Matplotlib: used for data visualization. matplotlib works on 2-D numpy arrays. it is basically used to draw or plot graphs, scatter plot,bar charts, histograms, legend title style plots. Data visualization: it is a graphical represntation of information and data. in form of charts, maps and graphs

for using matplotlib we must install it first using pip command

In [1]:
```
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib
\site-packages (3.3.2)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\ana
conda3\lib\site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: certifi>=2020.06.20 in c:\programdata\anac
onda3\lib\site-packages (from matplotlib) (2020.6.20)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 i
n c:\programdata\anaconda3\lib\site-packages (from matplotlib) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\l
ib\site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3
\lib\site-packages (from matplotlib) (8.0.1)
Requirement already satisfied: numpy>=1.15 in c:\programdata\anaconda3\li
b\site-packages (from matplotlib) (1.19.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anacon
da3\lib\site-packages (from matplotlib) (1.3.0)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\s
ite-packages (from python-dateutil>=2.1->matplotlib) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

we have import the library package before using it

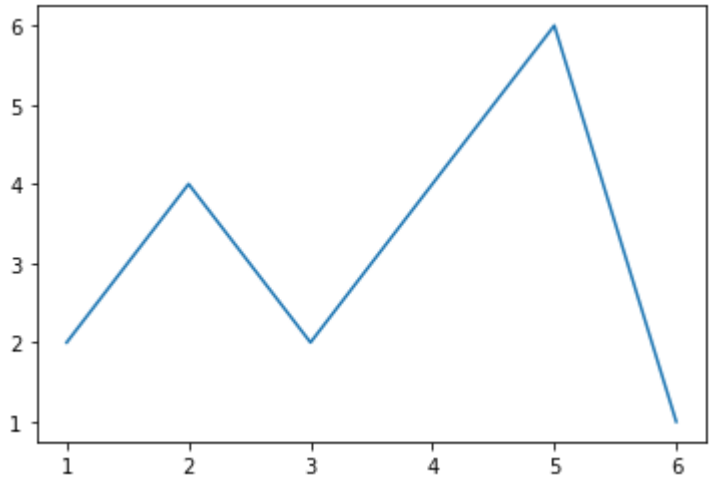
from matplotlib import pyplot as plt import matplotlib.pyplot as plt matplotlib.pyplot is basically an interface which is used to add style functions to the graphs craeted using matplotlib

our discussion will focus on: Line plot, mathematical plots,Scatter Plot, histograms, Bar chart, pie chart

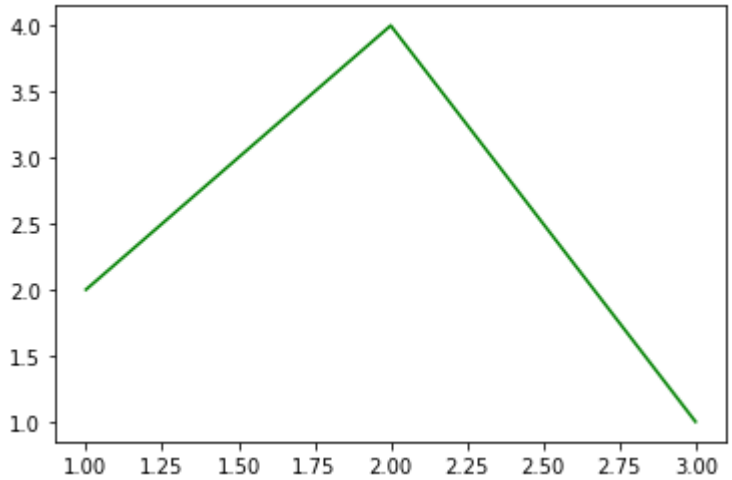
In [1]:
```
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
```

In [ ]:
```
#Line plot
```

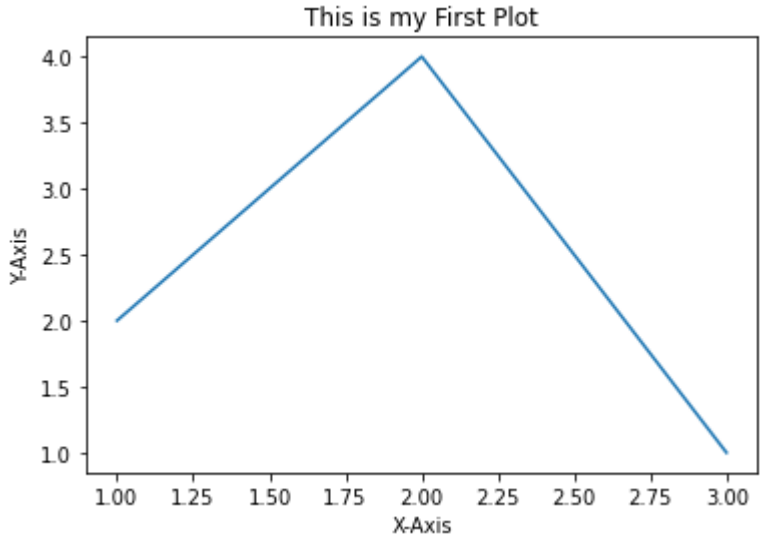
In [4]:
```
#plot of x vs y as line
x = [1,2,3,5,6]
y=[2,4,2,6,1]
plt.plot(x,y)#plot is a function used to draw
             #a 2-D plot or graph using values on x-axis and y-axis
plt.show() # to display the last plot drwan by plot() function.
```



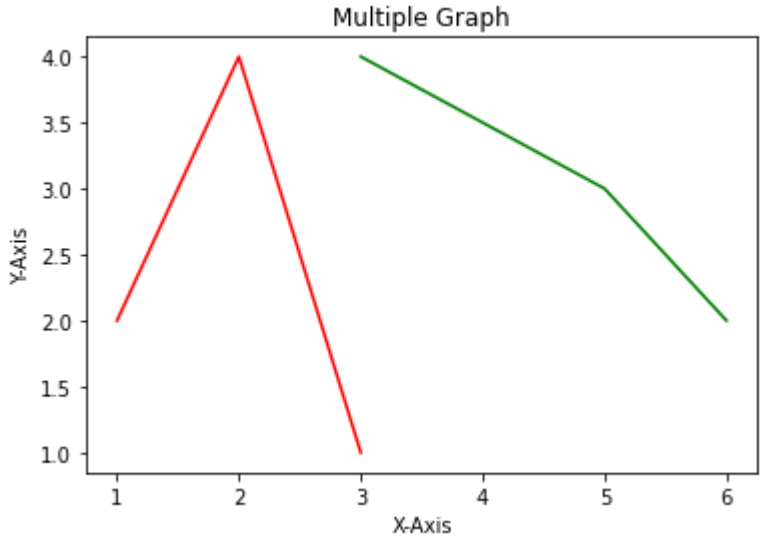

In [7]:
```
#plot of x vs y as line
x=[1,2,3]
y=[2,4,1]
plt.plot(x,y,color = 'GREEN')# color parameter is used to color the plot
plt.show()
```



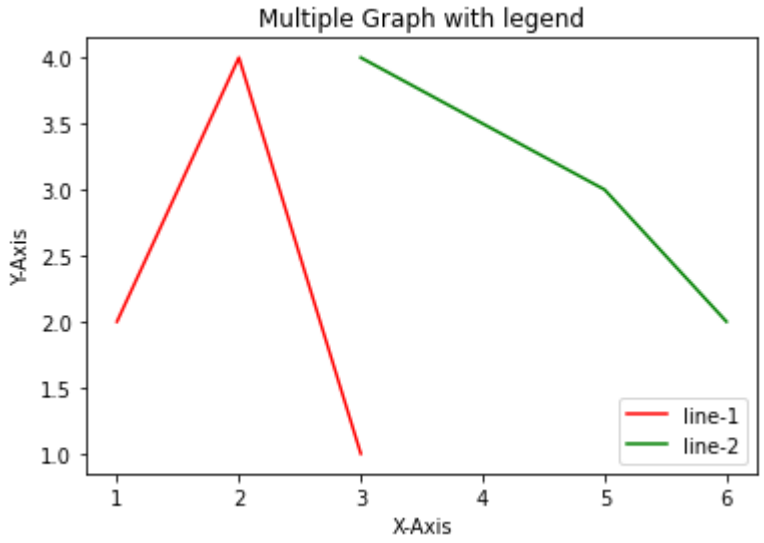
