DataSets Files Type

1 - Functions

- Functions accept arbitrary objects as parameters and return values
- Types of parameters and return values are unspecified
- Functions without explicit return value return None

In [7]:
```
# to define any function in python using (keword def then name of function then function parameters)
def hello_world ():
    print (" Hello World !")

a = hello_world()
print (a)
```

```
 Hello World !
None
```

In [5]:
```
# to define any function in python using (keword def then name of function then fnction parameters)
def add (a , b ):
    """ Returns the sum of a and b."""
    mysum = a + b
    return mysum

# call the function
print(add(6,7))
help(add)
```

```
13
Help on function add in module __main__:

add(a, b)
    Returns the sum of a and b.
```

In [10]:
```
# Multiple return values are realised using tuples or lists:
def sumMul(a , b):
    return (a+b , a*b )

ret = sumMul(5,6)
(sum1 , mul1 ) = sumMul(5,6)
print("the two value is: ",ret," the summation",sum1,"the multiplication ",mul1,sep="\t")
```

```
the two value is: 	(11, 30)	 the summation	11	the multiplication 	30
```

Optional Parameters – Default Values

- Parameters can be defined with default values.
- Hint: It is not allowed to define non-default parameters after default parameters

In [15]:
```
def fline (x , m =1 , b =0): # f(x) = m*x + b
    return m * x + b

for i in range (5):
    print ( fline ( i ) , end =" ")

# force newline
print ()

for i in range (5):
    print ( fline (i , -1 ,1) , end =" ")
```

```
0 1 2 3 4
1 0 -1 -2 -3
```

Positional Parameters

- Parameters can be passed to a function in a different order than specified:

In [14]:
```
def printContact ( name , age , location ):
    print (" Person : ", name )
    print ("Age : ", age , " years ")
    print (" Address : ", location )
printContact ( name =" Peter Pan", location =" Neverland ", age =10)
```

```
 Person :   Peter Pan
Age :  10  years
 Address :   Neverland
```

In [ ]:

2 - OOP

- So far: procedural programming
  - Data (values, variables, parameters, . . .)
  - Functions taking data as parameters and returning results
- Alternative: Group data and functions belonging together to form custom data types

In [17]:
```
# Simple Classes as Structs
class Point :
    pass

p = Point ()
p.x = 2.0
p.y = 3.3
print(p.x,p.y)

#Class: Custom date type (here: Point )
#Object: Instance of a class (here: p )
#Attributes (here x , y ) can be added dynamically
#Hint: pass is a No Operation (NOOP) function
```

```
2.0 3.3
```

In [18]:
```
# Classes - Constructor
class Point :
    def __init__ (self , x , y ):
        self.x = x
        self.y = y

p = Point (2.0 , 3.0)
print ( p .x , p . y )
#__init__ : Is called automatically after creating an object
```

```
2.0 3.0
```

In [23]:
```
# Methods on Objects
import math

class Point :
    def __init__ (self , x , y ):
        self . x = x
        self . y = y
    def norm ( self ):
        n = math . sqrt ( self . x **2 + self . y **2)
        return n

p = Point (2.0 , 3.0)
print ( p .x , p .y , p . norm ())

# Method call: automatically sets the object as first parameter
# traditionally called self
```

```
2.0 3.0 3.605551275463989
```

In [24]:
```
#Class Variables Have the same value for all instances of a class:
class Point :
    count = 0 # Count all point objects
    def __init__ (self , x , y ):
        Point.count += 1 # self . __class__ . count += 1

p1 = Point (2 , 3); p2 = Point (3 , 4)
print(Point.count)
```

```
2
```

Inheritance

- There are often classes that are very similar to each other.
- Inheritance allows for:
  - Hierarchical class structure (is-a-relationship)
  - Reusing of similar code -Example: Different types of phones Phone Mobile phone (is a phone with additional functionality) Smart phone (is a mobile phone with additional functionality)

In [25]:
```
class Phone :
    def call ( self ):
        print('phone normally make calls')
# single inheritance
class MobilePhone ( Phone ):
    def send_text ( self ):
        print('mobile phone can send text')

h = MobilePhone ()
h . call () # inherited from Phone
h . send_text () # own method
```

```
phone normally make calls
mobile phone can send text
```

