To install:

pip install numpy

pip install matplotlib

pip install pandas

```
#importing library
import numpy as np
```

```
#creating numpy array
#initialization
arr1=np.array([1,2,3,4,5])
print(arr1)
```

```
[1 2 3 4 5]
```

```
#2-dim array
arr2=np.array([[1,2,3],[4,5,6]])
print(arr2)
```

```
[[1 2 3]
 [4 5 6]]
```

```
#3-dim array
arr3=np.array([[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]])
print(arr3)
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

```
#creating array in range
arr4=np.arange(5,50)
print(arr4)
```

```
[ 5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49]
```

```
#random function
arr5=np.random.rand(2)
print(arr5)
```

```
[0.52704466 0.97231439]
```

```
#creating arrays of zeros
arr6=np.zeros((2,3))
print(arr6)
```

```
[[0. 0. 0.]
 [0. 0. 0.]]
```

```
#reshaping array
arr7=np.arange(12).reshape(3,4)
print(arr7)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
```

```
#printing the dim
print(arr7.ndim)
print(arr3.ndim)
```

```
2
3
```

```
#dtype: to display the types of element
print(arr7.dtype)
```

```
int64
```

```
arr8=np.array([1.2,2.3,3.4],dtype=np.int8)
print(arr8.dtype)
```

```
int8
```

```
#itemsize: size of elements
print(arr8.itemsize)
```

```
1
```

```
#size of array using size
arr9=np.array([[1,2,3],[4,5,6]])
print(arr9.size)
```

```
6
```

```
#sclicing :extracting particular set
arr10=np.array([1,2,3,4,5,6,7])
print(arr10[1:5])
```

```
[2 3 4 5]
```

```
#sclicing in 2-dim array
arr11=np.array([[1,2,3,4,5],[6,7,8,9,10]])
print(arr11[0,1:4])
```

```
[2 3 4]
```

```
#addition subtraction multiplication and division
x=np.array([1,2,3,4,5])
y=np.array([1,2,3,4,5])
#addition
add=x+y
print("addition : ",add)
#subtraction
sub=x-y
print("subtraction : ",sub)
#multiplication
mul=x*y
print("multiplication : ",mul)
#division
div=x/y
print("division : ",div)
```

```
addition :  [ 2  4  6  8 10]
subtraction :  [0 0 0 0 0]
multiplication :  [ 1  4  9 16 25]
division :  [1. 1. 1. 1. 1.]
```

```
#shape of array
arr12=np.array([[1,2,3],[4,5,6]])
print(arr12.shape)
```

```
(2, 3)
```

```
#max min sum of array
arr13=np.array([[1,2,3],[4,5,6]])
print(arr13.max())
print(arr13.min())
print(arr13.sum())
```

```
6
1
21
```

```
#axis =0 : col axis =1 : rows
arr14=np.array([[1,2,3],[4,5,6]])
print(arr14.sum(axis=0))
print(arr14.sum(axis=1))
```

```
[5 7 9]
[ 6 15]
```

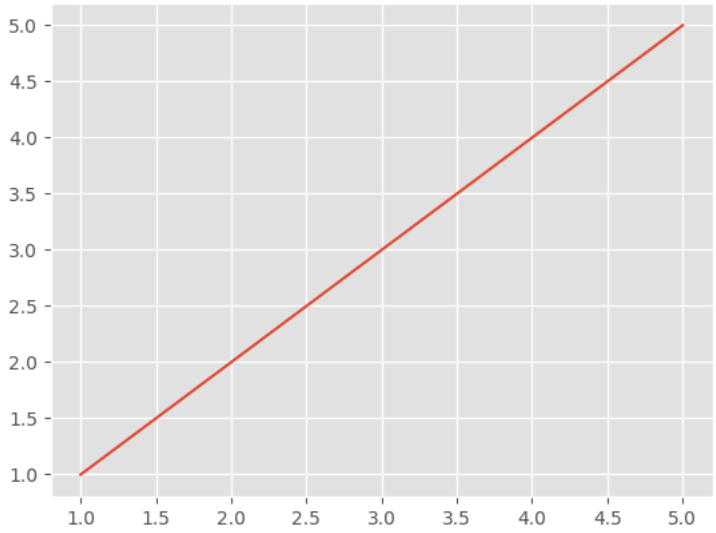
```
#ravel : converting multidim arry into single dim
arr15=np.array([[1,2,3],[4,5,6]])
print(arr15.ndim)
arr16=arr15.ravel()
```

