In [35]:

```
#for see run time: %%time   #include it on top of cell
#built in function
#type(),str(),format(),int(),len()

#method:

#var.lower()
#var.upper()
#var.capitalize()
#var.replace("which","by-which")
#var.split(',')
#var.strip()
#print(var1,var2)
#print(var1+' '+var2)  //if int use str()
```

In [50]:

```
#list=[1,2,3,4]
#list(range(1,9))  #create list 1to 8
#list.method()
#list.lower()
#list.append("value")
#list.insert(1,"value")
#list2=list.copy()
#list.pop(3)   //delete index 3
#list1+list2
#list[1:3]  1 to 2
#list.remove("val")
```

In [2]:

```
dictionary={
    'name':'Asad',
    'id':1911022,
    'age':22
}
```

In [1]:

```
#dict.keys()
#dict.values()
#dict.items()
#dict['new_key']="val"   appending
#dict['old_keys']="New_val"   modifying
```

In [2]:

```
thistuple=(1,2,3,4,5,6,7,8,9)
#tuple(range(1,9))
```

In [23]:

```
thistuple.count(5) #count() method returns the number of times a specified val
ue appears in the tuple.
```

Out[23]:

```
1
```

In [5]:

```
#if in var:
var=3
if(var==3):
    print("found")
else:
    print("Not Found")
```

```
found
```

In [7]:

```
#if in list
list=['asad','shanto','rahat']
friend='shanto'
if friend in list:
     print("Found")
```

```
Found
```

In [8]:

```
#factorial by while:

num=5
fact=1
while(num>0):
    fact=fact*num
    num=num-1
print(fact)
```

```
120
```

In [9]:

```
#triangle by while loop:

length=10
line="*"
while(len(line)<=length):
    print(line)
    line+="*"

while(len(line)>0):
    print(line)
    line=line[:-1]
```

```
*
**
***
****
*****
******
*******
********
*********
**********
***********
**********
*********
********
*******
******
*****
****
***
**
*
```

In [10]:

```
#for in var:

name="asadul islam hamzah"
for x in name:
    print(x)
```

```
a
s
a
d
u
l
 
i
s
l
a
m
 
h
a
m
z
a
h
```

In [37]:

```
#for in list
for x in list:
    print(x)

for i in range(len(list)):
    print("key",i,"value=",list[i])   #you can use + ,but also str()

list2=['samira','erdogan','toha']

for i in zip(list,list2):
    print(i)

for x,y in zip(list,list2):
    print("list1=",x,"list2=",y)
```

```
asad
shanto
rahat
key 0 value= asad
key 1 value= shanto
key 2 value= rahat
('asad', 'samira')
('shanto', 'erdogan')
('rahat', 'toha')
list1= asad list2= samira
list1= shanto list2= erdogan
list1= rahat list2= toha
```

In [18]:

```
#for in dict

for key in dictionary:
    print(key)

for key in dictionary:
    print(dictionary[key])

for key in dictionary:
    print("Key=",key," and value=",dictionary[key])

for key in dictionary.items():
    print(key)
```

```
name
id
age
Asad
1911022
22
Key= name  and value= Asad
Key= id  and value= 1911022
Key= age  and value= 22
('name', 'Asad')
('id', 1911022)
('age', 22)
```

In [7]:

```
#for in range

for i in range(10):   #0 to 9
    print(i)

print('br')

for i in range(10,16): #10 to 15
    print(i)

print('br')

for i in range(10,25,3):  #10 to 24 , i=i+3
    print(i)

for i in range(10):
    for j in range(5):
        print(i," ",j)
```

```
0
1
2
3
4
5
6
7
8
9
br
10
11
12
13
14
15
br
10
13
16
19
22
0   0
0   1
0   2
0   3
0   4
1   0
1   1
1   2
1   3
1   4
2   0
2   1
2   2
2   3
2   4
3   0
3   1
3   2
3   3
3   4
4   0
4   1
4   2
4   3
4   4
5   0
5   1
5   2
5   3
5   4
6   0
6   1
6   2
6   3
```

```
6   4
7   0
7   1
7   2
7   3
7   4
8   0
8   1
8   2
8   3
8   4
9   0
9   1
9   2
9   3
9   4
```

In [9]:

```
#Function
#function structure:
def func_name():
       print("Function created")
#call the function
func_name()

#function arguement:
def print_func(name):
    print("Hello",name) #space taken automatically
print_func('Asad')

#Named arguement:
def print_func(name):
    print("Hello",name) #space taken automatically
print_func(name='Asad')

def print_func(name='Asad'):
    print("Hellow",name,".Asad is default value")

print_func()
```

```
Function created
Hello Asad
Hello Asad
Hellow Asad .Asad is default value
```