In [26]:
```
# multi-level inheritance
class Phone :
    def call ( self ):
        print('phone normally make calls')
# single inheritance
class MobilePhone ( Phone ):
    def send_text ( self ):
        print('mobile phone can send text')

class Samsung ( MobilePhone ):
    def high_security ( self ):
        print('Samsung has high security')

class A30s ( Samsung ):
    def bad_wifi ( self ):
        print('A30s has bad wifi connection')
h = A30s ()
h . call ()
h . send_text ()
h.high_security()
h.bad_wifi()
```

```
phone normally make calls
mobile phone can send text
Samsung has high security
A30s has bad wifi connection
```

In [27]:
```
# multible inheritance
class Phone :
    def call ( self ):
        print('phone normally make calls')

class MobilePhone ( ):
    def send_text ( self ):
        print('mobile phone can send text')

class Samsung ( ):
    def high_security ( self ):
        print('Samsung has high security')

class A30s (Phone,MobilePhone, Samsung ):
    def bad_wifi ( self ):
        print('A30s has bad wifi connection')
h = A30s ()
h . call ()
h . send_text ()
h.high_security()
h.bad_wifi()
```

```
phone normally make calls
mobile phone can send text
Samsung has high security
A30s has bad wifi connection
```

In [29]:
```
# hierichal
class Phone :
    def call ( self ):
        print('phone normally make calls')

class Samsung ( Phone ):
    def high_security ( self ):
        print('Samsung has high security')

class Iphone (Phone):
    def good_wifi ( self ):
        print('Iphone has good wifi connection')
i = Iphone ()
s = Samsung()
h . call ()
s . call()
```

```
phone normally make calls
phone normally make calls
```

3- TEXT FILES

method 1 to deal with text file

In [253…
```
%%writefile text1.txt
hello my name is hossam
my age is 21 years old
```

```
Overwriting text1.txt
```

In [254…
```
%%writefile -a text1.txt
This is more text being appended to text1.txt
And another line here.
```

```
Appending to text1.txt
```

In [255…
```
my_file = open('text1.txt')
```

In [256…
```
my_file.seek(0)
#will return list of lines
print(my_file.readlines())
#will read all text file
print(my_file.read())
#return cursor to first
my_file.seek(0)
#will read only one line
print(my_file.readline(),end='')
print(my_file.readline())
#we should close the file after finish
my_file.close()
```

```
['hello my name is hossam\n', 'my age is 21 years old\n', 'This is more text being appended to text1.txt\n', 'And another line here.\n']

hello my name is hossam
my age is 21 years old
```

method 2 to deal into text file

In [257…
```
#first mode of open file is w mean open file to write only w+ mean write or anything else
my_file = open('text1.txt','w')
#will erase all text in file then write new text
my_file.write('This is a new first line')
my_file.close()
```

In [258…
```
#seccond mode of open file is a mean open file to append only a+ mean append| or anything else
my_file = open('text1.txt','a')
#will append to the file
my_file.write('\nThis is a new second line')
my_file.close()
```

In [260…
```
#seccond mode of open file is r mean open file to read only r+ mean read or anything else
my_file = open('text1.txt','r')
#will append to the file
print(my_file.read())
my_file.close()
```

