MATPLOT VISUALIZATION , SEABORN

June 29, 2022

[37]:
```
#MATPLOT VISUAIZATION
```

[38]:
```
#its a visualization library
#matplot isnt used much but seaborn is used.
#but this is the basics before going to the seaborn
```

[39]:
```
#before using matplot we have to write 2 codes which are must to be followed␣
 ↪while using jupyter notebook
```

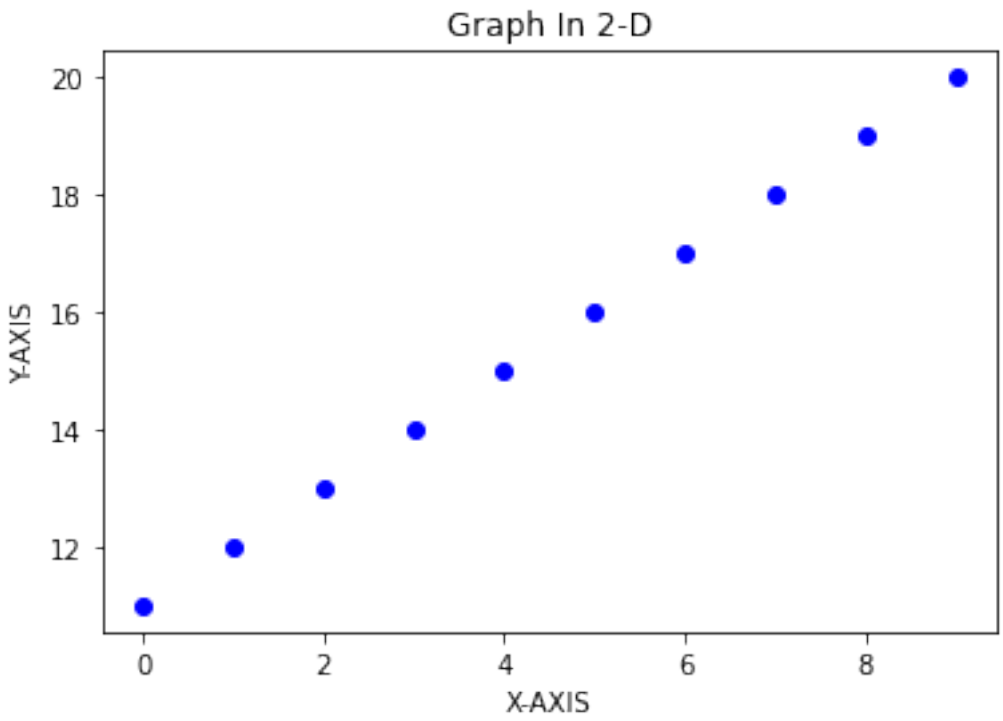
[40]:
```
#inbuilt functions in matplot lib:
#1 .scatter()-this function is used to scatter the values of x and y in a 2-d␣
 ↪graph
#2 .xlabel()-this function is used to label the x axis
#3 .ylabel()-this function is used to label y axis
#4 .title()-using this function we can label the graph
#5 .savefig()-using this function we can the save the image of the graph in our␣
 ↪folder
#5 .plot()-this function is used to plot the graph
#6 .subplot()-this is used to plot multiple slots in one slot
```