In [11]:

```
#Module
#import module_name
import math,numpy
#module_name.function_name()
math.ceil(1.3)
```

Out[11]:

```
2
```

In [12]:

```
#try except:
#If there is any error in try,execution will stop in try,and execute except pr
ogram

try:
     print('Computing..')
     result=5/0  #here is an error

except:
     print('Falid')
     result='none'

print(result)
```

```
Computing..
Falid
none
```

In [20]:

```
#numpu module:
import numpy

#numpy.function():

#np_array=numpy.array(an_array)
#np_array=np.arange(3,100,5)  # 3 to 100 with 5 increament
#np_array=np.random.rand(4)   # 4 means dimension:4,0

#dot_product=numpy.dot(list1,list2)   #or  list1 @ list2
#matix_multiplication=numpy.matmul(list1,list2)

#numpy array:

#np_array.shape
#np_array.reshape(4,3)
#np_array.dtype

#array manipulation:
#numpy.array_split(np_array, 2)   #divide into 2 array
#numpy.sort(np_array)   #asc

#numpy query:
#numpy.where(np_array == 4)    #return index no.

#numpy all func:

#Mathematics: np.sum(np_list) , np.exp(np_list) , np.round(np_list) //float value to round value
#Array manipulation: np.reshape() , np.stack() , np.concatenate() , np.split()
#Linear Algebra: np.matmul() , np.dot() , np.transpose() , np.eigvals
#Statistics: np.mean(np_list), np.median(np_list), np.mode(np_list) , np.std(np_list) , np.var(np_list) , np.max(np_list)

#Mean - The average value
#Median - The mid point value
#Mode - The most common value
#Std - A standard deviation means that most of the numbers are close to the mean (average) value.
#Var - Variance is another number that indicates how spread out the values are.
```

In [24]:

```
#txt file:                                  remember: urlretrieve() , genfrom
txt()
import urllib.request as url
#download part:
#url.urlretrieve('url','file-saving name.txt')  #this for any type file
#in_var=np.genfromtxt('climate.txt',delimiter=',',skip_header=1)  #delimiter i
s sperator of data.which is coma  .shape=10000,
#adding part:
#in_var_new_col=np.concatenate((in_var,new_col_list.reshape(10000,1)),axis=1)

#saving latest data as txt:
#np.savetxt(
#       'climate_result.txt',   #file name
#        in_var_new_col,        #the array
#       fmt='%.2f',
#       delimiter=',',
#       header='temperature,rainfall,humidity,yeild_apples',
#       comments=''
#           )
```

In [23]:

```
url_='https://gist.github.com/BirajCoder/a4ffcb76fd6fb221d76ac2ee2b8584e9/raw/
4054f90adfd361b7aa4255e99c2e874664094cea/climate.csv'

url.urlretrieve(url_,'climate.txt')
```

Out[23]:

```
('climate.txt', <http.client.HTTPMessage at 0x20366631b88>)
```

In [25]:

```
#pandas:
import pandas

#data=pandas.read_csv('name.csv')
#data.info()
#data.describe()  #show   mean, standard deviation, minimum/maximum values,

#data.columns
#data.shape

#col data:
#data['col_name']  #take col value  or, data.col_name
#data[['col1','col2']]
#data['col_name'][120]   #of index 120   or, data.at[120,'col_name']


#col1=data['col_name'].copy()

#row data:
#data.loc[108:113]  #show 108 to 113 index
#data.head(5)   #show 1st five
#data.tail(5)   #show last 5
#data.sample(5)  #take 5 row randomly


#query:

#q_data=data[data.col_name>100]   #where > 100

#q_data=data[(data.col1/data.col2)>10]


#delete column

#data.drop(columns=['col'],inplace=True)


#sort
#desc:
#data.sort_values('col',ascending=False) # asc-:-ascending=True
#asc:
#data.sort_values('col')


#col operation:
#data.at[172,'new_cases']=data['new_cases'][171]+data.at[173,'new_cases']
#data.col.mean()


#goup date:
#pandas.DatetimeIndex(data.date).year/month

#adding new col:
#data['newc_col']=new_data

#grouping time
```

```
#data['year']=pandas.DatetimeIndex(data.date).year

#aggregation data by time:
#data.groupby('col_month')['col'].sum()
#data_month = data.groupby('col_month')[['col1','col2','col3']].sum()   #col v
alue will be sum based on same month


#comulative sum.sum row by row
#data.col.cumsum()


#merge data
#merged_data=data.merge(data2,on='base_col')


#saving data
#merged_data.to_csv('results.csv', index=None)



#plotting:

#data.col.plot(title='Title',kind='bar')  #kind is plotting type
```