In [8]:
```
#Adding title to the plot and label to the axis.
x=[1,2,3]
y=[2,4,1]
#adding label to axis
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
#adding Title to the plot
plt.title("This is my First Plot")
plt.plot(x,y)#plot is a function used to draw
             #a 2-D plot or graph using values on x-axis and y-axis
plt.show() # to display the last plot drwan by plot() function.
```



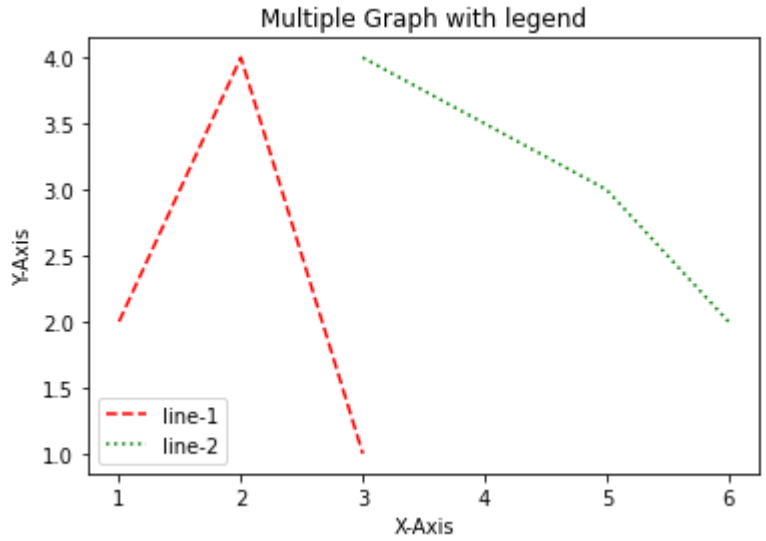
In [9]:
```
#Multiple graph in one plot
x=[1,2,3]
y=[2,4,1]
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("Multiple Graph")
plt.plot(x,y,color='red')
x1=[3,5,6]
y1=[4,3,2]
plt.plot(x1,y1,color='green')
plt.show()#it will display multiple graph
```



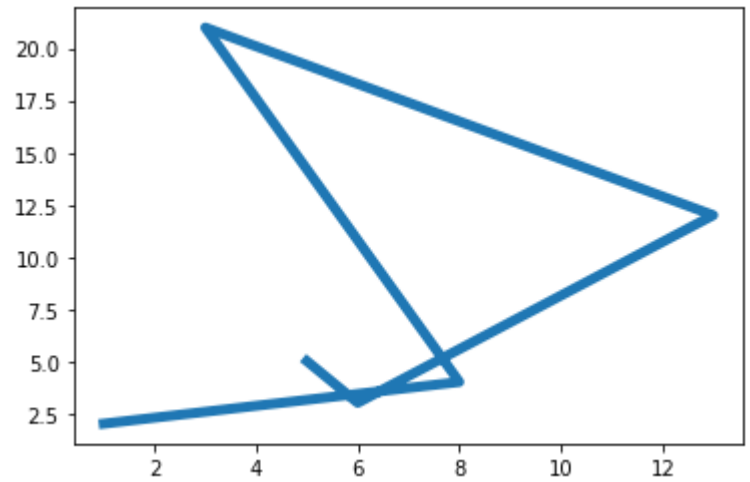
In [15]:
```
#Displaying legend in the plot
x=[1,2,3]
y=[2,4,1]
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("Multiple Graph with legend")
plt.plot(x,y,color='red',label='line-1')#label parameter is used to define
                                        #name of plot in legend
x1=[3,5,6]
y1=[4,3,2]
plt.plot(x1,y1,color='green',label='line-2')
plt.legend(loc=4)#legend function is used to display legends in the graph
                 #loc value will decide the place where legend will be disp
plt.show()
```



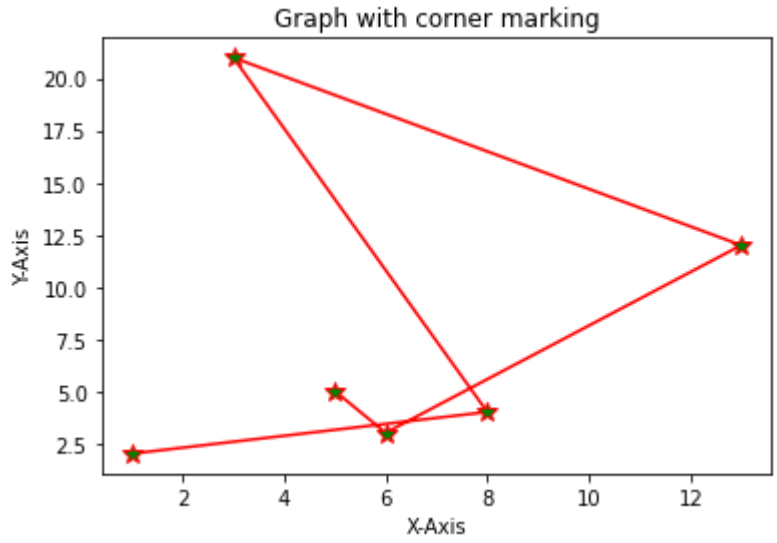
In [17]:
```
#Displaying legend in the plot
x=[1,2,3]
y=[2,4,1]
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("Multiple Graph with legend")
#'-' solid line(default)
#'--' dashed line
#'-.' dashed dot line
#':' dotted line
plt.plot(x,y,color='red',linestyle='--',label='line-1')
x1=[3,5,6]
y1=[4,3,2]
plt.plot(x1,y1,color='green',linestyle=':',label='line-2')
plt.legend(loc=3)
plt.show()
```




In [3]:
```
#graph which intersects itself
x=[1,8,3,13,6,5]
y=[2,4,21,12,3,5]
plt.plot(x,y,linewidth=5)#to linewidth is used to decide the width of plote
plt.show()
```



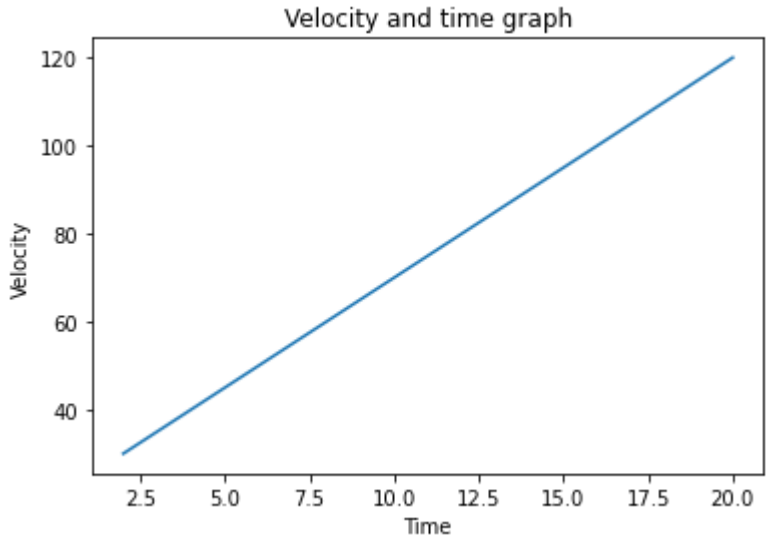
In [18]:
```
#highlighting corners of a plot using marker parameter
x=[1,8,3,13,6,5]
y=[2,4,21,12,3,5]
plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("Graph with corner marking")
plt.plot(x,y,color='red',marker='*',markersize=10,markerfacecolor='green')
plt.show()
```



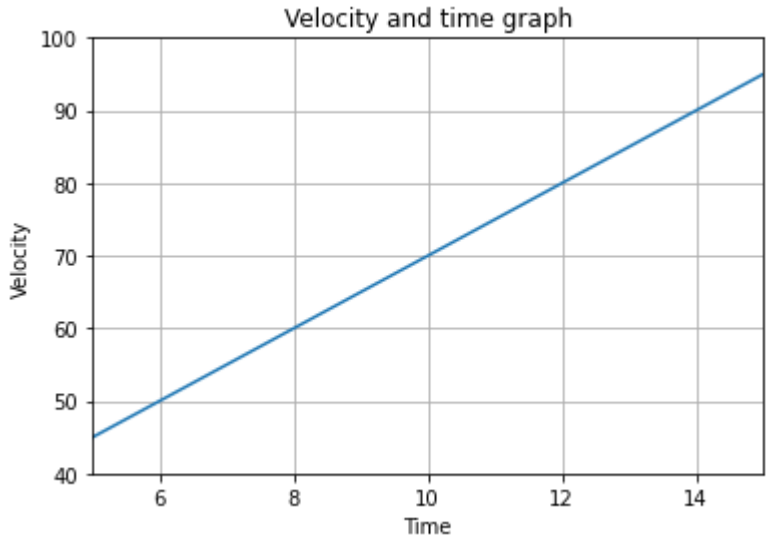

In [11]:
```
#plotting a velocity graph by using v=u+a*t
#calculation
u=20
a=5
t=np.arange(2,21)
v=[]
for i in t:
    v.append(u+a*i)
print(t)
print(v)
```