[41]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
```

[42]:
```
x=np.arange(0,10)
y=np.arange(11,21)
```

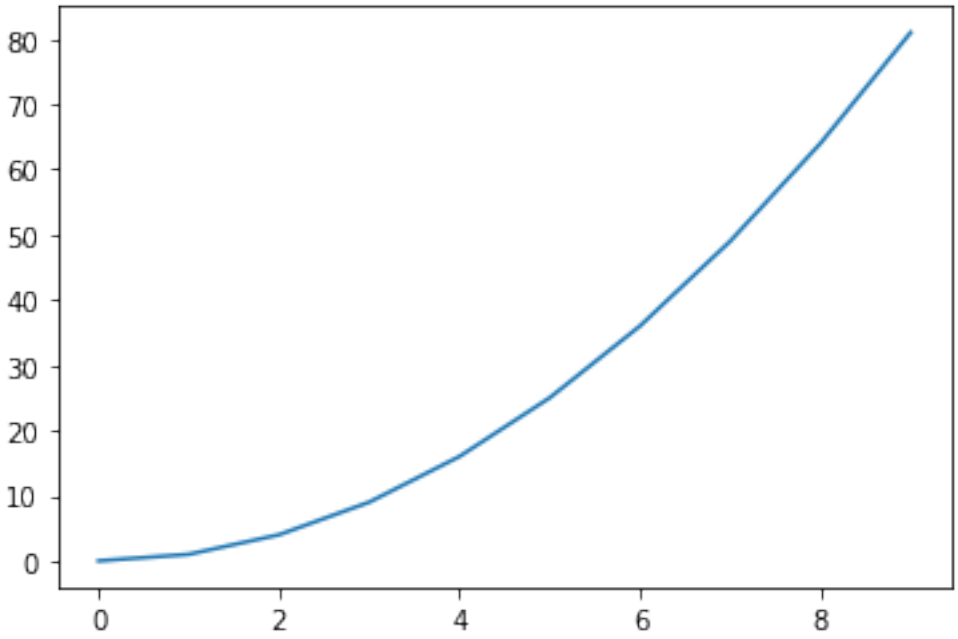
[43]:
```
#below is an simple example
plt.scatter(x,y,c='b')
plt.xlabel('X-AXIS')
plt.ylabel('Y-AXIS')
plt.title('Graph In 2-D')
```

[43]:
```
Text(0.5, 1.0, 'Graph In 2-D')
```



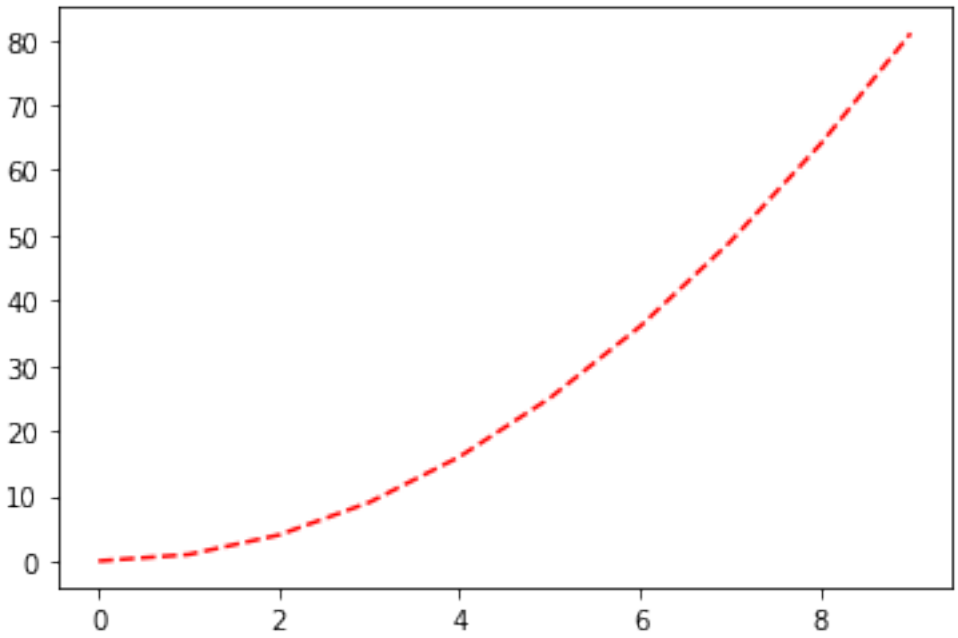
[44]:
```
y=x*x
plt.plot(x,y)
#the y axis increases because we have given y=x*x.
```

[44]: [<matplotlib.lines.Line2D at 0x2501e5a3130>]



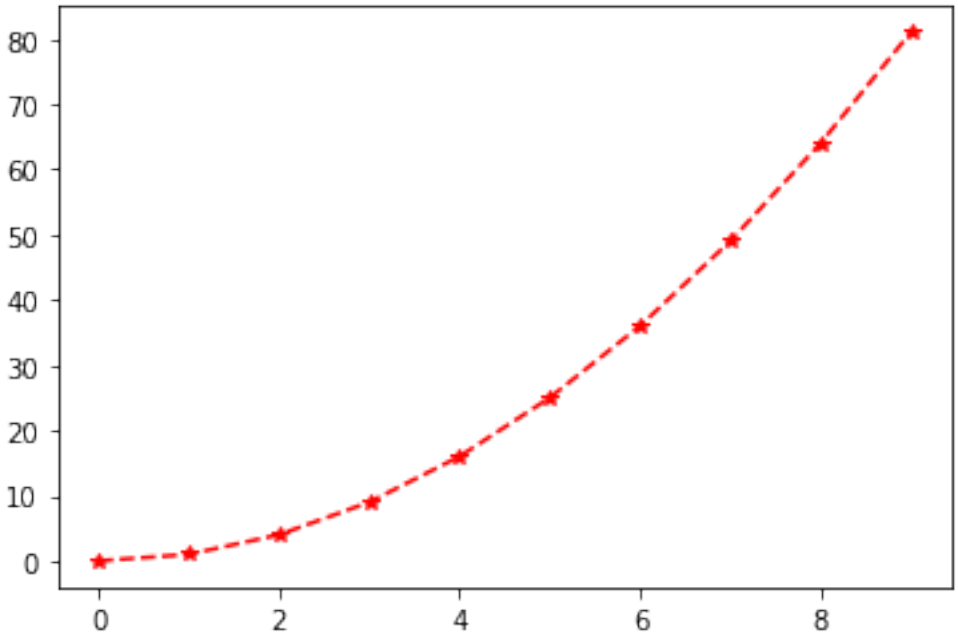
[45]:
```
#there are some paramaters we can use in plot function, follow the below code
plt.plot(x,y,'r--')
#the red line appears because we have mentioned r--
```

[45]: [<matplotlib.lines.Line2D at 0x2501e5cbcd0>]



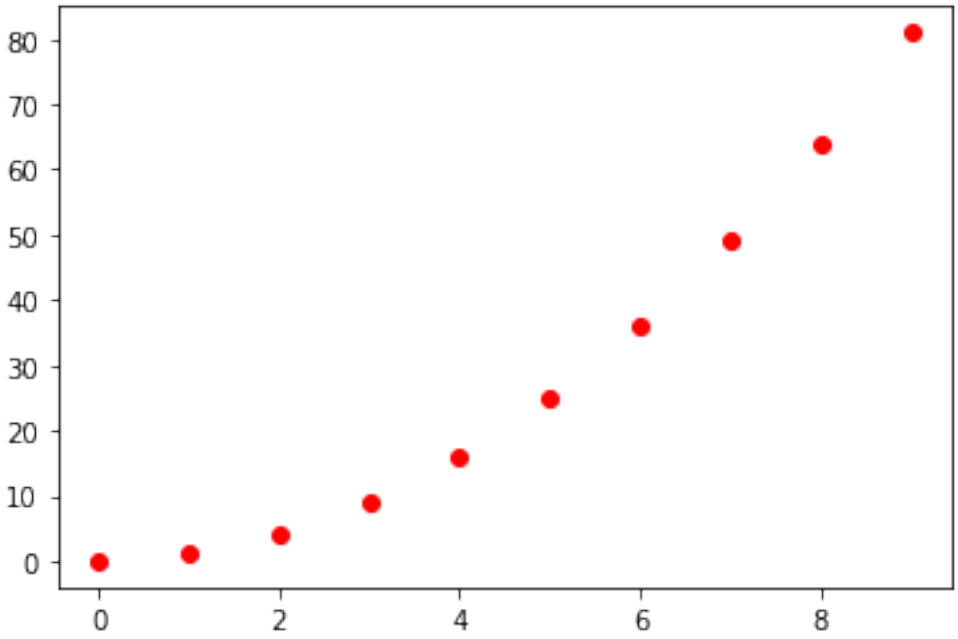
[46]:
```
plt.plot(x,y,'r*--')
#the star appears because we have given r*
```

[46]: [<matplotlib.lines.Line2D at 0x2502039b970>]



[47]:
```
plt.plot(x,y,'ro')
```

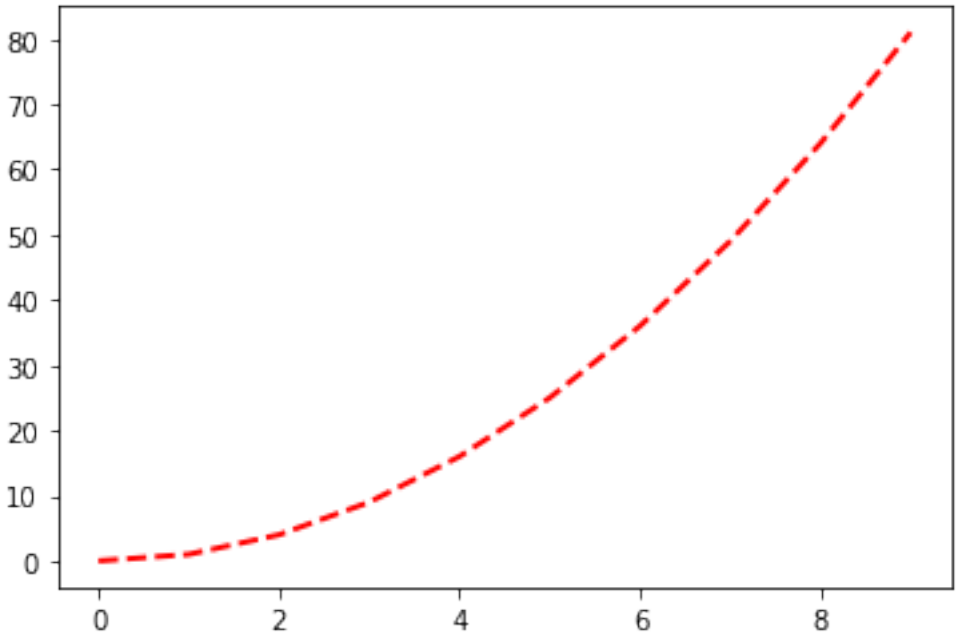
[47]: [<matplotlib.lines.Line2D at 0x2501e571940>]



[48]:
```
#these are the different ways of using the plot function
```

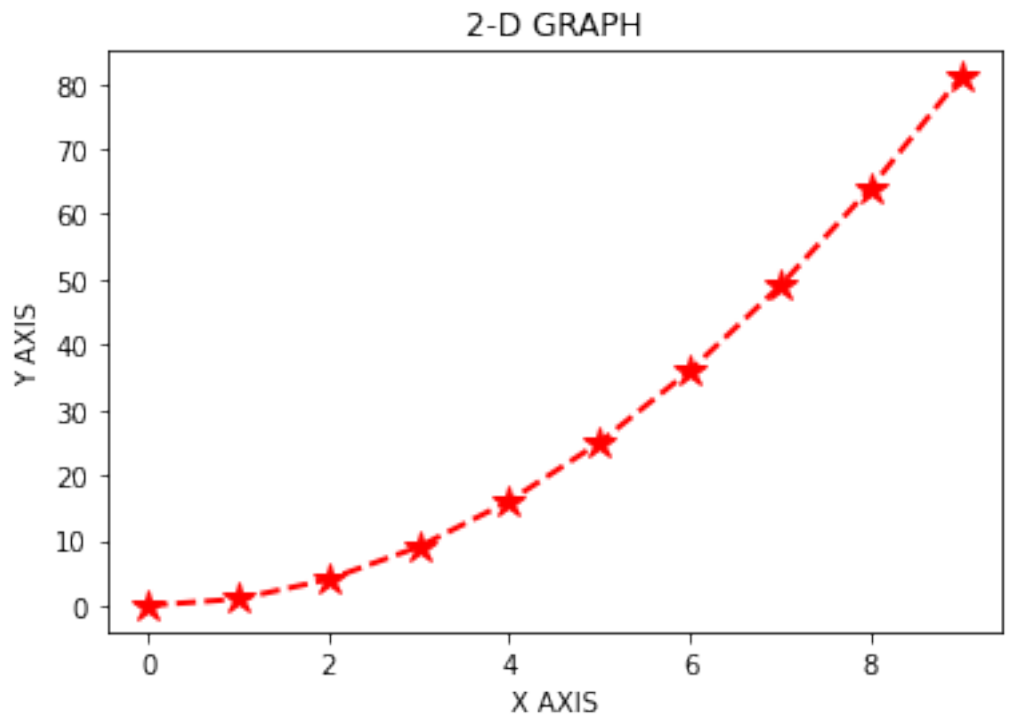
[49]:
```
#there are somemore paramters that can be used while using plot function
#linewidth - thickeness of the graph line
#markersize - thickness of the marks placed
#linestyle - the style of the line can be specified here
plt.plot(x,y,'r',linestyle='dashed',linewidth=2,markersize=12)
```

[49]: [<matplotlib.lines.Line2D at 0x25020458f70>]



[50]:
```
#we can also use in this way
plt.plot(x,y,'r*',linestyle='dashed',linewidth=2,markersize=12)
plt.xlabel('X AXIS')
plt.ylabel('Y AXIS')
plt.title('2-D GRAPH')
```

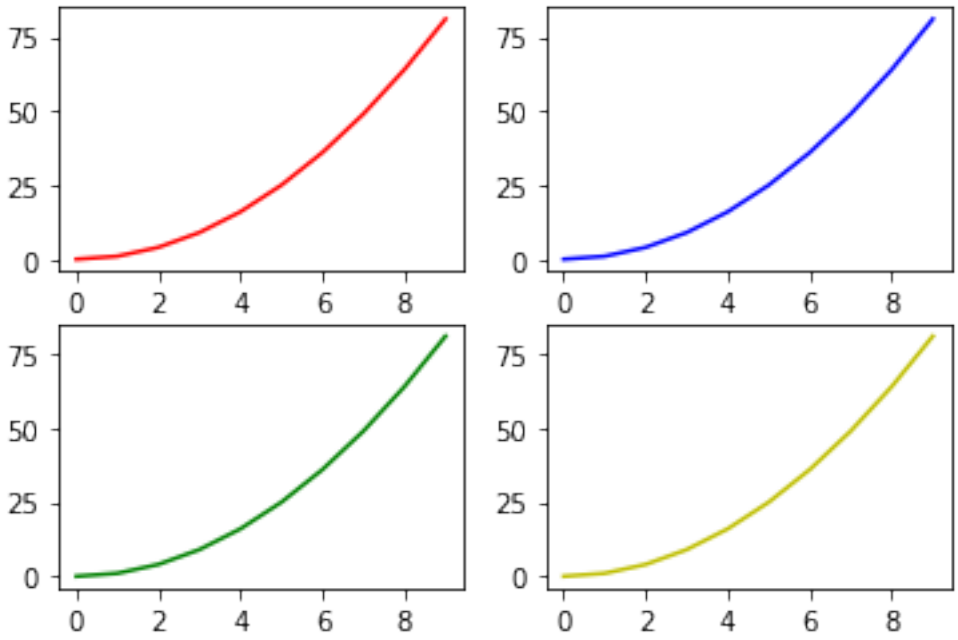
[50]: Text(0.5, 1.0, '2-D GRAPH')



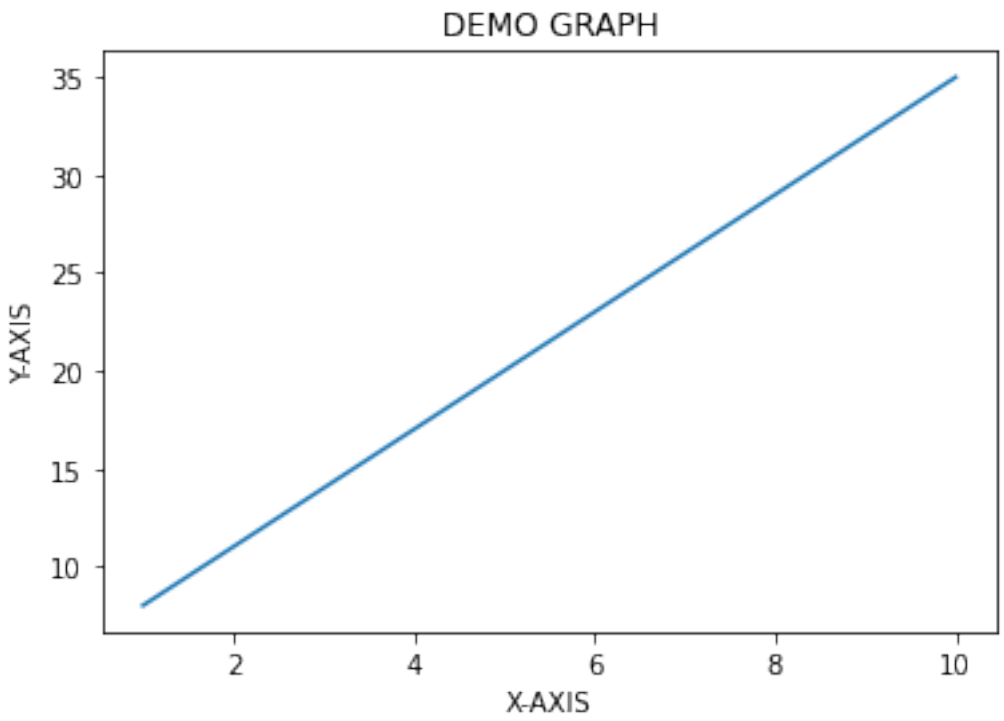
[51]:
```
#creating subplots
#subplots means creating multiple subplots in one slot
#in this subplot parameter there are three things to be mentioned one is row ,␣
 ↪columns and the final one index
plt.subplot(2,2,1)
plt.plot(x,y,'r')