```
print(arr16)
print(arr16.ndim)
```

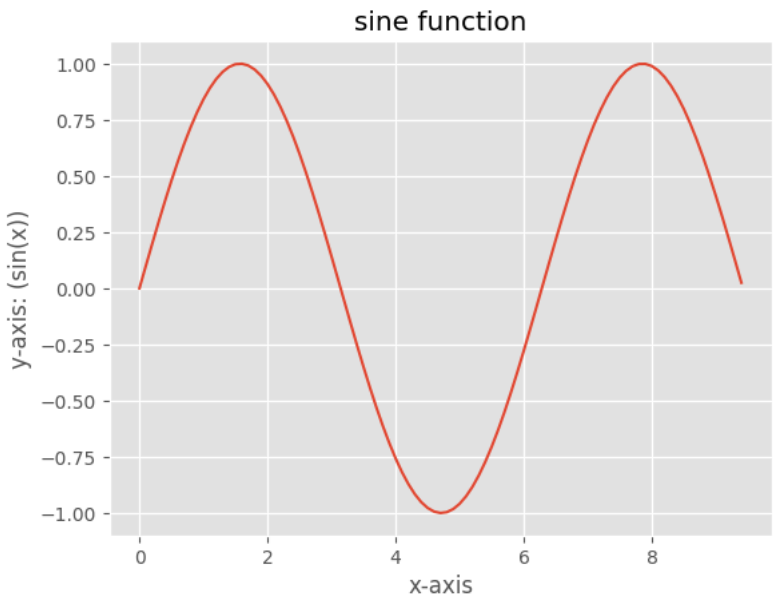
```
2
[1 2 3 4 5 6]
1
```

```
from matplotlib import pyplot as plt
```

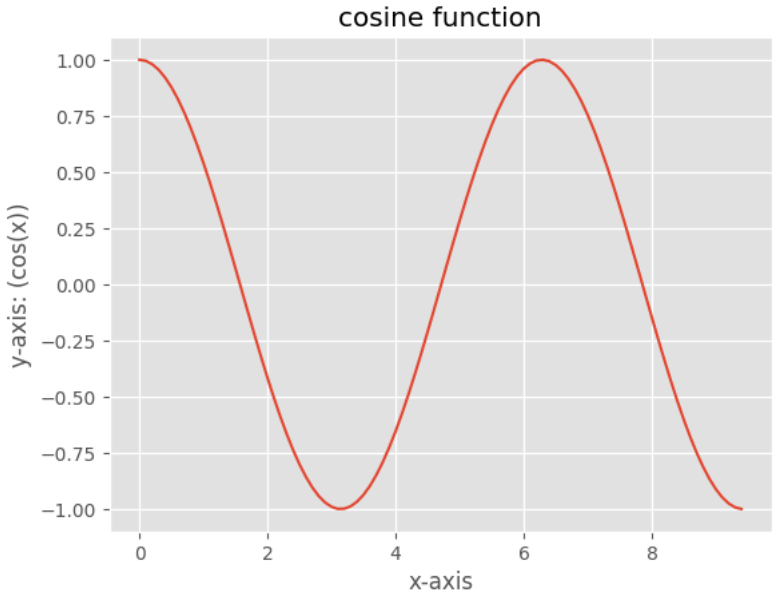
```
#basic line chart
x=np.array([1,2,3,4,5])
y=np.array([1,2,3,4,5])
plt.plot(x,y)
plt.show()
```



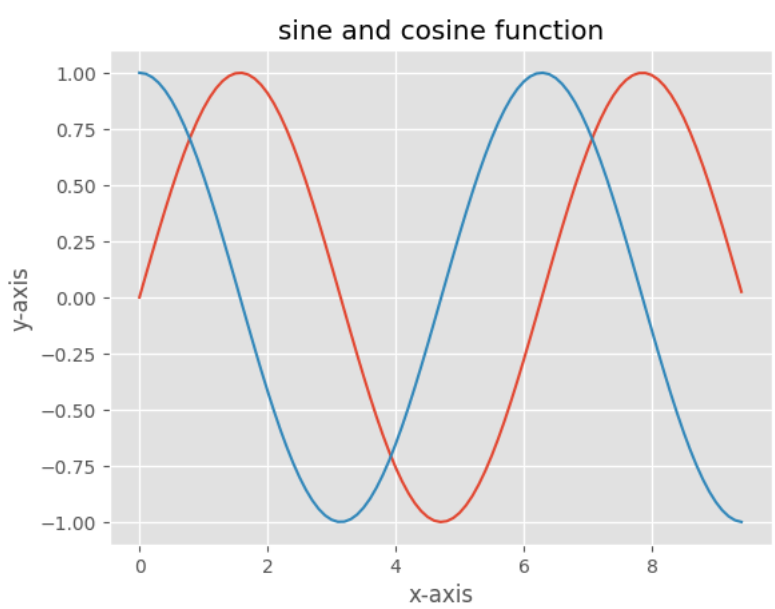