```
[ 2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20]
[30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 1
15, 120]
```

In [12]:
```
#plotting the graph
plt.plot(t,v)
plt.xlabel('Time')
plt.ylabel('Velocity')
plt.title('Velocity and time graph')
plt.show()
```




In [18]:
```
#plotting the graph
plt.plot(t,v)
plt.xlim(5,15)# is used to clip(limit) the x axis
plt.ylim(40,100)#is used to clip(limit) the y axis
plt.xlabel('Time')
plt.ylabel('Velocity')
plt.title('Velocity and time graph')
plt.grid()
plt.show()
```

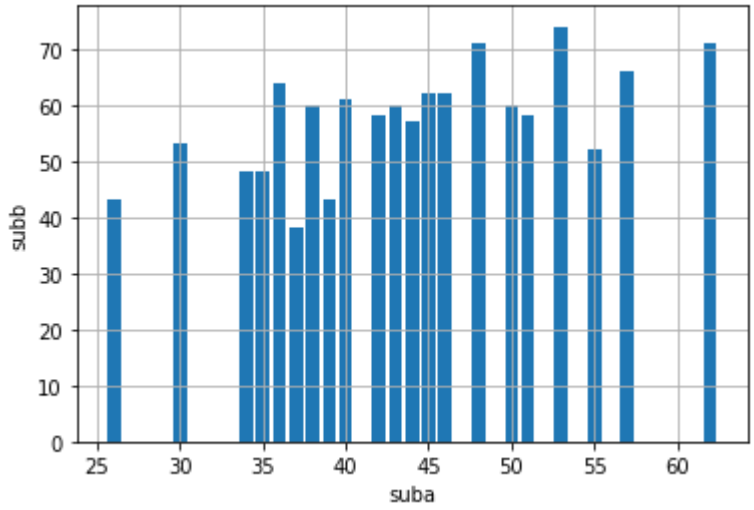



#Bar Plot: it is a rectagular plot which performs #comparison between 2 or more identities

In [7]:
```
data=pd.read_csv(r'C:\desktop\result.csv')
print(data) # print all data
#data.head() #print first five rows
#data.tail() # show last five rows
```

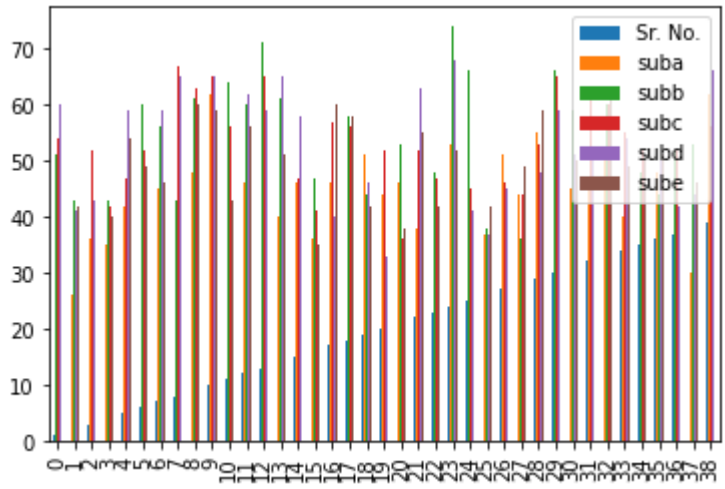
| | Sr. No. | Name of Student | suba | subb | subc | subd | sube |
|---|---|---|---|---|---|---|---|
| 0 | 1 | ABHISHEK KUMAR KUMAR | 42 | 51 | 54 | 60 | 53 |
| 1 | 2 | AINUL | 26 | 43 | 42 | 41 | 42 |
| 2 | 3 | AJAY KUMAR | 36 | 48 | 52 | 43 | 51 |
| 3 | 4 | ALOK MOHAN | 35 | 43 | 42 | 32 | 40 |
| 4 | 5 | ANAND KUMAR | 42 | 58 | 47 | 59 | 54 |
| 5 | 6 | ANKIT RAJPUT | 43 | 60 | 52 | 53 | 49 |
| 6 | 7 | ASHISH KUMAR | 45 | 56 | 50 | 59 | 46 |
| 7 | 8 | ASHISH TOMAR | 39 | 43 | 67 | 65 | 37 |
| 8 | 9 | DHARMESH SAINI | 48 | 61 | 63 | 56 | 60 |
| 9 | 10 | DIKSHA SINGH | 62 | 71 | 65 | 65 | 59 |
| 10 | 11 | DILEEP KUMAR CHATURVEDI | 36 | 64 | 56 | 56 | 43 |
| 11 | 12 | GAURAV SHARMA | 46 | 60 | 55 | 62 | 56 |
| 12 | 13 | HAMZA NAEEM | 48 | 71 | 65 | 59 | 56 |
| 13 | 14 | HASIM ALI | 40 | 61 | 60 | 65 | 51 |
| 14 | 15 | HUNAR VAISH | 46 | 53 | 47 | 58 | 54 |
| 15 | 16 | KRISHNA KUMAR YADAV | 36 | 47 | 41 | 48 | 35 |
| 16 | 17 | MANINDRA KR KANNAUJIYA | 46 | 62 | 57 | 40 | 60 |
| 17 | 18 | NEHA DESHWAL | 51 | 58 | 56 | 62 | 58 |
| 18 | 19 | NITIN | 51 | 44 | 44 | 46 | 42 |
| 19 | 20 | NITIN KUMAR | 44 | 57 | 52 | 33 | 40 |
| 20 | 21 | NITIN MALIK | 46 | 53 | 36 | 38 | 38 |
| 21 | 22 | PUNEET KUMAR | 38 | 60 | 52 | 63 | 55 |
| 22 | 23 | PUSHPENDRA SINGH | 34 | 48 | 47 | 51 | 42 |
| 23 | 24 | RAHUL KUMAR | 53 | 74 | 63 | 68 | 52 |
| 24 | 25 | RAXIT KUMAR | 53 | 66 | 45 | 41 | 57 |
| 25 | 26 | SACHIN KUMAR SISODIA | 37 | 38 | 37 | 37 | 42 |
| 26 | 27 | SANJEEV KUMAR | 51 | 42 | 46 | 45 | 44 |
| 27 | 28 | SAURABH KUMAR | 44 | 36 | 44 | 42 | 49 |
| 28 | 29 | SOHIT SHARMA | 55 | 52 | 53 | 48 | 59 |
| 29 | 30 | VIJAY PAL | 57 | 66 | 65 | 59 | 58 |
| 30 | 31 | YASHOBHARDHAN DWIVEDI | 45 | 59 | 57 | 51 | 50 |
| 31 | 32 | ABHISHEK KUMAR | 45 | 62 | 63 | 52 | 61 |
| 32 | 33 | ROHIT GAUR | 50 | 60 | 60 | 47 | 62 |
| 33 | 34 | ABHISHEK SHANKAR MISHRA | 40 | 50 | 55 | 54 | 49 |
| 34 | 35 | ARPIT GUPTA | 35 | 48 | 52 | 59 | 50 |
| 35 | 36 | AMIT KUMAR MISHRA | 48 | 44 | 59 | 50 | 51 |
| 36 | 37 | Ashutosh MISHRA | 42 | 52 | 52 | 42 | 46 |
| 37 | 38 | MD. Niyaz(DIP) | 30 | 53 | 37 | 44 | 46 |
| 38 | 39 | Shekhar Mittal(DIP) | 62 | 64 | 56 | 66 | 71 |

In [14]:
```
plt.bar(data['suba'],data['subb'])#it is a function of matplotlib
plt.xlabel('suba')
plt.ylabel('subb')
plt.grid()#it creates grids corresponding to the x and y axis
plt.show()
```



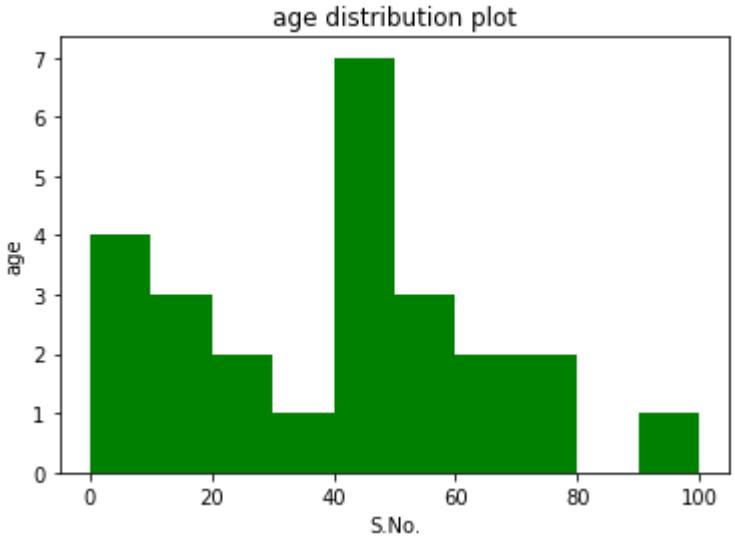

In [17]:
```
data.plot.bar()#by using plot function of DF
plt.legend(loc=1)
```

Out[17]: <matplotlib.legend.Legend at 0x1a265920df0>