#this means we are mentioning 2 rows 2 columns and the graph has to be placed␣
 ↪in first position
plt.subplot(2,2,2)
plt.plot(x,y,'b')

#this means we are mentioning 2 rows 2 columns and the graph is to be placed in␣
 ↪the second pos
plt.subplot(2,2,3)
plt.plot(x,y,'g')

#this means we are mentioning 2 rows 2 columns and the graph is to be placed in␣
 ↪the third pos
plt.subplot(2,2,4)
plt.plot(x,y,'y')
```

```
#this means we are mentioning 2 rows 2 columns and the graph is to be placed in␣
 ↪the 4th position
```

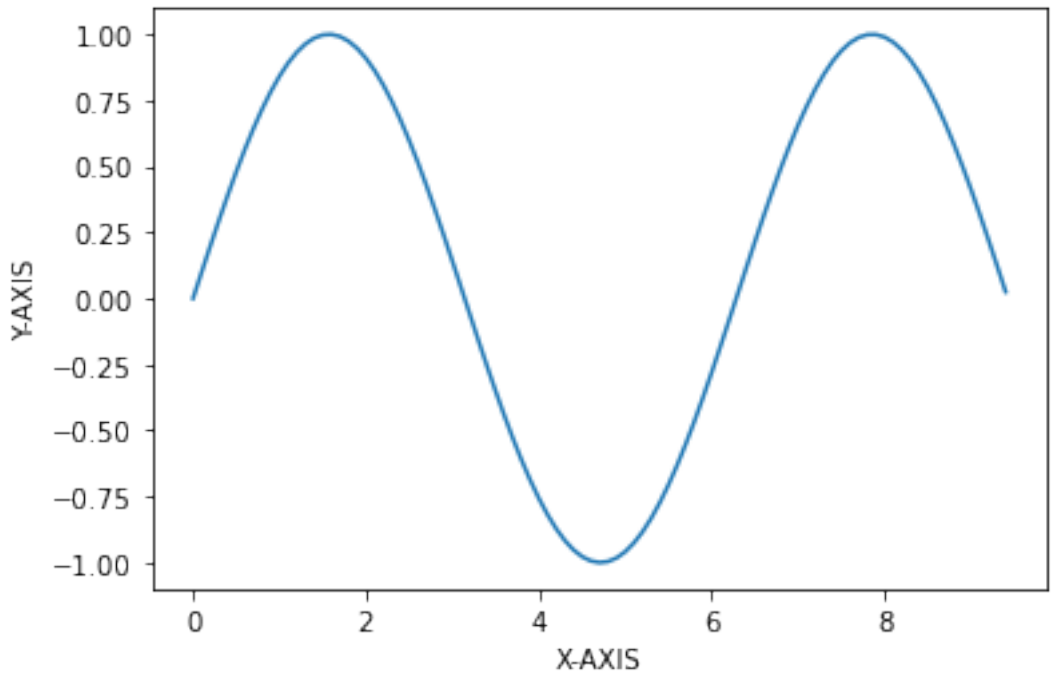
[51]: [<matplotlib.lines.Line2D at 0x250205be970>]



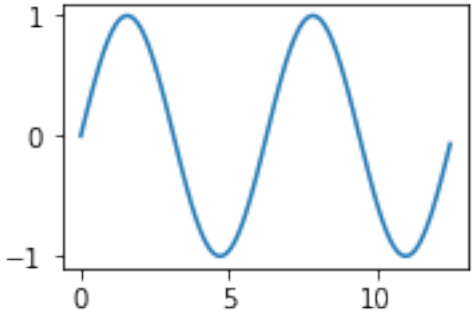
[52]:
```
x=np.arange(1,11)
y=3*x+5
plt.xlabel('X-AXIS')
plt.ylabel('Y-AXIS')
plt.title('DEMO GRAPH')
plt.plot(x,y,)
plt.show()
```

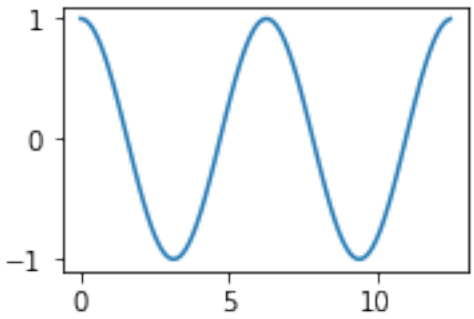


[53]:
```
#creating a sin curve using matplot lin
x=np.arange(0,3*np.pi,0.1)
y=np.sin(x)
plt.plot(x,y)
plt.xlabel('X-AXIS')
plt.ylabel('Y-AXIS')
plt.show()
```

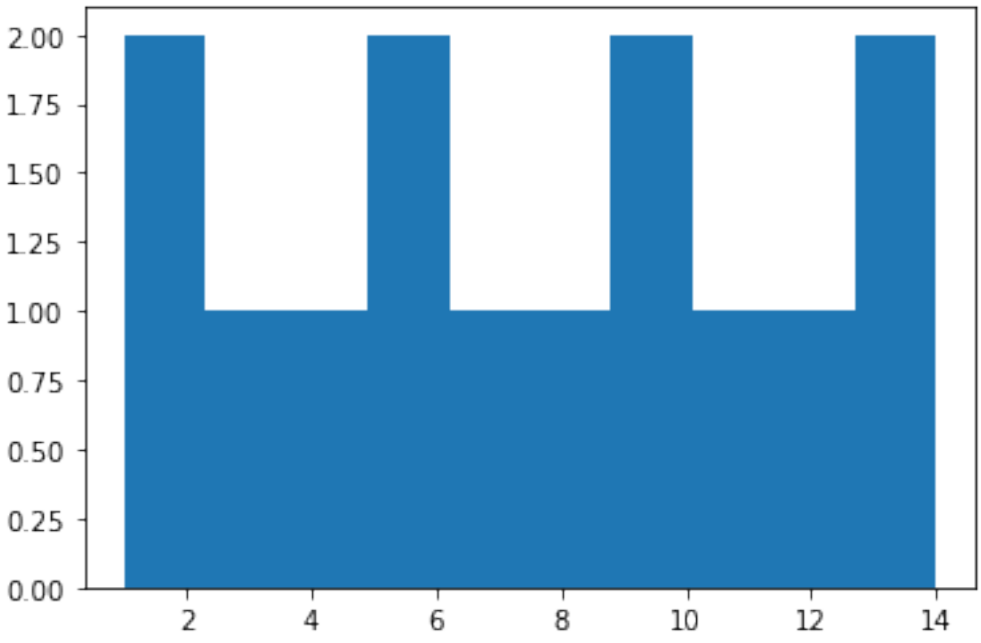


[54]:
```
#drawing sin waves and cos waves using subplot
x=np.arange(0,4*np.pi,0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)
plt.subplot(2,2,1)
plt.plot(x,y_sin)
plt.show()
#this above plot is for the sin curve