```
#sine function
x=np.arange(0,3*np.pi,0.1)
plt.xlabel("x-axis")
plt.ylabel("y-axis: (sin(x))")
plt.title("sine function")
y=np.sin(x)
plt.plot(x,y)
plt.show()
```



```
#cosine function
x=np.arange(0,3*np.pi,0.1)
plt.xlabel("x-axis")
plt.ylabel("y-axis: (cos(x))")
plt.title("cosine function")
y=np.cos(x)
plt.plot(x,y)
plt.show()
```

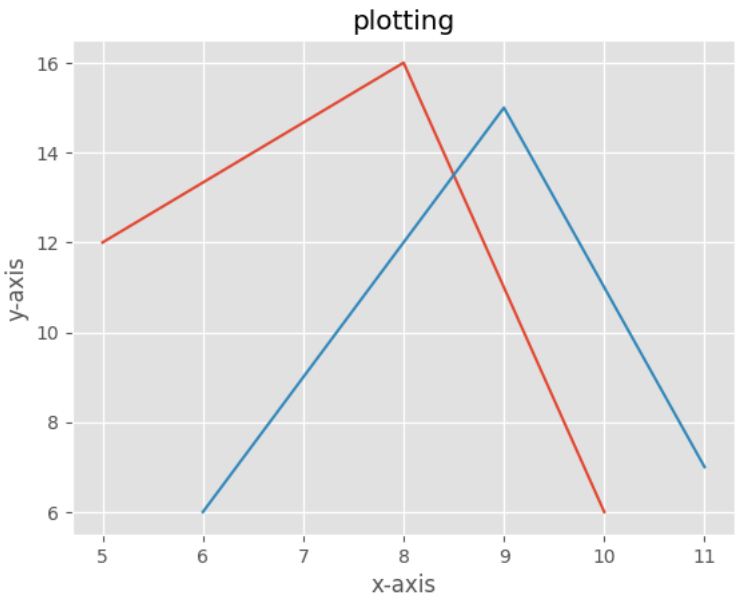