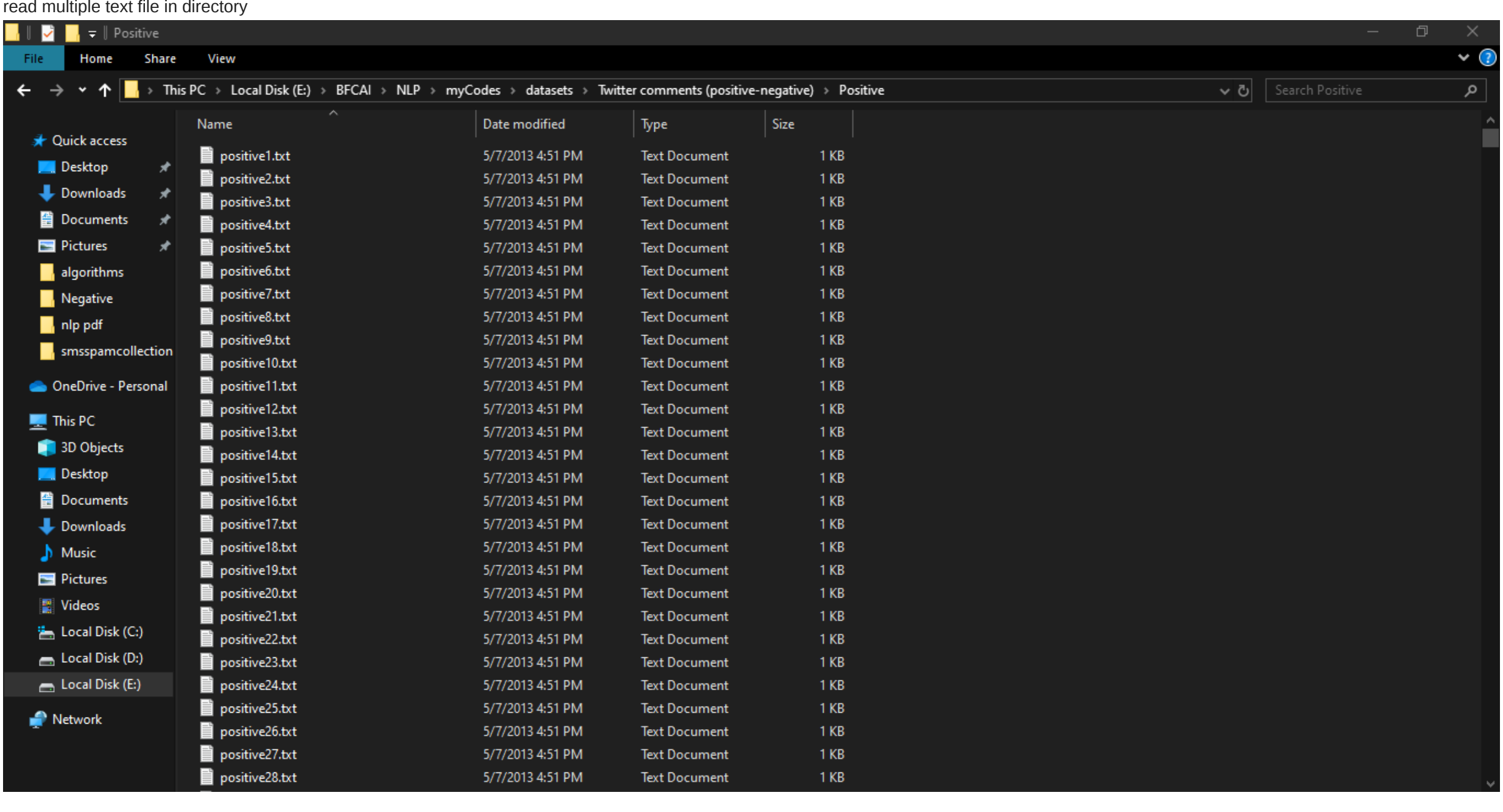
```
This is a new first line
This is a new second line
```

In [261…
```
#with is used to automatically close file after finish
with open('text1.txt','r') as txt:
    first_line = txt.readlines()[0]
    print(first_line)
'''
txt.seek(0)
print(txt.read())
'''
with open('text1.txt','r') as txt:
    for line in txt.readlines(): #will read line by line
        print(line, end='')
```

```
This is a new first line

This is a new first line
This is a new second line
```