plt.subplot(2,2,2)
plt.plot(x,y_cos)
plt.show()
#this above plot is for the cos curve
```



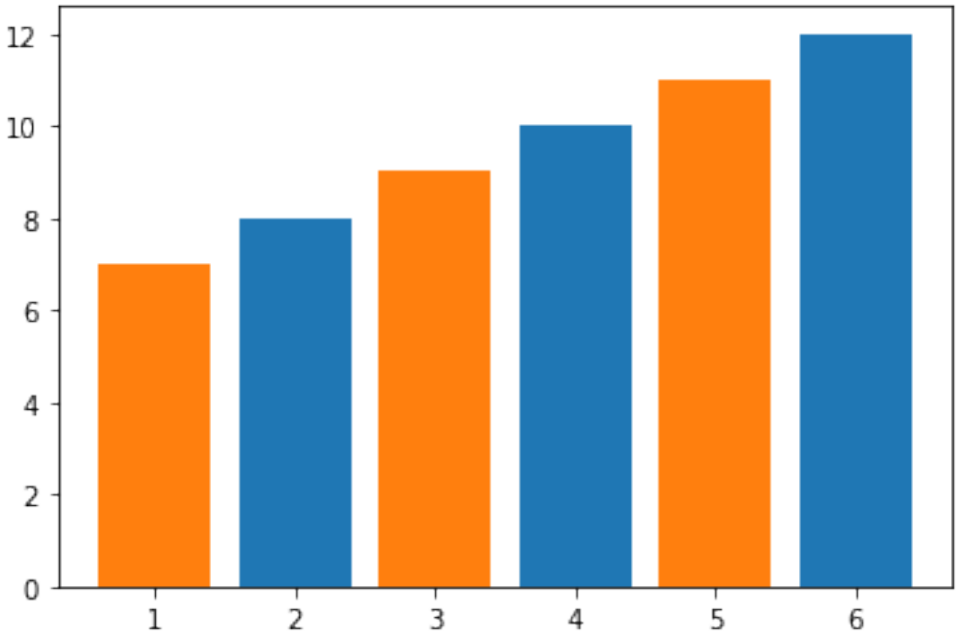


[55]:
```
#creating histogram(data distribution)

a=np.array([1,2,3,4,5,6,7,8,9,10,11,12,13,14])
plt.hist(a)
plt.show()
```



[56]:
```
#creating bar graph(data distribution)
x=[2,4,6]
y=[8,10,12]
plt.bar(x,y)
```

```
x2=[1,3,5]
y2=[7,9,11]
plt.bar(x2,y2)
plt.show()
```



[58]:
```
#distributions and different mode ( machine learning):
#mean-the average value
#median-the midpoint value
#mode-the value that has occured most times
```

[59]:
```
#finding the mean value
# .mean()-this function is used to find the value of mean

speed = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18]
mean_value = np.mean(speed)
print(mean_value)
```

```
9.5
```

[60]:
```
#finding the median
# .median()-this function is used to find the value of the median
speed = [5,6,7,8,9,15,88,34,22,5]
median_value=np.median(speed)
print(median_value)
```

```
8.5
```

[61]:
```
#finding the mode
#to calculate the mode we have to use 2 things we have to import scipy and use␣
 ↪mode() functions

from scipy import stats

speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]

x = stats.mode(speed)

print(x)
```

```
ModeResult(mode=array([86]), count=array([3]))
```

[62]:
```
#standard deviations
#Standard deviation is a number that describes how spread out the values are.
#A low standard deviation means that most of the numbers are close to the mean␣
 ↪(average) value.
#A high standard deviation means that the values are spread out over a wider␣
 ↪range.
#to understand this better we have to follow the below code
```

[63]:
```
#low standard deviation means the most of the numbers are distrubuted close to␣
 ↪the mean value
#high standard deviation means most of the numbers are distrubuted far away␣
 ↪from the mean value
```

[64]:
```
speed = [86,87,88,86,87,85,86]
meanzz = np.mean(speed)

std_dev = np.std(speed)

print("the mean value of the above array is : ",meanzz)
print("the value of standard deviation : ",std_dev)
#from the below ouput we can understand the mean value is 86.4 and the values␣
 ↪are distributed through cl0se to mean value
#that means its low standard deviation
```

```
the mean value of the above array is :  86.42857142857143
the value of standard deviation :  0.9035079029052513
```

[65]:
```
speed = [32,111,138,28,59,77,97]
meanss=np.mean(speed)
std_dev = np.std(speed)
#this output values says it hight standard deviation
print(meanss)
print(std_dev)
```

77.42857142857143
37.84501153334721

[66]:
```
#percentiles
#_Percentiles are used in statistics to give you a number that describes the␣
 ↪value that a given percent of the values are lower than.
```

[67]:
```
ages = [5,31,43,48,50,41,7,11,15,39,80,82,32,2,8,6,25,36,27,61,31]

x = np.percentile(ages, 75)

print(x)
```

43.0

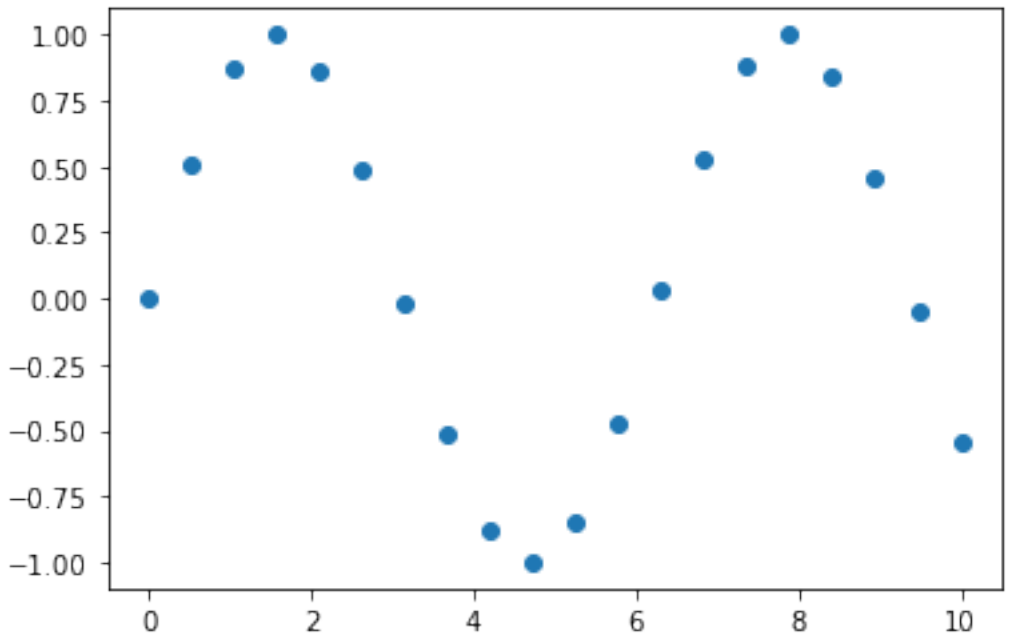
[68]:
```
#creating pie chart
#some things to be learned before learning to code pie chart
#labels-the data you want to be plotted
#sizes-its the proportion to the data you want to be plotted
#colours-is the colours you want to give to the different data
#explode-its the data you want to be splitted
#shadow-its the shadow or the extra graphics

y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
mycolors = ["black", "hotpink", "b", "#4CAF50"]

