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)   // socket af_inet is and IPv4 address and SOCK_STREAM is port number

s.connect((HOST, PORT))   \\ this will connect to open port ip and port

NOW   OPEN NEW TERMINAL

>> nc -nvlp 123 \\ netcat and -nvlp means open a listening port



```
root@kali:/home/kali# nc -nvlp  123
listening on [any] 123 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 50174
root@kali:/home/kali# 
```

Port Scanner

BUILDING A PORT SCANNER

Main it is based on

Run the file Scanner.py <IPAddress>   \\ we are going to run the ip address selected range are try to return results and whether or not port is open

SCRIPT

import sys | we are going to use some built in system functions

import socket | we are use to connect one node to another

from datetime import datetime | use the date time function to connect date time

#define the target

if len (sys.argv) == 2:   # this is same like $1 like bash

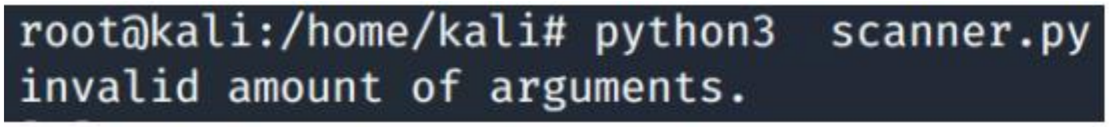
target = socket .gethostbyname(sys.argv[1])   # translating the host name by

IPV4 | this is also same like $1 in bash

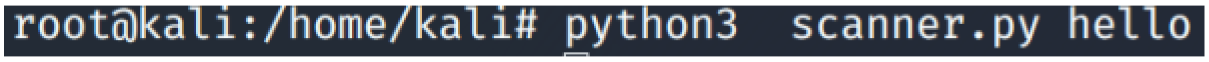
else:

print("invalid amount of arguments.")

    print("Syntex: python3 scanner.py <ipaddress>.")



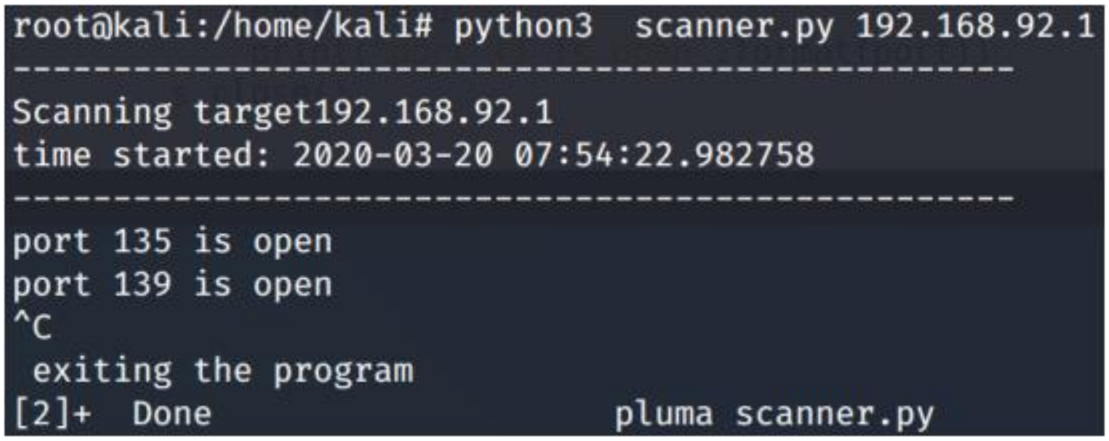
```
root@kali:/home/kali# python3  scanner.py
invalid amount of arguments.
```



```
root@kali:/home/kali# python3  scanner.py hello
```

```
import sys
import socket
from datetime import datetime

#define the target

if len (sys.argv) == 2:                              # this is same like $1 like bash
        target = socket.gethostbyname(sys.argv[1])   # translate host name too ip address
else:
                print("invalid amount of arguments.")

#adding the pretty banner

print("-" * 50)
print("Scanning target" +target)
print("time started: " +str(datetime.now()))
print("-" * 50)

try:
        for port in range (50, 80):                                   #(50, 1343546)
                s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                socket.setdefaulttimeout(1)
                result = s.connect_ex((target, port))
#print ("checking port".format(target, port))
                if result == 0:
                        print("port {} is open".format(port))      # Returns an error indicator
                s.close()

except KeyboardInterrupt:                                  #this is the keyboard interrupt
                print("\n exiting the program")
                sys.exit()
```

```
except socket.gaierror:
        print("host name could not be resolved")
        sys.exit()

except socket.error:
        print("couldn't connect to server")
        sys.exit()
```



```
root@kali:/home/kali# python3  scanner.py 192.168.92.1
--------------------------------------------------
Scanning target192.168.92.1
time started: 2020-03-20 07:54:22.982758
--------------------------------------------------
port 135 is open
port 139 is open
^C
 exiting the program
[2]+  Done                    pluma scanner.py
```