read multiple text file in directory



Positive
File
Home
Share
View
This PC › Local Disk (E:) › BFCAI › NLP › myCodes › datasets › Twitter comments (positive-negative) › Positive
Search Positive
Quick access
Desktop
Downloads
Documents
Pictures
algorithms
Negative
nlp pdf
smsspamcollection
OneDrive - Personal
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Local Disk (D:)
Local Disk (E:)
Network
Name
Date modified
Type
Size
positive1.txt 5/7/2013 4:51 PM Text Document 1 KB
positive2.txt 5/7/2013 4:51 PM Text Document 1 KB
positive3.txt 5/7/2013 4:51 PM Text Document 1 KB
positive4.txt 5/7/2013 4:51 PM Text Document 1 KB
positive5.txt 5/7/2013 4:51 PM Text Document 1 KB
positive6.txt 5/7/2013 4:51 PM Text Document 1 KB
positive7.txt 5/7/2013 4:51 PM Text Document 1 KB
positive8.txt 5/7/2013 4:51 PM Text Document 1 KB
positive9.txt 5/7/2013 4:51 PM Text Document 1 KB
positive10.txt 5/7/2013 4:51 PM Text Document 1 KB
positive11.txt 5/7/2013 4:51 PM Text Document 1 KB
positive12.txt 5/7/2013 4:51 PM Text Document 1 KB
positive13.txt 5/7/2013 4:51 PM Text Document 1 KB
positive14.txt 5/7/2013 4:51 PM Text Document 1 KB
positive15.txt 5/7/2013 4:51 PM Text Document 1 KB
positive16.txt 5/7/2013 4:51 PM Text Document 1 KB
positive17.txt 5/7/2013 4:51 PM Text Document 1 KB
positive18.txt 5/7/2013 4:51 PM Text Document 1 KB
positive19.txt 5/7/2013 4:51 PM Text Document 1 KB
positive20.txt 5/7/2013 4:51 PM Text Document 1 KB
positive21.txt 5/7/2013 4:51 PM Text Document 1 KB
positive22.txt 5/7/2013 4:51 PM Text Document 1 KB
positive23.txt 5/7/2013 4:51 PM Text Document 1 KB
positive24.txt 5/7/2013 4:51 PM Text Document 1 KB
positive25.txt 5/7/2013 4:51 PM Text Document 1 KB
positive26.txt 5/7/2013 4:51 PM Text Document 1 KB
positive27.txt 5/7/2013 4:51 PM Text Document 1 KB
positive28.txt 5/7/2013 4:51 PM Text Document 1 KB

In [65]:
```
# READ MULTIPLE TEXT FILES
import os
path = "E:\\BFCAI\\NLP\\myCodes\\datasets\\Twitter comments (positive-negative)\\Positive"
index = 1
data_pos = []
for file in os.listdir(path):
    if file.endswith('.txt'):
        file_path = os.path.join(path,file)
        with open(file_path,encoding='utf-8-sig') as f:
            data_pos.append(f.readline().replace('\n',''))
```

In [67]:
```
data_pos[0:10]
```

Out[67]:
```
['حقا',
 'صح جدا\xa0',
 'اه و النعمة ',
 'كلامك جميل ورائع',
 'و خير الكلام ما قل و دل',
 'ولا تنسو تبسمك في وجه اخيك صدقه',
 'ليتنا نستطيع الابتسامه في الاوقات المؤلمه',
 'و بحب الناس الرايقه الي بتضحك على طول',
 'ابتسم امام كل الناس لتسرهم و تجبر خواطرهم فتبسمك في وجوههم صدقه',
 'و الله انت حبيبي ']
```

CSV FILES (COMMA SEPRATED VALUE)

Read CSV File

In [262…
```
import pandas as pd
#read csv file
data = pd.read_csv('prototype.csv')#  skiprows = 5)
data.head()
```

Out[262…

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	blackheads	scurring	skin_peeling	silver_like_dusting	small
0	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	
1	0	1	1	0	0	0	0	0	0	0	...	0	0	0	0	
2	1	0	1	0	0	0	0	0	0	0	...	0	0	0	0	
3	1	1	0	0	0	0	0	0	0	0	...	0	0	0	0	
4	1	1	1	0	0	0	0	0	0	0	...	0	0	0	0	

5 rows × 133 columns

Create CSV File

In [264…
```
outfile = open('hossam.csv', 'w')
outfile.write('name,age,gender,salary\n')
outfile.write('hossam,23,male,5000\n')
outfile.write('shimaa,23,female,nan\n')
outfile.write('shimaa,nan,female,6000\n')
outfile.close()
```

In [265…
```
data = pd.read_csv('hossam.csv')#  skiprows = 5)
data.head()
```

Out[265…

	name	age	gender	salary
0	hossam	23.0	male	5000.0
1	shimaa	23.0	female	NaN
2	shimaa	NaN	female	6000.0

In [294…
```
#to deal with arabic we need to specify encoding
# some other files need another encoding because it contain special character that need special encoding to read
# if this problem occured search on google to find the specific encoding
f= open('ar.txt','w', encoding="utf8")
f.write('حسام فارس')
f.close()
```

In [282…
```
f= open('ar.txt','r', encoding="utf8")
print(f.read())
```

```
حسام فارس
```