plt.pie(y, labels = mylabels, colors = mycolors)
plt.show()
```

[69]:
```
#SKILLVERTEX NOTES
```

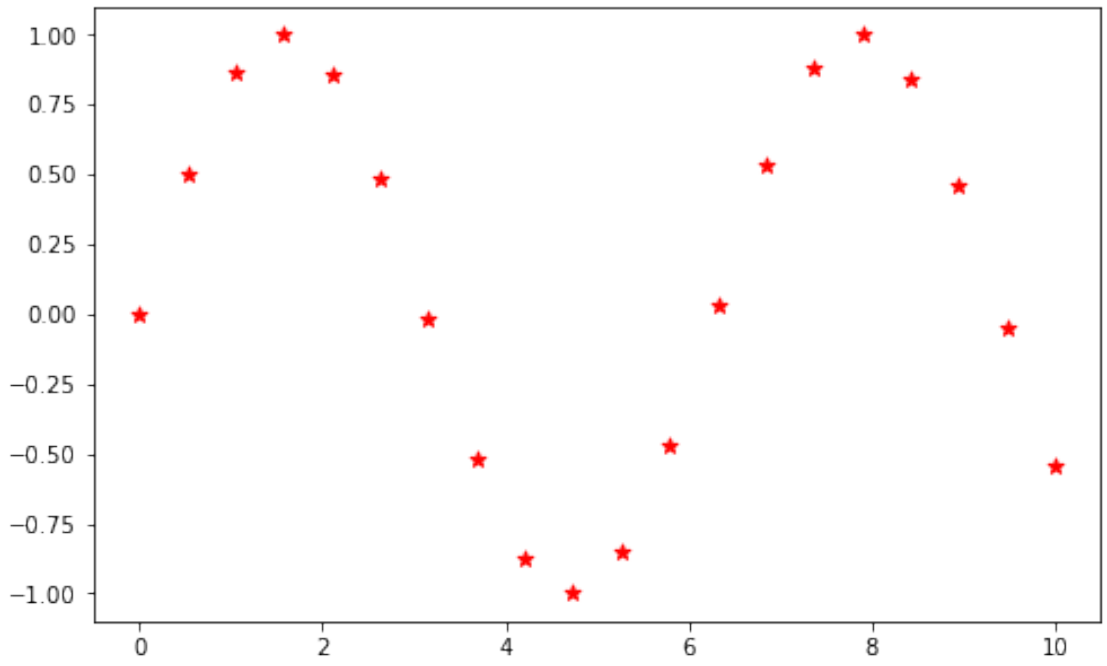
[70]:
```
#linspace(start,stop,value)-
np.linspace(0,10,20)
#this means we are getting an array of 20 values from 0-10
```

[70]:
```
array([ 0.        ,  0.52631579,  1.05263158,  1.57894737,  2.10526316,
        2.63157895,  3.15789474,  3.68421053,  4.21052632,  4.73684211,
        5.26315789,  5.78947368,  6.31578947,  6.84210526,  7.36842105,
        7.89473684,  8.42105263,  8.94736842,  9.47368421, 10.        ])
```

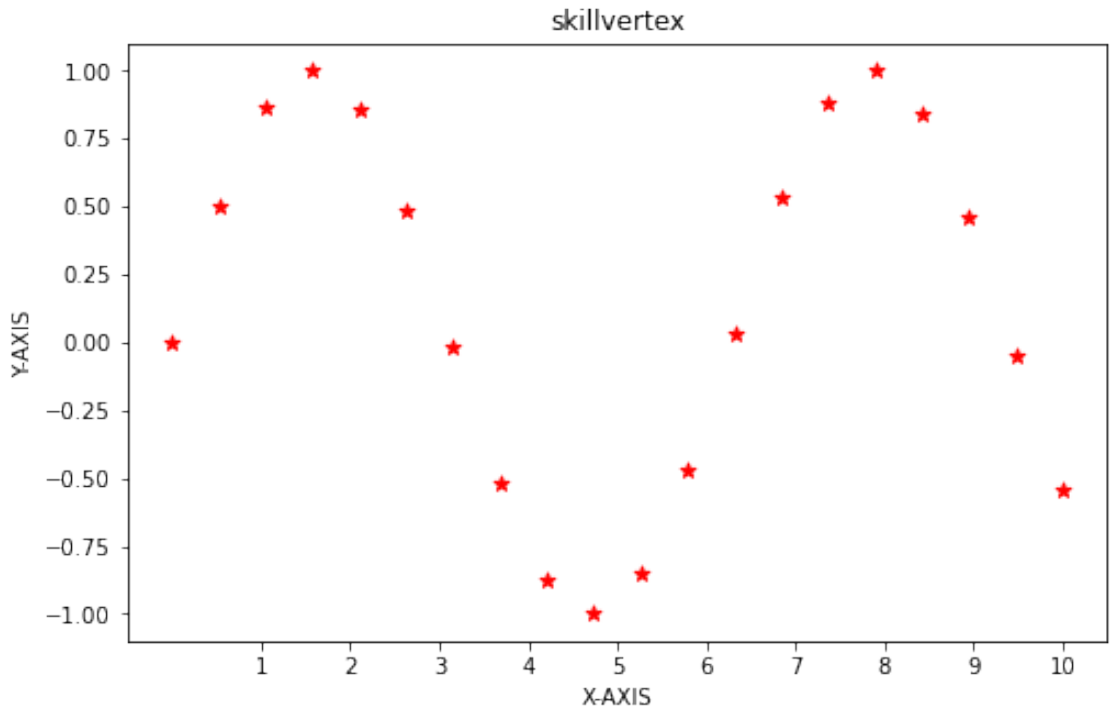
[73]:
```
#using scatter function
x=np.linspace(0,10,20)
y=np.sin(x)
sct_val=plt.scatter(x,y)
plt.show()
```



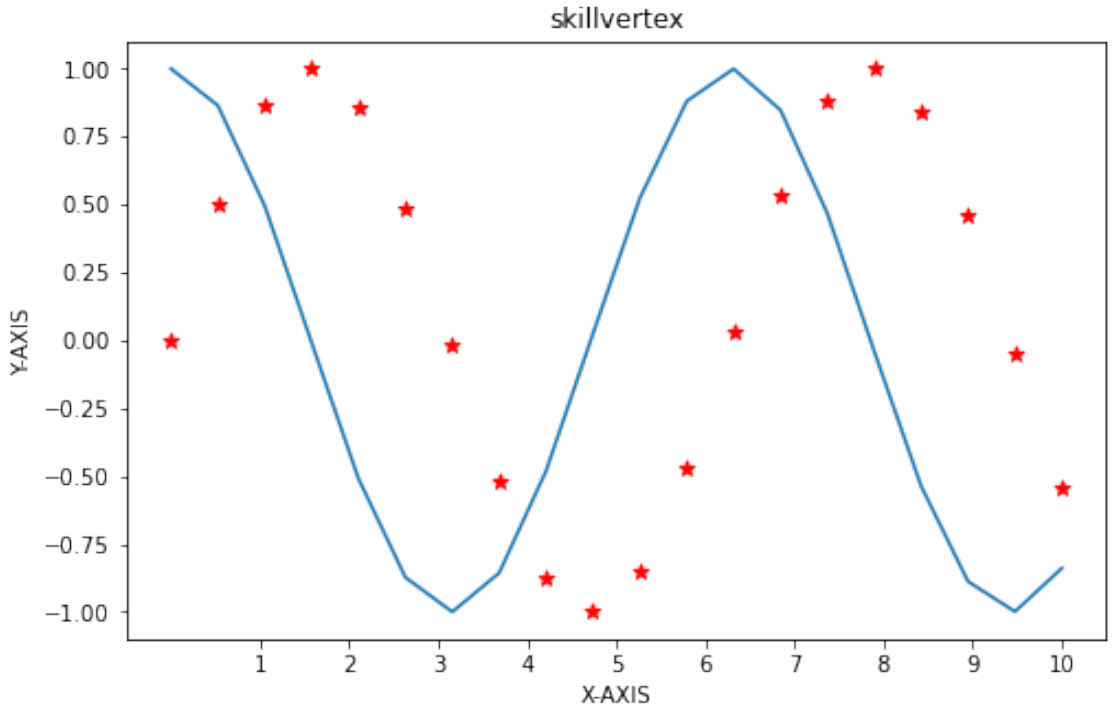
[88]:
```
#increasing the figure size
plt.figure(figsize=(8,5))
plt.scatter(x,y,c="r",marker="*",s=50)
plt.show()
#now we can see the figure of the graph is increased
```



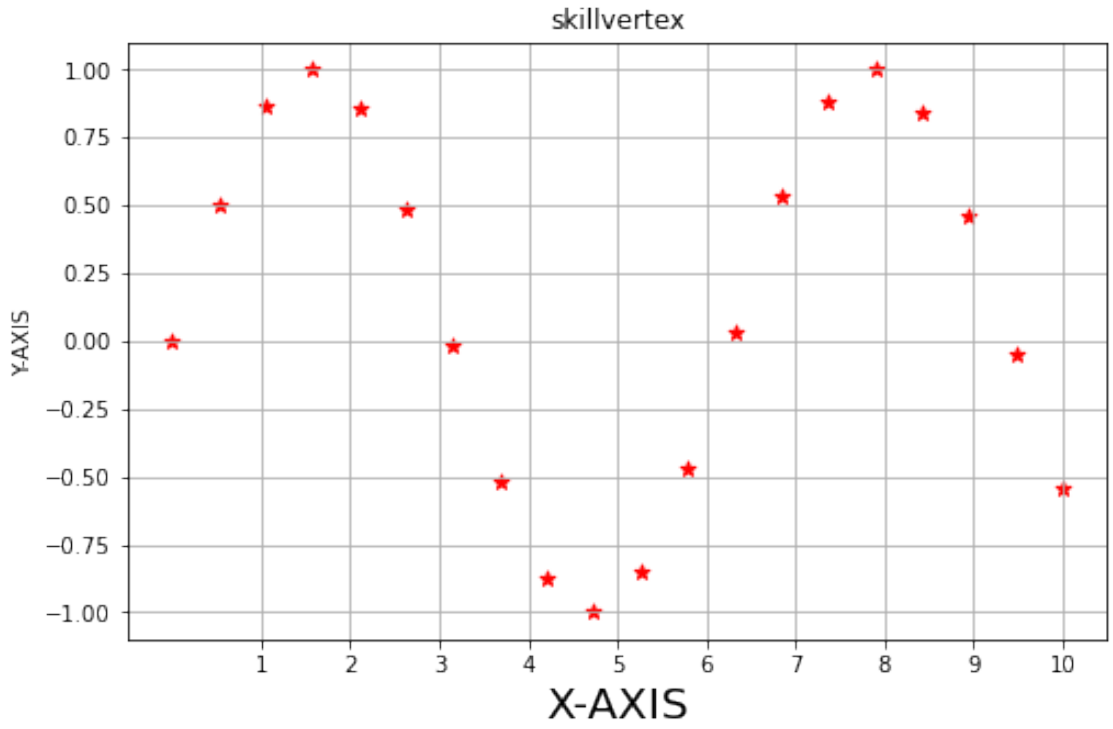
[87]:
```
#if you want the x values in the graph completely on the axis line follow the␣
 ↪below code
plt.figure(figsize=(8,5))
plt.scatter(x,y,c="r",marker="*",s=50)
plt.title("skillvertex")
plt.xlabel('X-AXIS')
plt.ylabel('Y-AXIS')
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.show()
#after getting the graph we can see the x axis values are completely given what␣
 ↪we have mentioned
```



[86]:
```
#creating a line plot in the same above graph
z=np.cos(x)
plt.figure(figsize=(8,5))
plt.scatter(x,y,c="r",marker="*",s=50)
plt.plot(x,z)
plt.title("skillvertex")
plt.xlabel('X-AXIS')
plt.ylabel('Y-AXIS')
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.show()
```



[85]:
```
#increasing the font size
#and plotting a grid over the graph
plt.figure(figsize=(8,5))
plt.scatter(x,y,c="r",marker="*",s=50)
plt.title("skillvertex")
plt.xlabel('X-AXIS',fontsize=20)
plt.ylabel('Y-AXIS')
plt.grid()
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.show()
```

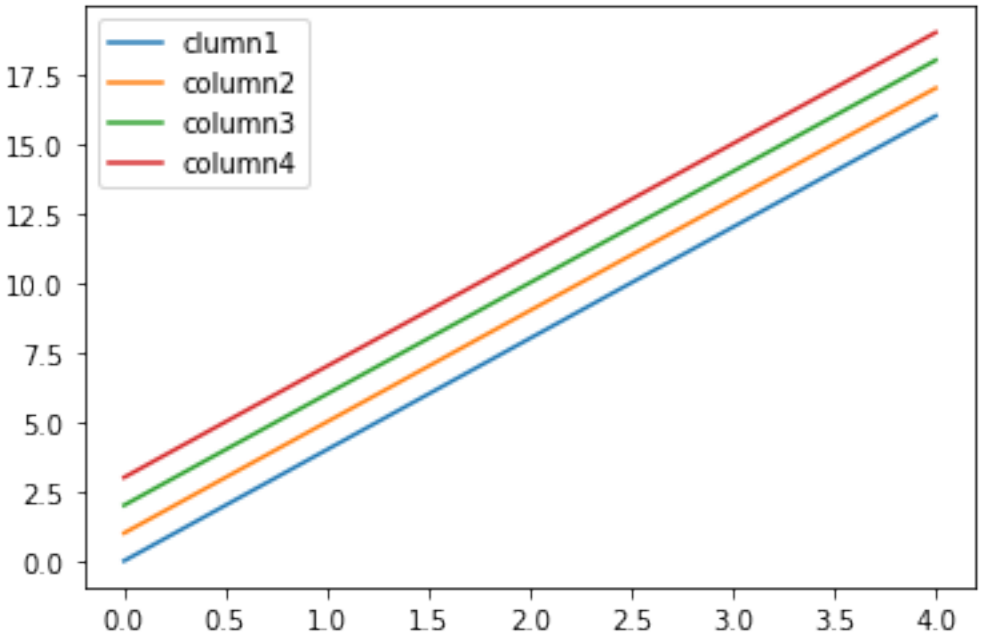


[100]:
```
import pandas as pd
import numpy as np
show_data = pd.read_csv('test1.csv')
print(show_data)
#this is the data present in the above file
```