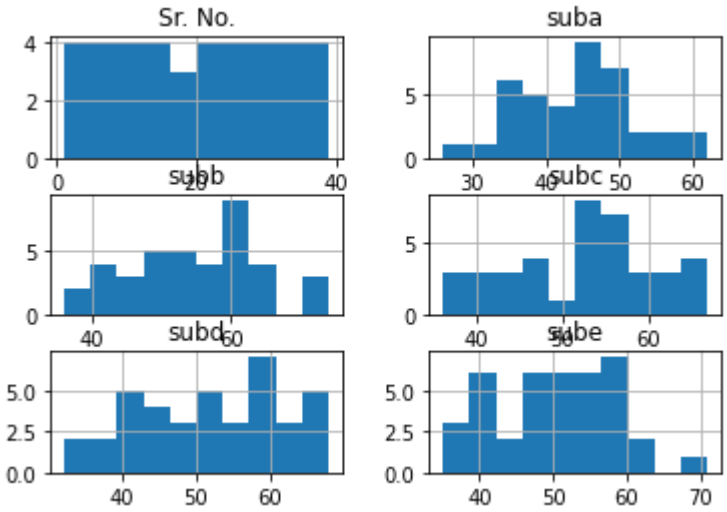

Histograms: it is used for frequency distribution.

In [9]:
```
#Histogram using matplotlib function
age=[2,5,70,40,3,55,45,50,45,43,40,44,60,7,13,57,18,11,90,77,32,21,20,40,67
range=(0,100)
inter=10
#hist(): function to plot histogram
plt.hist(age,inter,range,color='green')
plt.xlabel('S.No.')
plt.ylabel('age')
plt.title('age distribution plot')
plt.show()
```




In [21]:
```
data.hist()#using hist function of DF
```

Out[21]:
```
array([[<AxesSubplot:title={'center':'Sr. No.'}>,
        <AxesSubplot:title={'center':'suba'}>],
       [<AxesSubplot:title={'center':'subb'}>,
        <AxesSubplot:title={'center':'subc'}>],
       [<AxesSubplot:title={'center':'subd'}>,
        <AxesSubplot:title={'center':'sube'}>]], dtype=object)
```



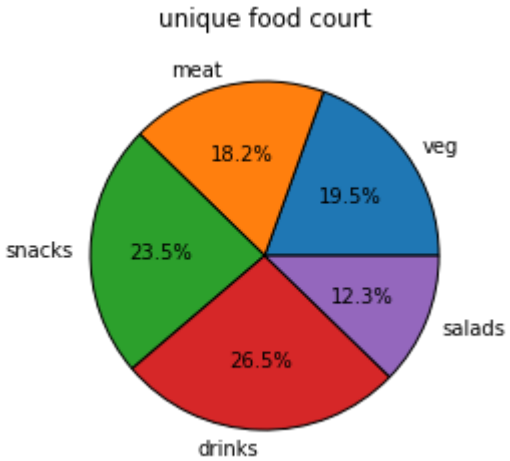
In [10]:
```
plt.pie(age,wedgeprops={'edgecolor':'black'},autopct='%1.1f%%')
```

Out[10]:
```
([<matplotlib.patches.Wedge at 0x2295035a6b0>,
  <matplotlib.patches.Wedge at 0x2295035ace0>,
  <matplotlib.patches.Wedge at 0x2295035b340>,
  <matplotlib.patches.Wedge at 0x2295035ba60>,
  <matplotlib.patches.Wedge at 0x229503981c0>,
  <matplotlib.patches.Wedge at 0x229503988e0>,
  <matplotlib.patches.Wedge at 0x22950399000>,
  <matplotlib.patches.Wedge at 0x22950399720>,
  <matplotlib.patches.Wedge at 0x22950399e40>,
  <matplotlib.patches.Wedge at 0x2295039a560>,
  <matplotlib.patches.Wedge at 0x2295035a680>,
  <matplotlib.patches.Wedge at 0x2295039b370>,
  <matplotlib.patches.Wedge at 0x2295039ba90>,
  <matplotlib.patches.Wedge at 0x229503d01f0>,
  <matplotlib.patches.Wedge at 0x229503d0910>,
  <matplotlib.patches.Wedge at 0x229503d1030>,
  <matplotlib.patches.Wedge at 0x229503d1750>,
  <matplotlib.patches.Wedge at 0x229503d1e70>,
  <matplotlib.patches.Wedge at 0x229503d2590>,
```

pie chart or pie plot: it is used to show percentage of different partcipating entities in a whole

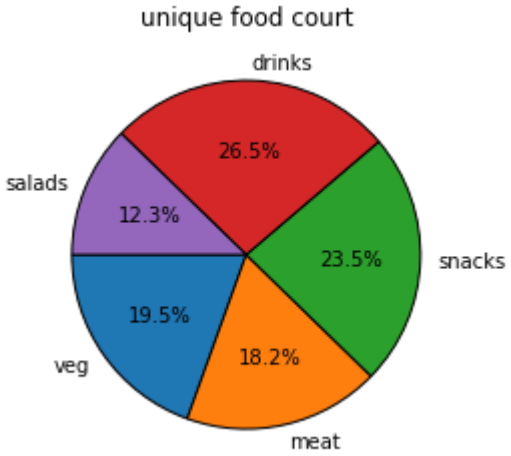
In [31]:
```
#piechart: pie() function is used
revenue=[7267,6785,8765,9876,4567]
item=['veg','meat','snacks','drinks','salads']
plt.pie(revenue,labels=item,wedgeprops={'edgecolor':'black'},autopct='%1.1f
plt.title('unique food court')
```

Out[31]:
```
Text(0.5, 1.0, 'unique food court')
```



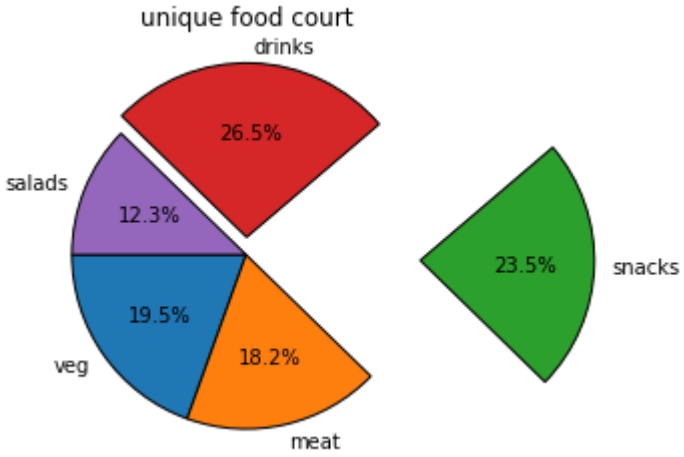
In [32]:
```
#piechart: pie() function is used
revenue=[7267,6785,8765,9876,4567]
item=['veg','meat','snacks','drinks','salads']
plt.pie(revenue,labels=item,wedgeprops={'edgecolor':'black'},autopct='%1.1
plt.title('unique food court')
```

Out[32]: Text(0.5, 1.0, 'unique food court')



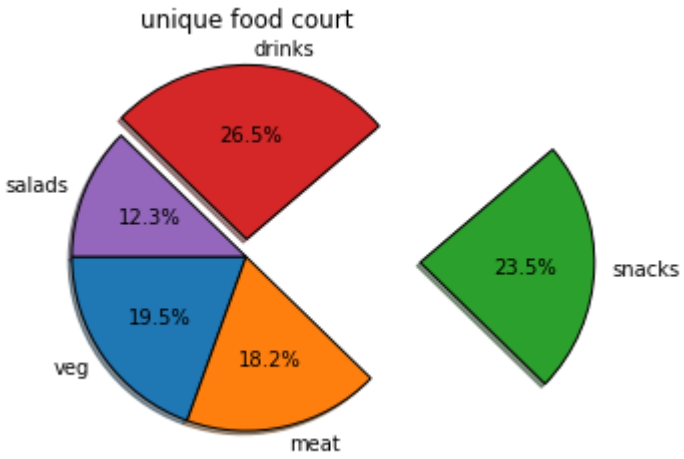

In [34]:
```
#piechart: pie() function is used
revenue=[7267,6785,8765,9876,4567]
ex=[0,0,1.0,0.1,0]
item=['veg','meat','snacks','drinks','salads']
plt.pie(revenue,labels=item,wedgeprops={'edgecolor':'black'},explode=ex,
        autopct='%1.1f%%',startangle=180)
plt.title('unique food court')
```

Out[34]: Text(0.5, 1.0, 'unique food court')



In [35]:
```
#piechart: pie() function is used
revenue=[7267,6785,8765,9876,4567]
ex=[0,0,1.0,0.1,0]
item=['veg','meat','snacks','drinks','salads']
plt.pie(revenue,labels=item,wedgeprops={'edgecolor':'black'},explode=ex,
        shadow=True,autopct='%1.1f%%',startangle=180)
plt.title('unique food court')
```

Out[35]: Text(0.5, 1.0, 'unique food court')



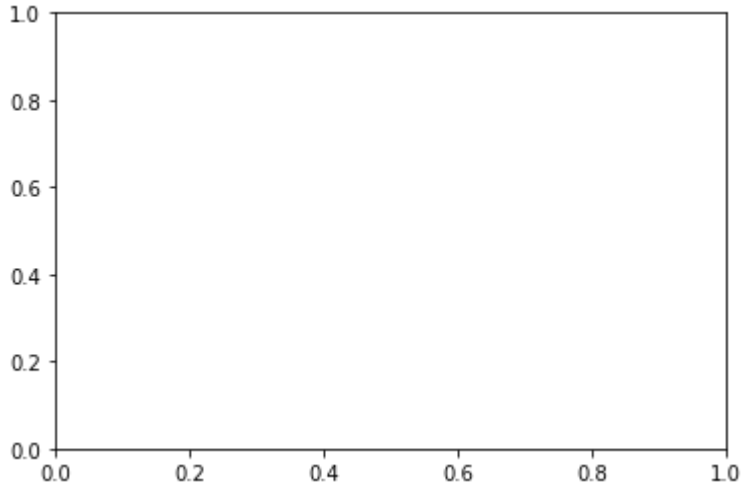

Stacked or Area plot: used to compare two areas of given identities.

In [10]:

```
#stacked plot
months=['jan','feb','mar','apr','may','jun','jul','aug','sep','oct','nov',
sales_car=[120,30,56,89,90,120,34,56,78,12,45,56]
sales_bike=[240,27,34,56,98,110,150,34,10,23,72,78]
plt.bar(sales_car,sales_bike,labels=['car','bike'])
plt.legend()
plt.title("car vs bike sale")
plt.show()
```

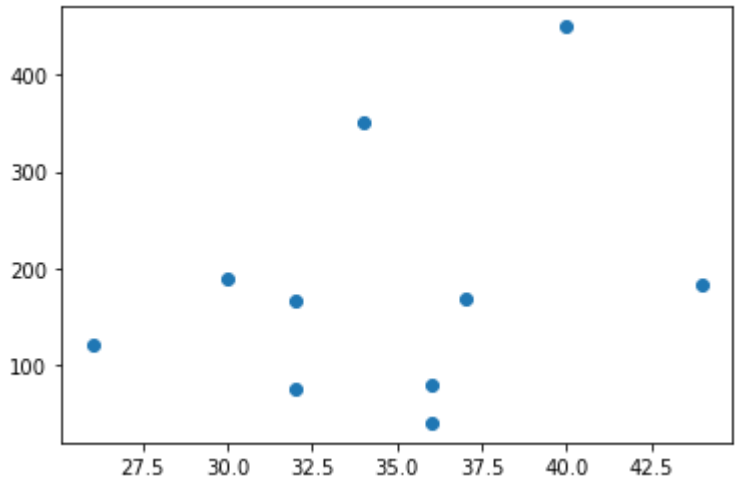
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [10], in <module>
      3 sales_car=[120,30,56,89,90,120,34,56,78,12,45,56]
      4 sales_bike=[240,27,34,56,98,110,150,34,10,23,72,78]
----> 5 plt.bar(sales_car,sales_bike,labels=['car','bike'])
      6 plt.legend()
      7 plt.title("car vs bike sale")

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\pyplot.py:2399, in bar(x, height, width, bottom, align, data, **kwargs)
   2395 @_copy_docstring_and_deprecators(Axes.bar)
   2396 def bar(
   2397         x, height, width=0.8, bottom=None, *, align='center',
   2398         data=None, **kwargs):
-> 2399     return gca().bar(
   2400         x, height, width=width, bottom=bottom, align=align,
   2401         **({"data": data} if data is not None else {}), **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\__init__.py:1412, in _preprocess_data.<locals>.inner(ax, data, *args, **kwargs)
   1409 @functools.wraps(func)
   1410 def inner(ax, *args, data=None, **kwargs):
   1411     if data is None:
-> 1412         return func(ax, *map(sanitize_sequence, args), **kwargs)
   1414     bound = new_sig.bind(ax, *args, **kwargs)
   1415     auto_label = (bound.arguments.get(label_namer)
   1416                   or bound.kwargs.get(label_namer))

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\axes\_axes.py:2403, in Axes.bar(self, x, height, width, bottom, align, **kwargs)
   2394 for l, b, w, h, c, e, lw, htch in args:
   2395     r = mpatches.Rectangle(
   2396         xy=(l, b), width=w, height=h,
   2397         facecolor=c,
   (...)
   2401         hatch=htch,
   2402         )
-> 2403     r.update(kwargs)
   2404     r.get_path()._interpolation_steps = 100
   2405     if orientation == 'vertical':

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\artist.py:1064, in Artist.update(self, props)
   1062             func = getattr(self, f"set_{k}", None)
   1063             if not callable(func):
-> 1064                 raise AttributeError(f"{type(self).__name__!r} object "
   1065                                      f"has no property {k!r}")
   1066             ret.append(func(v))
   1067 if ret:
```

```
AttributeError: 'Rectangle' object has no property 'labels'
```

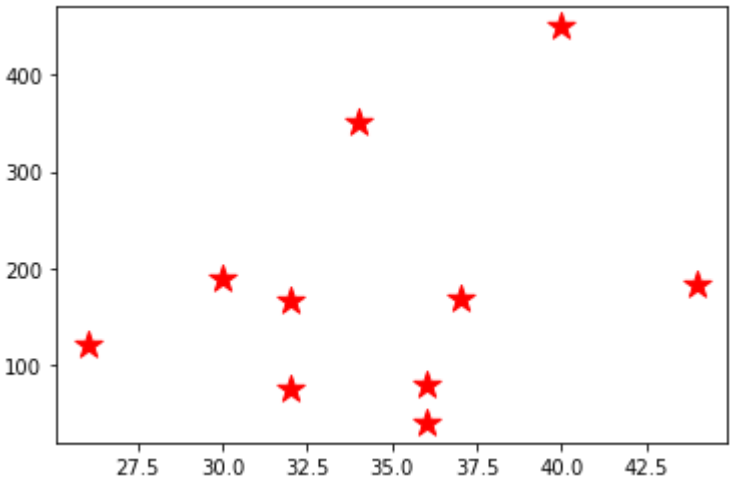



In [ ]:
```
#Scatter plot : used in machine learning for classification and clustering
```

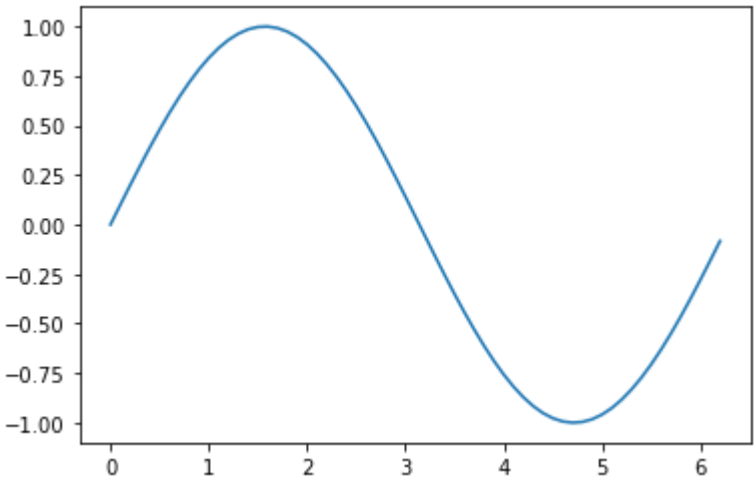
In [37]:
```
plt.scatter(data['Age'],data['Income'])
plt.show()
```

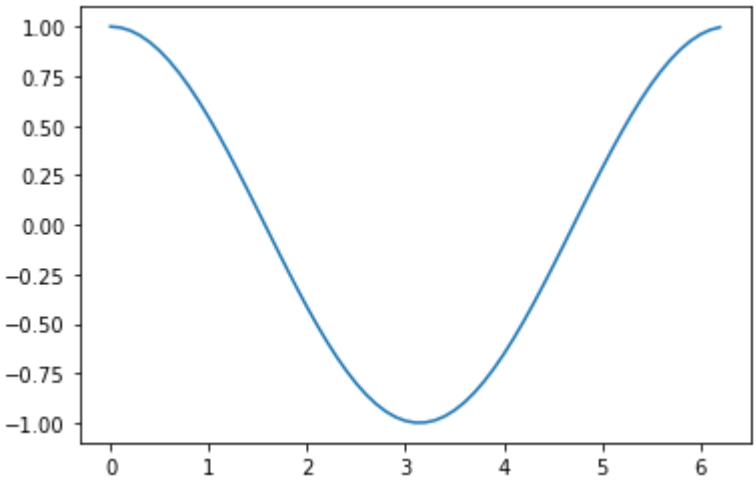


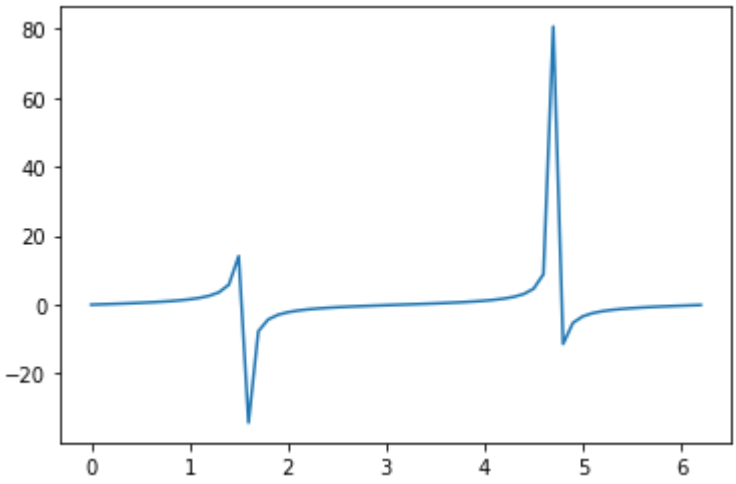
In [38]:
```
plt.scatter(data['Age'],data['Income'],color='red',marker='*',s=200)
plt.show()
```



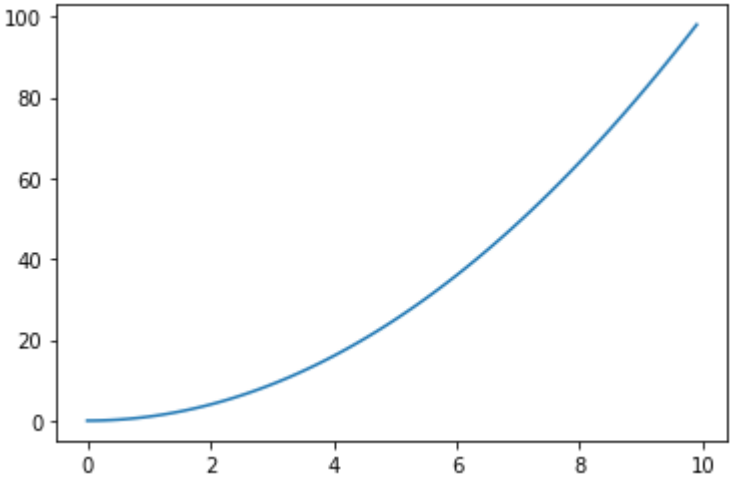
In [43]:
```
#ploting curves of given equation:
x=np.arange(0,2*(np.pi),0.1)#np.pi is value pi in radian
y=np.sin(x)
plt.plot(x,y)
plt.show()
y=np.cos(x)
plt.plot(x,y)
plt.show()
y=np.tan(x)
plt.plot(x,y)
plt.show()
```