In [3]:

```
#Matplotlib and Seaborn:

import matplotlib.pyplot as plt
import seaborn as sns

#plot:
#plt.plot(list) #just line

#single plot:
#plt.plot(year,list)

#Double plot:          in a 1 visualization
"""
plt.plot(year,list1)
plt.plot(year,list2)          #its line
"""

#scatterplot:
"""
sns.scatterplot(x=col1_data,y=col2_data)     #its dot
sns.scatterplot(x=length_data,y=width_data,hue=data.col, s=100); # s is point
 size  #hue indicate name & color of that col

"""

#change setting:
"""
plt.figure(figsize=(12, 6))   #changing figure size
plt.xlabel('Years')
plt.ylabel('Yields')
plt.title('A title')
plt.legend(['Apples','Orange'])  #indicating the line
sns.set_style('whitegrid')  #darkgrid
"""

#standard plot:
"""
plt.figure(figsize=(12, 6))
plt.plot(years,apples,marker='o',c='red')
plt.plot(years,oranges,marker='x',ls='--')

plt.xlabel('Years')
plt.ylabel('Yields')

plt.title('A title')

plt.legend(['Apples','Orange'])  #indicating the line
"""

#standard scatterplot:
"""
sns.scatterplot(x='sepal_length',
                y='sepal_width',
                hue='species',
                s=100,
                data=flowers_df);  #datasheet here
```

```
"""

#plot,scatterplot together:
"""
plt.title('Sepal Dimensions')
sns.scatterplot(x=length,y=width,hue=flowers_df.species, s=100)
"""





#load data by seaborn:
#sns.load_dataset('iris')   #iris from internal serve


#you can manipulate,query column like as numpy

#except:
#sns_data.col.unique()   #return unique value,just for sns data



#histogram:    hist is vertical bar,
"""
plt.hist(col)
plt.hist(col,bins=5)  #divide into 5 bar
plt.hist(width,bins=np.arange(2,5,0.25))   #limit all bar by array

"""



#double hist:
"""
plt.hist(col1,bins=5)
plt.hist(col2,bins=5)
plt.hist([col1,col2],bins=5)  #at 1 line

"""



#standard Histogram:
"""
plt.title('Distribution of Sepal Width')

plt.hist([setosa_df.sepal_width, versicolor_df.sepal_width, virginica_df.sepal
_width],
         bins=np.arange(2, 5, 0.25),
         stacked=True);

plt.legend(['Setosa', 'Versicolor', 'Virginica']);

"""



#Bar:
"""
plt.bar(years,col); #sns.barplot() from sns.barplot function which can automat
```

```
ically compute averages.

"""

#double bar:
"""
plt.bar(years, col1)
plt.bar(years, col2, bottom=col1);

"""

#standard barplot:
"""
sns.barplot(x='day', y='total_bill', hue='gender', data=tips_df);   #show 2 ba
r comparetively.

sns.barplot(x='total_bill', y='day', hue='sex', data=tips_df);   # its horizon
tal

"""

#Heatmap:   indicates by color opacity
"""
sns.heatmap(flights_df)

"""



#standard Heatmap:
"""
plt.title("No. of Passengers (1000s)")
sns.heatmap(flights_df, fmt="d", annot=True, cmap='Blues');

"""
```

```
Out[3]: "\nsns.scatterplot(x='sepal_length', \n                y='sepal_width', \n
        hue='species',\n                s=100,\n                data=flowers_df);  #d
        atasheet here\n"
```

In [6]:

```
#Image Module:

from PIL import Image

#read:
#img=Image.open('name.jpg')


#print:
#plt.imgshow(img)


#standard print:
"""
plt.grid(False)
plt.title('A data science meme')
plt.axis('off')
plt.imshow(img);
"""



#Crop Image:
"""
np_img=numpy.array(img)
plt.imshow(np_img[125:325,105:305]);

"""
```

Out[6]:

```
'\nnp_img=numpy.array(img)\nplt.imshow(np_img[125:325,105:305]);\n\n'
```

In [ ]:

```
#Multiple charts
```