```
  Unnamed: 0  clumn1  column2  column3  column4
0       row1       0        1        2        3
1       row2       4        5        6        7
2       row3       8        9       10       11
3       row4      12       13       14       15
4       row5      16       17       18       19
```

[101]:
```
show_data.plot()
#the above data of the file is plotted here
```

[101]: <AxesSubplot:>

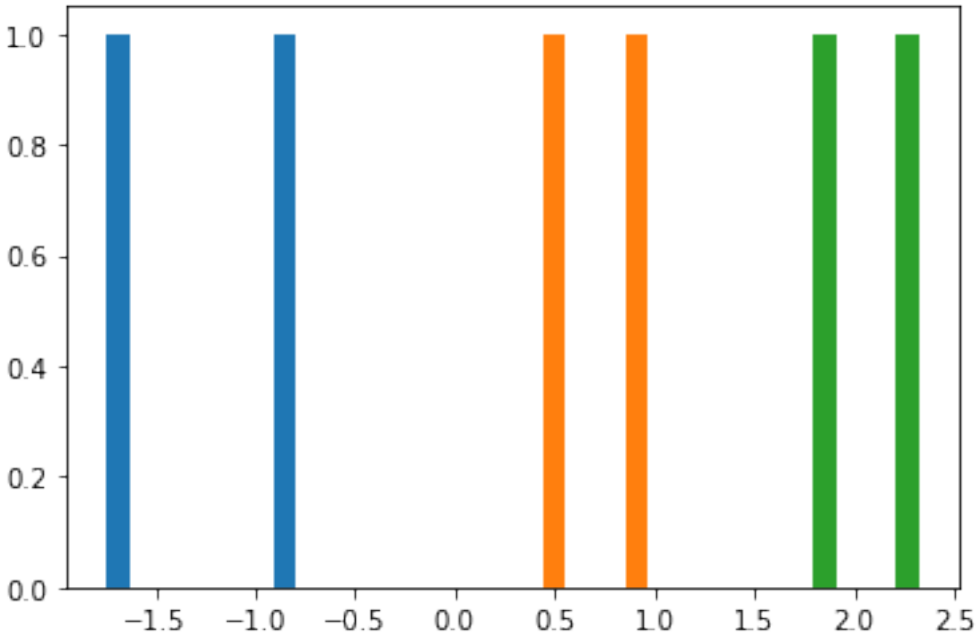


[105]:
```
#random distribution
z=np.random.normal(0,1,(2,3))
print(z)
#this means it has 100 rows and 3 columns from value 0-1
```

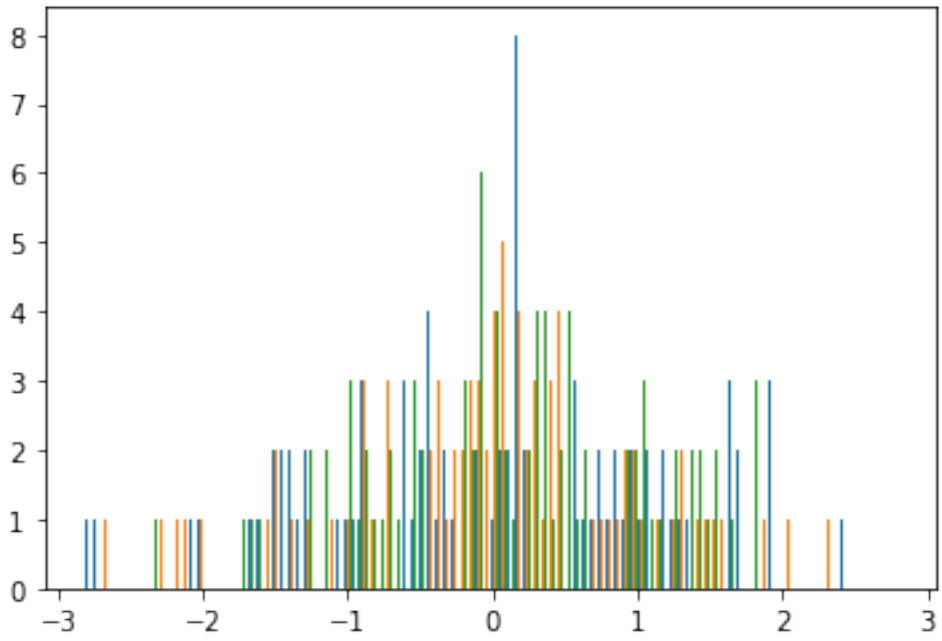
```
[[-0.87786599  0.30145131  1.67849316]
 [-1.78394043  1.01440473  2.36487805]]
```

[106]:
```
#plotting histogram from the normall distrubuted data
plt.hist(z)
```

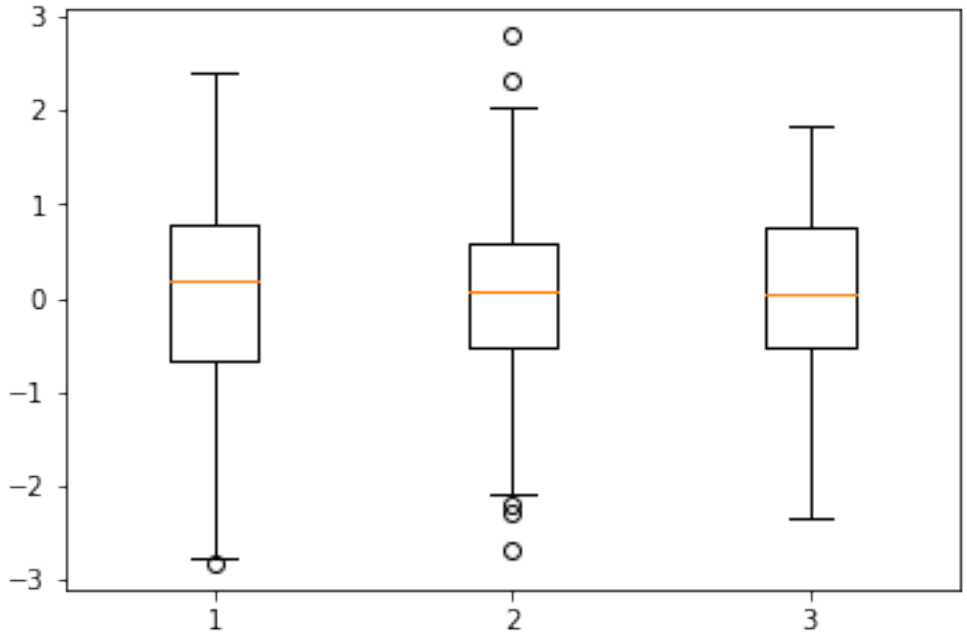
[106]:
```
(array([[1., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 1., 1., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 1., 1.]]),
 array([-1.78394043, -1.36905858, -0.95417674, -0.53929489, -0.12441304,
         0.29046881,  0.70535066,  1.1202325 ,  1.53511435,  1.9499962 ,
         2.36487805]),
 <a list of 3 BarContainer objects>)
```



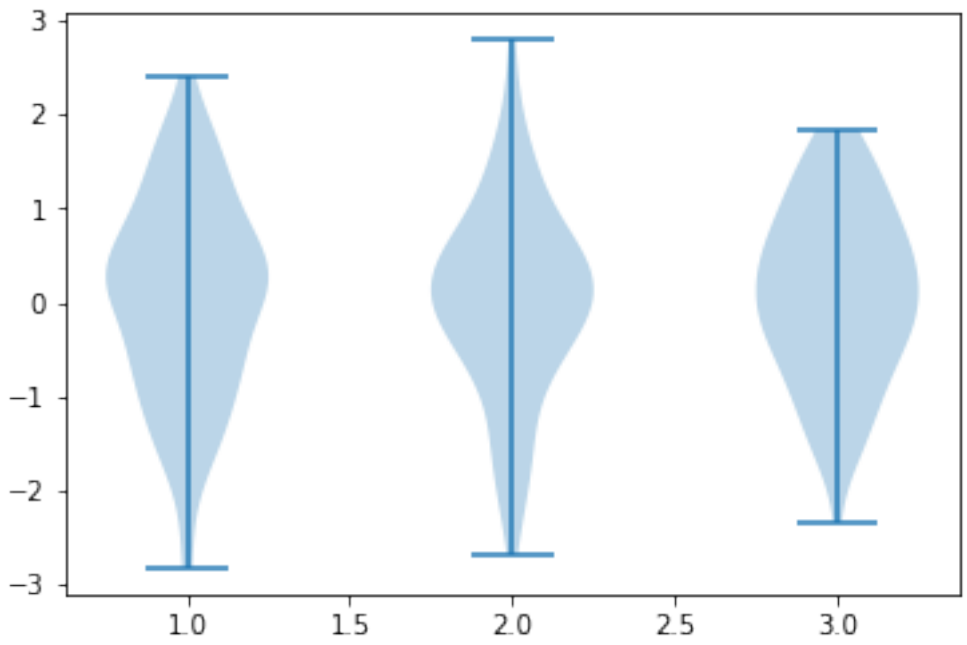
[109]:
```
#this is another example
err=np.random.normal(0,1,(100,3))
plt.hist(err,bins=100)
plt.show()
```



[110]:
```
#creating a boxplot
plt.boxplot(err)
plt.show()
#from the below graph the mav value is 3
```



[111]:
```
#to show the density of the random distribution we created above we can use␣
 ↪violin plot
plt.violinplot(err)
plt.show()
#this shows the density of the random distribution
#we can see the curve its where the density or distribution is more..
```

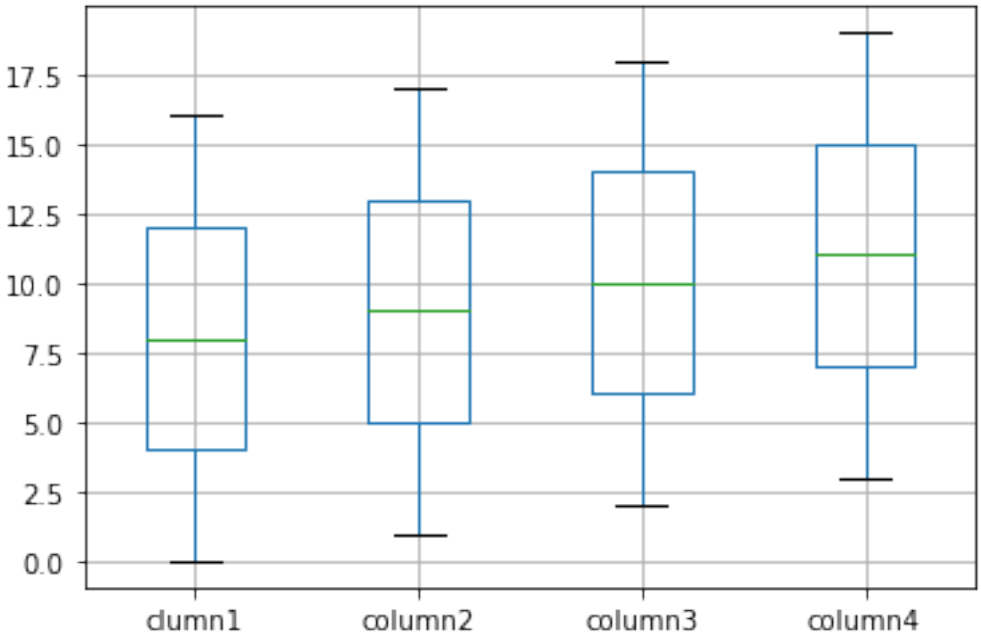


[116]:
```
#we can create boxplot using dataframe
df=pd.read_csv('test1.csv')
print(df)
```