```
#sine and cosine function together
x=np.arange(0,3*np.pi,0.1)
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("sine and cosine function")
y=np.sin(x)
y2=np.cos(x)
plt.plot(x,y)
plt.plot(x,y2)
plt.show()
```

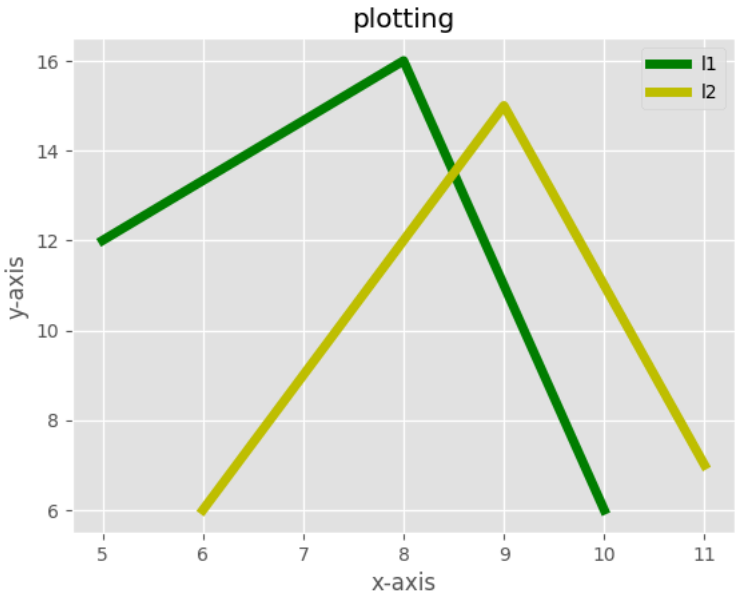



```
#basic graph
x=[5,8,10]
y=[12,16,6]
x2=[6,9,11]
y2=[6,15,7]

plt.plot(x,y)
plt.plot(x2,y2)
plt.title("plotting")
plt.ylabel("y-axis")
```

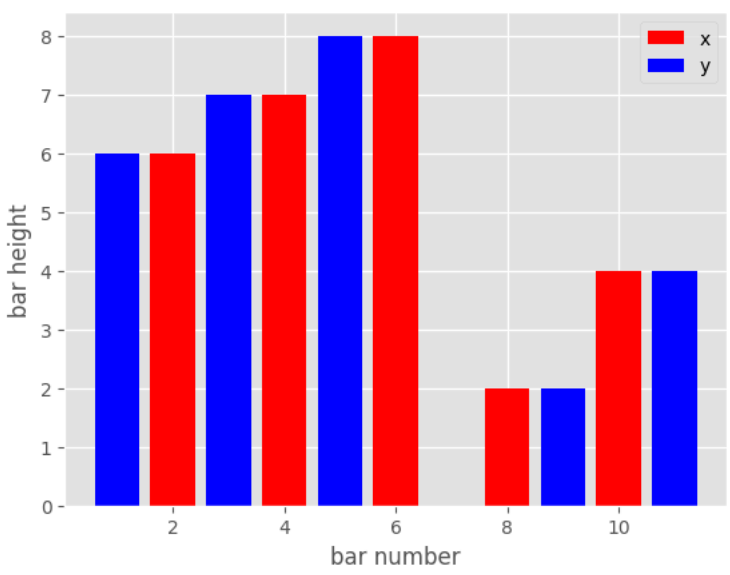
```
plt.xlabel("x-axis")
plt.show()
```



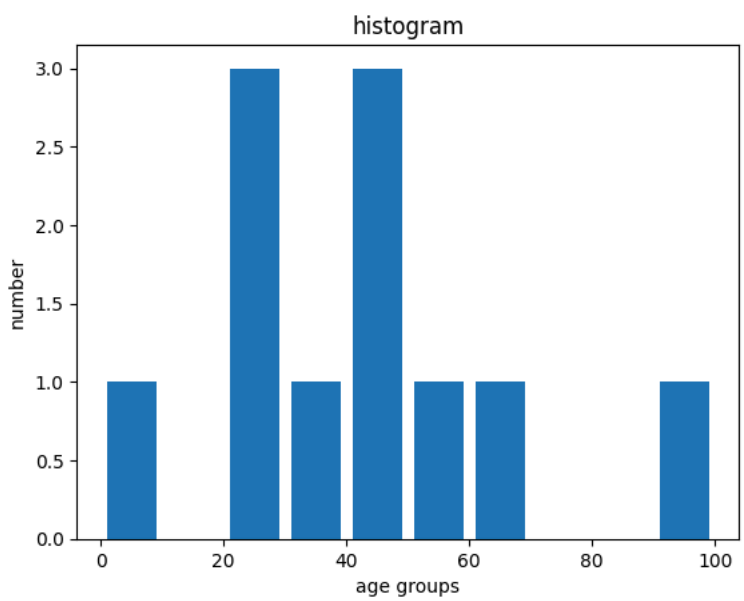

```
#for providing style
from matplotlib import style
style.use("ggplot")

x=[5,8,10]
y=[12,16,6]
x2=[6,9,11]
y2=[6,15,7]
#plt.plot(x-cordinate, y-cordinate,colour,label="label",linewidth="value")
plt.plot(x,y,"g",label="l1",linewidth=5)
plt.plot(x2,y2,"y",label="l2",linewidth=5)
plt.title("plotting")
plt.ylabel("y-axis")
plt.xlabel("x-axis")
plt.legend()
plt.show()
```



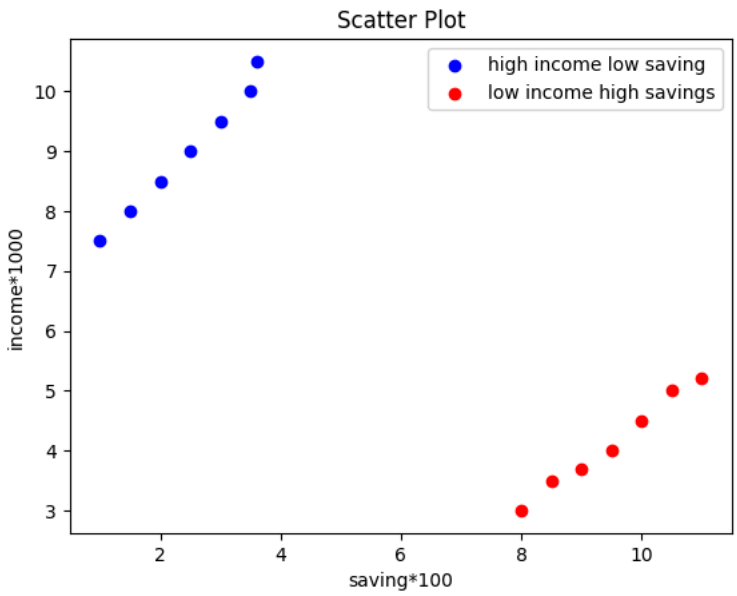

```
#bar chart/graph :compare things
x=[2,4,6,8,10]
y=[6,7,8,2,4]
x2=[1,3,5,9,11]
plt.bar(x,y,label="x",color="r")
plt.bar(x2,y,label="y",color="b")
plt.legend()
plt.xlabel("bar number")
plt.ylabel("bar height")
plt.show()
```



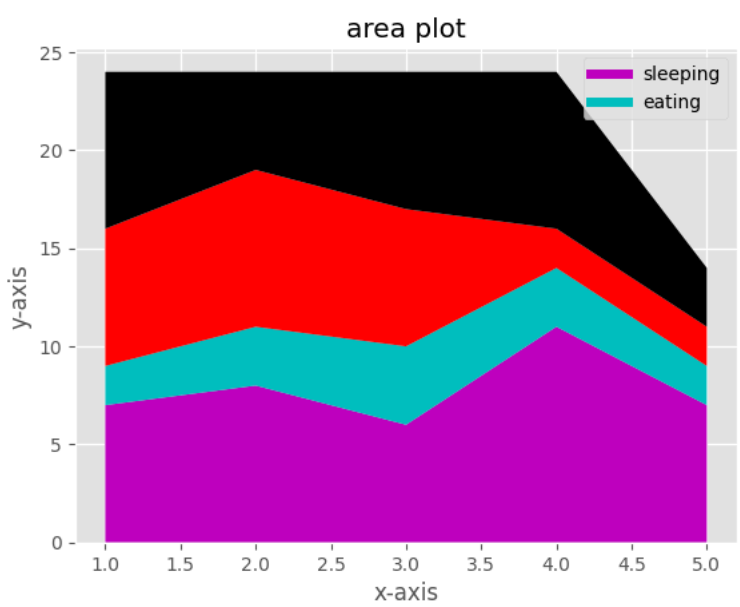

```
#histogram: discrete or continous data
population_age=[22,55,62,45,21,22,34,42,42,4,99]
bins=[0,10,20,30,40,50,60,70,80,90,100]
plt.hist(population_age,bins,histtype="bar",rwidth=0.8)
plt.xlabel("age groups")
plt.ylabel("number")
plt.title("histogram")
plt.show()
```




```
#scatter plot

x = [1,1.5,2,2.5,3,3.5,3.6]
y = [7.5,8,8.5,9,9.5,10,10.5]

x1=[8,8.5,9,9.5,10,10.5,11]
y1=[3,3.5,3.7,4,4.5,5,5.2]

plt.scatter(x,y, label='high income low saving',color='b')
plt.scatter(x1,y1,label='low income high savings',color='r')

plt.xlabel('saving*100')
plt.ylabel('income*1000')
plt.title('Scatter Plot')

plt.legend()
plt.show()
```




```
#area plot: used to show how things changes over time
days=[1,2,3,4,5]
sleeping=[7,8,6,11,7]
eating=[2,3,4,3,2]
working=[7,8,7,2,2]
playing=[8,5,7,8,3]
plt.plot([],[],color='m',label='sleeping',linewidth=5)
plt.plot([],[],color='c',label='eating',linewidth=5)
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("area plot")
plt.stackplot(days,sleeping,eating,working,playing,colors=['m','c','r','k'])
plt.legend()
plt.show()
```




```
# Pie Chart (Pie charts are generally used to show percentage or proportional data )

slices = [7,2,2,13]
activities = ['sleeping','eating','working','playing']
cols = ['y','m','r','g']

plt.pie(slices,
labels = activities,
colors = cols,
startangle = 90,
shadow = True,
explode = (0,0,0.1,0),
autopct = '%1.0f%%')
# autopct = '%1.2f%%')
```

```
plt.title('Pie chart')
plt.show()
```


```
import pandas as pd


#creating dataframe
#using a csv file
df=pd.read_csv("/content/drive/MyDrive/pythonLibraries/Weather.csv")
df
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
# Create your own dataframe
#using dic
weather_data={
    'day':['1/1/2017','1/2/2017','1/3/2017','1/4/2017','1/5/2017','1/6/2017'],
    'temperature':[32,35,28,24,32,31],
    'windspeed':[6,7,2,7,4,2],
    'event':['Rain','Sunny','Snow','Snow','Rain','Sunny']
}
#converting dict to dataframe
df1=pd.DataFrame(weather_data)
print(type(weather_data))
df1
```

```
<class 'dict'>
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#using tuple list
weather_data=[
    ('1/1/2017',32,6,'Rain'),
    ('1/2/2017',35,7,'Sunny'),
    ('1/3/2017',28,2,'Snow')
]
print(type(weather_data))
```

```
df2=pd.DataFrame(weather_data,columns=['day','temperature','windspeed','event'])
df2
```

```
<class 'list'>
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |

```
#save to csv
df.to_csv("/content/drive/MyDrive/pythonLibraries/new.csv")
```

```
#shape of df
df.shape
```

```
(6, 4)
```

```
df
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#to display first 5 lines
df.head()
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |

```
#to display particular number of lines from top
df.head(2)
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |

```
#to display last 5 lines
df.tail()
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#to display particular number of lines from bottom
df.tail(3)
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#sclicing the dataframe : extracting particular set of dataframes
df[2:4]
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |

```
#get col
df.columns
```

```
Index(['day', 'temperature', 'windspeed', 'event'], dtype='object')
```

```
#extracting single col
df['day']
```

| | day |
|---|---|
| **0** | 1/1/2017 |
| **1** | 1/2/2017 |
| **2** | 1/3/2017 |
| **3** | 1/4/2017 |
| **4** | 1/5/2017 |
| **5** | 1/6/2017 |

**dtype:** object

```
df['temperature']
```

| | temperature |
|---|---|
| **0** | 32 |
| **1** | 35 |
| **2** | 28 |
| **3** | 24 |
| **4** | 32 |
| **5** | 31 |

**dtype:** int64

```
#multiple col
df[['day','temperature']]
```

| | day | temperature |
|---|---|---|
| **0** | 1/1/2017 | 32 |
| **1** | 1/2/2017 | 35 |
| **2** | 1/3/2017 | 28 |
| **3** | 1/4/2017 | 24 |
| **4** | 1/5/2017 | 32 |
| **5** | 1/6/2017 | 31 |

```
df['temperature']
```

| | temperature |
|---|---|
| **0** | 32 |
| **1** | 35 |
| **2** | 28 |
| **3** | 24 |
| **4** | 32 |
| **5** | 31 |

**dtype:** int64

```
#max temp
df['temperature'].max()
```

```
35
```

```
#min temp
df['temperature'].min()
```

```
24
```

```
#mean temp
df['temperature'].mean()
```

```
30.333333333333332
```

```
#deviation
df['temperature'].std()
```

```
3.8297084310253524
```

```
#description of dataframe : only numeric fields
df.describe()
```

| | temperature | windspeed |
|---|---|---|
| **count** | 6.000000 | 6.000000 |
| **mean** | 30.333333 | 4.666667 |
| **std** | 3.829708 | 2.338090 |
| **min** | 24.000000 | 2.000000 |
| **25%** | 28.750000 | 2.500000 |
| **50%** | 31.500000 | 5.000000 |
| **75%** | 32.000000 | 6.750000 |
| **max** | 35.000000 | 7.000000 |

```
df
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#printing day when temp was max
df['day'][df['temperature']==df['temperature'].max()]
```

| | day |
|---|---|
| **1** | 1/2/2017 |

**dtype:** object

```
df
```

| | day | temperature | windspeed | event |
|---|---|---|---|---|
| **0** | 1/1/2017 | 32 | 6 | Rain |
| **1** | 1/2/2017 | 35 | 7 | Sunny |
| **2** | 1/3/2017 | 28 | 2 | Snow |
| **3** | 1/4/2017 | 24 | 7 | Snow |
| **4** | 1/5/2017 | 32 | 4 | Rain |
| **5** | 1/6/2017 | 31 | 2 | Sunny |

```
#handling the missing values
new_df=pd.read_csv("/content/drive/MyDrive/pythonLibraries/weathernew.csv")
new_df
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|
| **0** | 0 | 2017-01-01 | 32.0 | 6.0 | Rain |
| **1** | 1 | 2017-01-04 | NaN | 9.0 | Sunny |
| **2** | 2 | 2017-01-05 | 28.0 | NaN | Snow |
| **3** | 3 | 2017-01-06 | NaN | 7.0 | NaN |
| **4** | 4 | 2017-01-07 | 32.0 | NaN | Rain |
| **5** | 5 | 2017-01-08 | NaN | NaN | Sunny |
| **6** | 6 | 2017-01-09 | NaN | NaN | NaN |
| **7** | 7 | 2017-01-10 | 34.0 | 8.0 | Cloudy |
| **8** | 8 | 2017-01-11 | 40.0 | 12.0 | Sunny |

```
#fillna : to fill the missing values to
newdf1=new_df.fillna(10)
newdf1
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|
| **0** | 0 | 2017-01-01 | 32.0 | 6.0 | Rain |
| **1** | 1 | 2017-01-04 | 10.0 | 9.0 | Sunny |
| **2** | 2 | 2017-01-05 | 28.0 | 10.0 | Snow |
| **3** | 3 | 2017-01-06 | 10.0 | 7.0 | 10 |
| **4** | 4 | 2017-01-07 | 32.0 | 10.0 | Rain |
| **5** | 5 | 2017-01-08 | 10.0 | 10.0 | Sunny |
| **6** | 6 | 2017-01-09 | 10.0 | 10.0 | 10 |
| **7** | 7 | 2017-01-10 | 34.0 | 8.0 | Cloudy |
| **8** | 8 | 2017-01-11 | 40.0 | 12.0 | Sunny |

```
#fillna using col name and dict
newdf2=new_df.fillna({
    'temperature':27,
    'windspeed':5,
    'event':'no event'
})
newdf2
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|

```
#bfill and ffill
#fill
new_df
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|
| **0** | 0 | 2017-01-01 | 32.0 | 6.0 | Rain |
| **1** | 1 | 2017-01-04 | NaN | 9.0 | Sunny |
| **2** | 2 | 2017-01-05 | 28.0 | NaN | Snow |
| **3** | 3 | 2017-01-06 | NaN | 7.0 | NaN |
| **4** | 4 | 2017-01-07 | 32.0 | NaN | Rain |
| **5** | 5 | 2017-01-08 | NaN | NaN | Sunny |
| **6** | 6 | 2017-01-09 | NaN | NaN | NaN |
| **7** | 7 | 2017-01-10 | 34.0 | 8.0 | Cloudy |
| **8** | 8 | 2017-01-11 | 40.0 | 12.0 | Sunny |

```
newdf3=new_df.fillna(method='ffill')
newdf3
```

```
<ipython-input-444-c450d7e8c7d6>:1: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a futu
  newdf3=new_df.fillna(method='ffill')
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|
| **0** | 0 | 2017-01-01 | 32.0 | 6.0 | Rain |
| **1** | 1 | 2017-01-04 | 32.0 | 9.0 | Sunny |
| **2** | 2 | 2017-01-05 | 28.0 | 9.0 | Snow |
| **3** | 3 | 2017-01-06 | 28.0 | 7.0 | Snow |
| **4** | 4 | 2017-01-07 | 32.0 | 7.0 | Rain |
| **5** | 5 | 2017-01-08 | 32.0 | 7.0 | Sunny |
| **6** | 6 | 2017-01-09 | 32.0 | 7.0 | Sunny |
| **7** | 7 | 2017-01-10 | 34.0 | 8.0 | Cloudy |
| **8** | 8 | 2017-01-11 | 40.0 | 12.0 | Sunny |

```
#bfill
newdf4=new_df.fillna(method='bfill')
newdf4
```

```
<ipython-input-445-074f6ea430b7>:2: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a futu
  newdf4=new_df.fillna(method='bfill')
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|
| **0** | 0 | 2017-01-01 | 32.0 | 6.0 | Rain |
| **1** | 1 | 2017-01-04 | 28.0 | 9.0 | Sunny |
| **2** | 2 | 2017-01-05 | 28.0 | 7.0 | Snow |
| **3** | 3 | 2017-01-06 | 32.0 | 7.0 | Rain |
| **4** | 4 | 2017-01-07 | 32.0 | 8.0 | Rain |
| **5** | 5 | 2017-01-08 | 34.0 | 8.0 | Sunny |
| **6** | 6 | 2017-01-09 | 34.0 | 8.0 | Cloudy |
| **7** | 7 | 2017-01-10 | 34.0 | 8.0 | Cloudy |
| **8** | 8 | 2017-01-11 | 40.0 | 12.0 | Sunny |

```
#dropns
newdf5=new_df.dropna()
newdf5
```

| | Unnamed: 0 | day | temperature | windspeed | event |
|---|---|---|---|---|---|