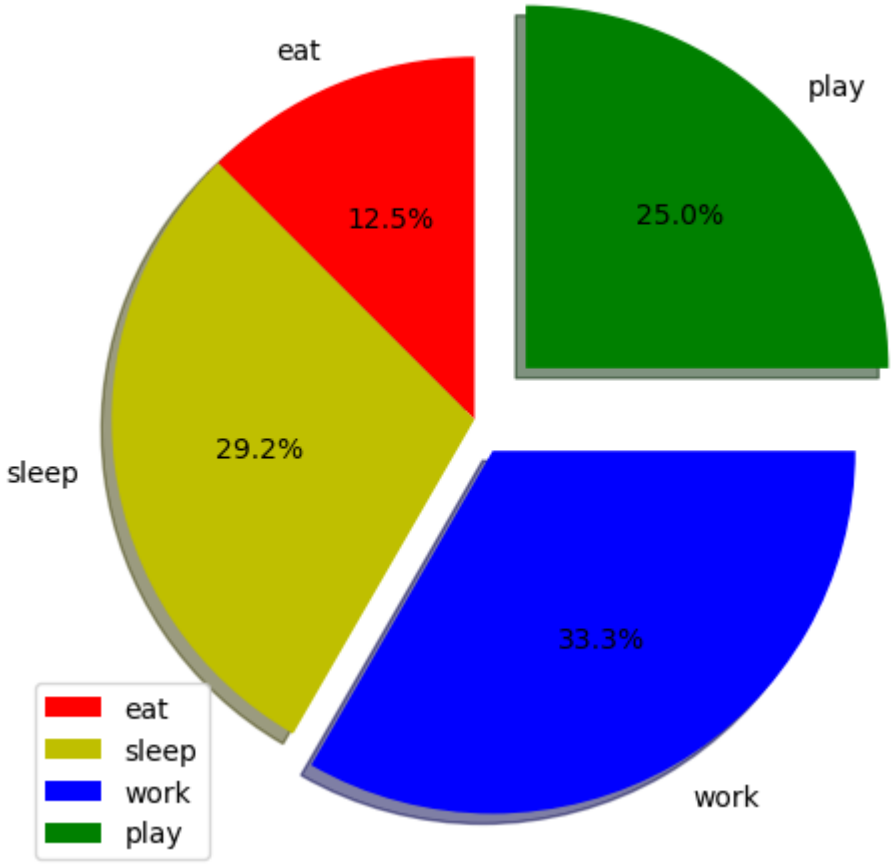





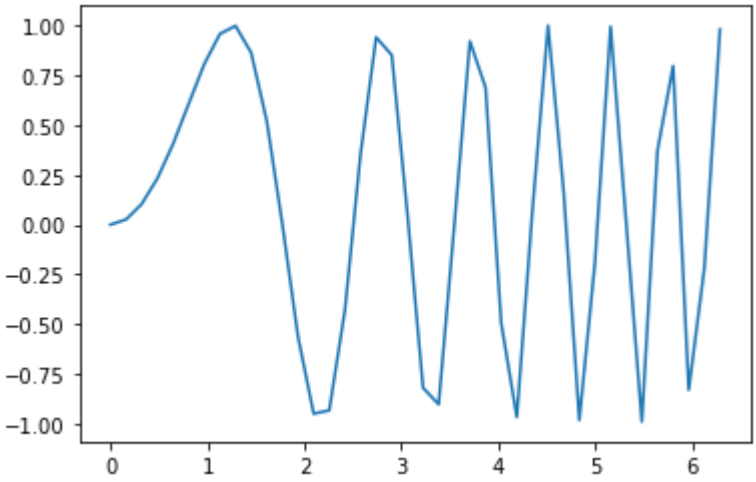

In [46]:
```
#any mathematical curve can be ploted
x=np.arange(0,10,0.1)
y=x*x
plt.plot(x,y)
plt.show()
```



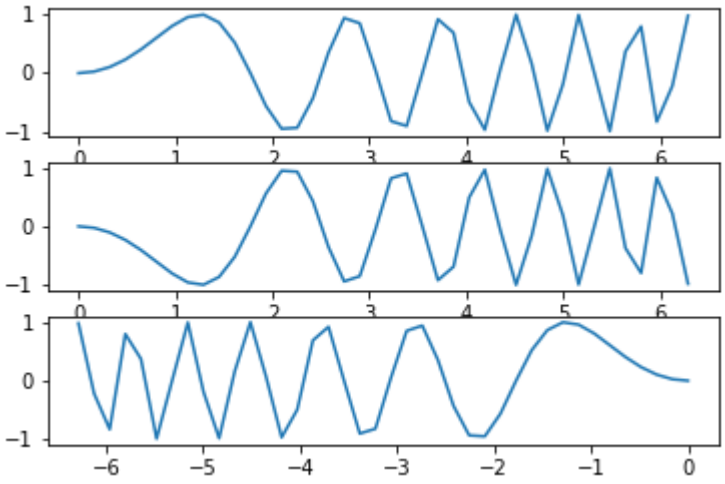
In [55]:
```
#Creating your figure:
#figure() function is used to create a figure of your own plots
fig=plt.figure(figsize=(6,6),dpi=100,facecolor='white')
acts=['eat','sleep','work','play']
time=[3,7,8,6]
color=['r','y','b','g']
plt.pie(time,labels=acts,colors=color,startangle=90,
        explode=[0,0,0.1,0.2],shadow=True,autopct='%1.1f%%')
plt.legend(loc=3)
#plt.show()#do not show the plot in case of saving it as figure
plt.savefig(r'C:\Users\Admin\Desktop\my_fig1.jpg')
#it is used to save your figure on desired location
```



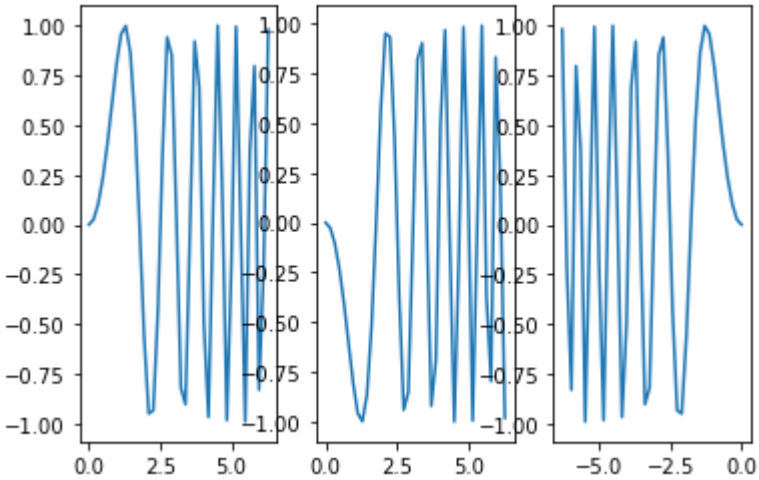
In [69]:
```
#subploting: dividing a plot into multiple plots
x=np.linspace(0,2*np.pi,40)
y=np.sin(x**2)
fig,ax=plt.subplots()# it returns two values one is figure and other
                     #is axis(one or more) in the form of numpy array
ax.plot(x,y)
plt.show()
```



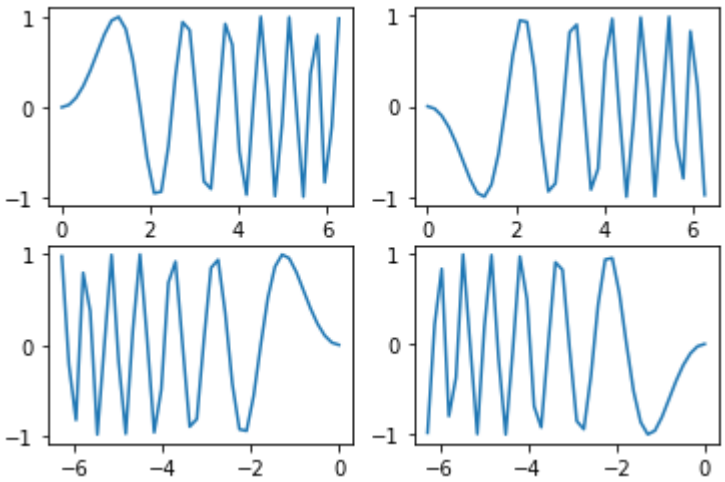

In [71]:
```
fig,ax=plt.subplots(3,1)#returns figure and 2d numpy array of 3x1 order
                        # it is used for vertical ploting of multiple plots
ax[0].plot(x,y)
ax[1].plot(x,-y)
ax[2].plot(-x,y)
plt.savefig(r'C:\Users\Admin\Desktop\my_fig2.pdf')
# return fig is saved in memory
```



In [72]:
```
fig,ax=plt.subplots(1,3)# used for Horizantal ploting of multiple plots(1x3
ax[0].plot(x,y)
ax[1].plot(x,-y)
ax[2].plot(-x,y)
plt.show()
#plt.savefig(r'C:\Users\Admin\Desktop\my_fig2.pdf')
```



In [73]:
```
fig,ax=plt.subplots(2,2)#returns figure and 2d numpy array of 2x2 order
                        # it is used for matrix ploting of multiple plots(2
ax[0,0].plot(x,y)
ax[0,1].plot(x,-y)
ax[1,0].plot(-x,y)
ax[1,1].plot(-x,-y)
plt.show()
#plt.savefig(r'C:\Users\Admin\Desktop\my_fig2.pdf')
# return fig is saved in memory
```



In [ ]:
```
#matplotlib ends
```