```
  Unnamed: 0  clumn1  column2  column3  column4
0       row1       0        1        2        3
1       row2       4        5        6        7
2       row3       8        9       10       11
3       row4      12       13       14       15
4       row5      16       17       18       19
```

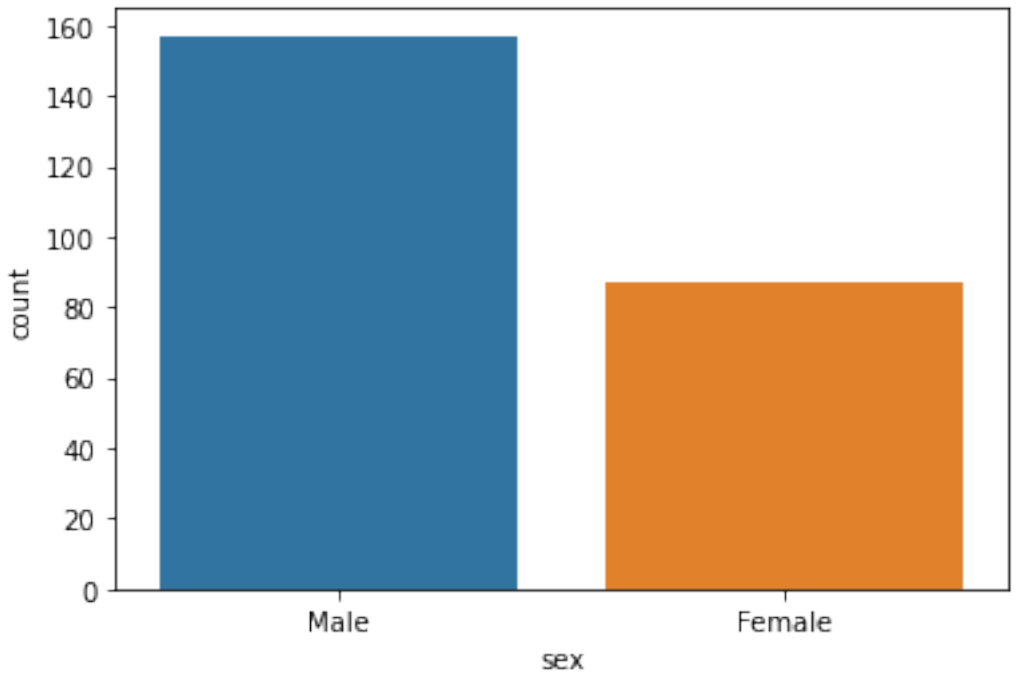
[119]:
```
df.boxplot()
```

[119]: <AxesSubplot:>

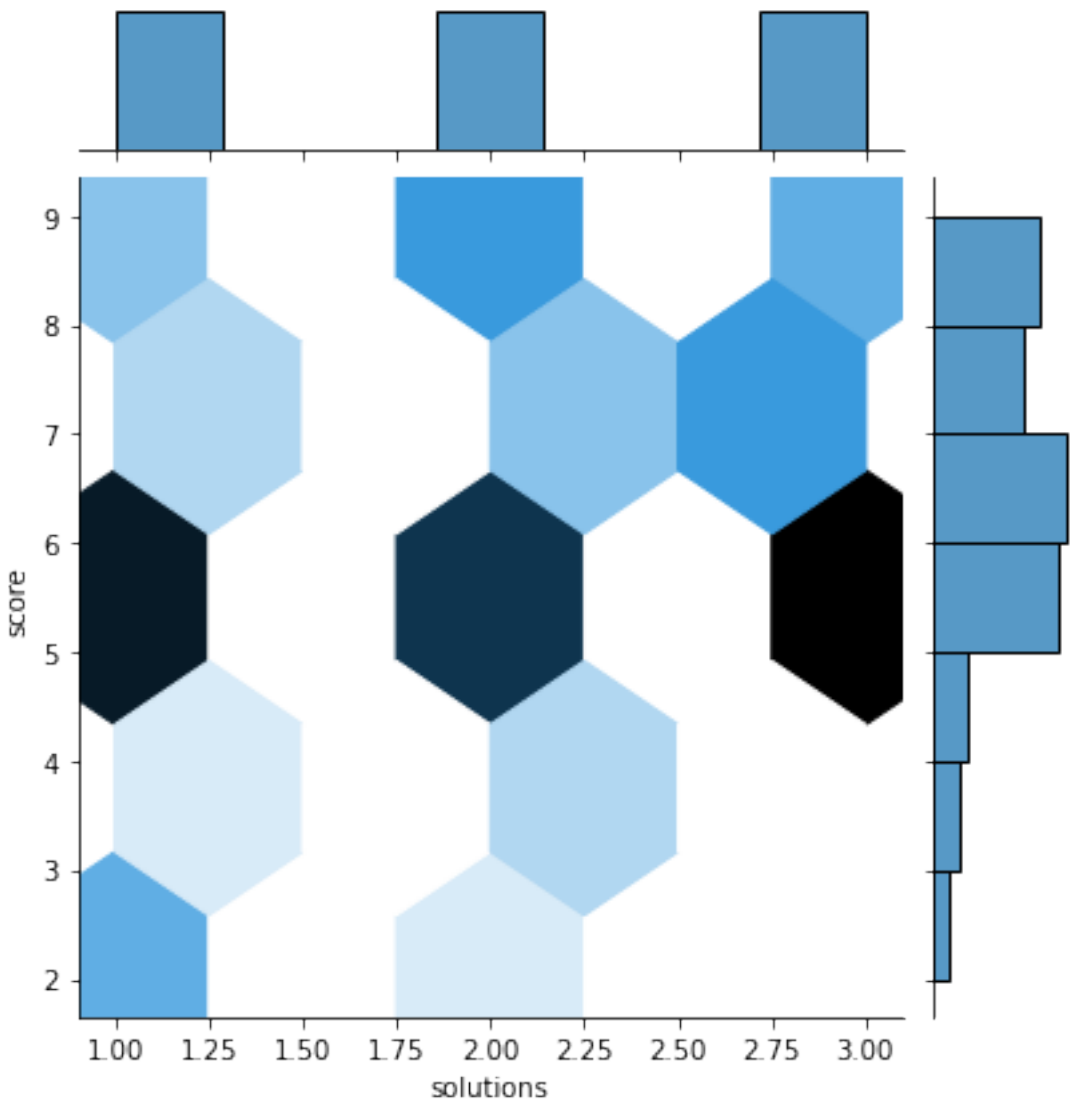


[123]:
```
#seaborn interlinked with matplotlib
import seaborn as sns
#now we are using countplot
#The countplot is used to represent the occurrence (counts) of the observation␣
 ↪present in the categorical variable. It uses the concept of a bar chart for␣
 ↪the visual depiction.
```

[128]:
```
# importing the required library

import seaborn as sns
import matplotlib.pyplot as plt

# read a tips.csv file from seaborn library
df = sns.load_dataset('tips')

# count plot on single categorical variable
sns.countplot(x ='sex', data = df)

# Show the plot
plt.show()
#from this below graph we cane see more than 150 males are there and less than␣
 ↪100 females
```

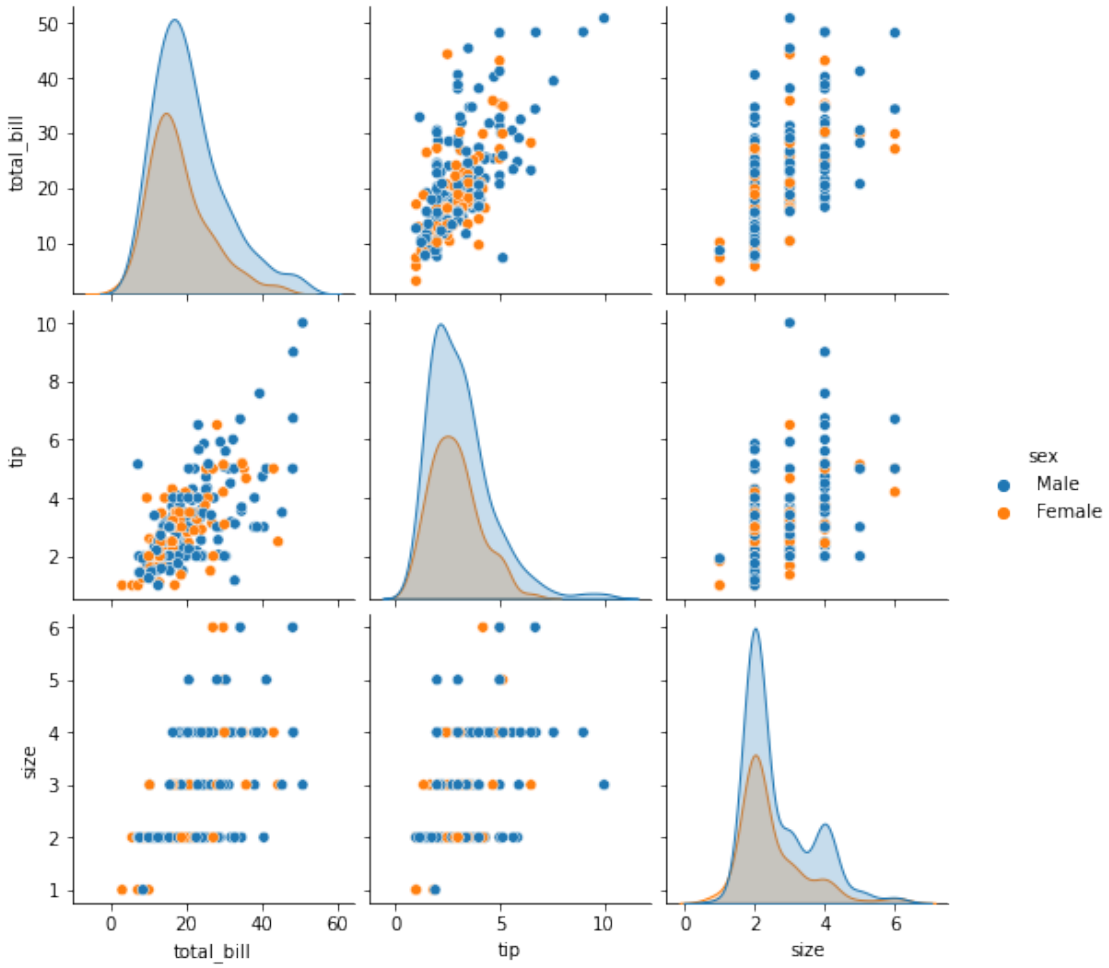


[129]:
```
#using jointplot
# importing required packages
import seaborn as sns
import matplotlib.pyplot as plt

# loading dataset
data = sns.load_dataset("attention")

# draw jointplot with
# hex kind
sns.jointplot(x = "solutions", y = "score",
                        kind = "hex", data = data)
# show the plot
plt.show()

# This code is contributed
# by Deepanshu Rustagi.
```



[130]:
```
#using pairplot
# importing packages
import seaborn
import matplotlib.pyplot as plt

############## Main Section ############
# loading dataset using seaborn
df = seaborn.load_dataset('tips')
# pairplot with hue sex
seaborn.pairplot(df, hue ='sex')
# to show
plt.show()
```

```
# This code is contributed by Deepanshu Rustagi.
```



[ ]: