Data visulization:

- matplotlib
- seaborn

Matplotlib:

- matplotlib is a one of the best python library for data visulization
- it is written by john D Hunter in 2003

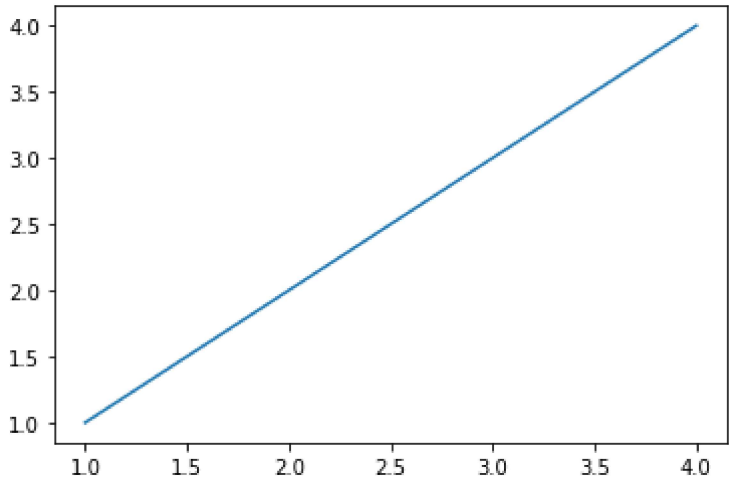
Different types of plots:

- Lineplot
- Scatterplot
- Bar plot
- Box plot
- Histogram plot : 1D data visulization
- Pie plot
- how to read a image and how to display image in matplotlib
- how to save the image

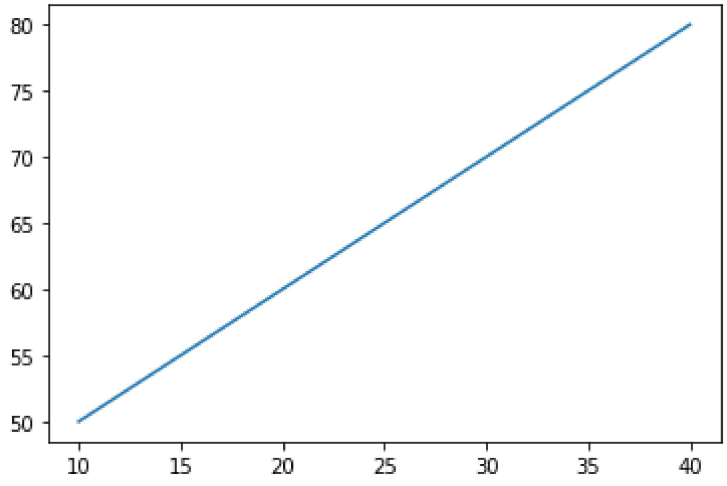
In [1]:
```
# import the lib

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Line plot:

In [2]:
```
x=[1,2,3,4]
y=[1,2,3,4]

plt.plot(x,y)
plt.show()
```



In [3]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.plot(x,y)
plt.show()
```

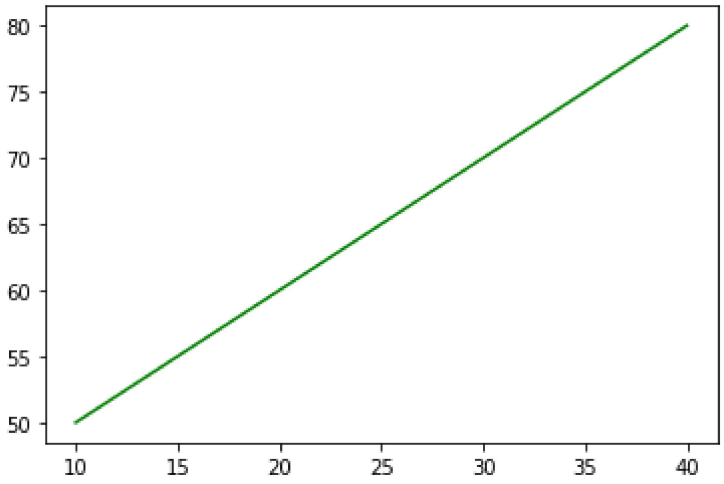


In [4]:
```
help(plt.plot)
```

```
Help on function plot in module matplotlib.pyplot:

plot(*args, scalex=True, scaley=True, data=None, **kwargs)
    Plot y versus x as lines and/or markers.

    Call signatures::

        plot([x], y, [fmt], *, data=None, **kwargs)
        plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)

    The coordinates of the points or line nodes are given by *x*, *y*.

    The optional parameter *fmt* is a convenient way for defining basic
    formatting like color, marker and linestyle. It's a shortcut string
    notation described in the *Notes* section below.

    >>> plot(x, y)        # plot x and y using default line style and color
    >>> plot(x, y, 'bo')  # plot x and y using blue circle markers
    >>> plot(y)           # plot y using x as index array 0..N-1
```

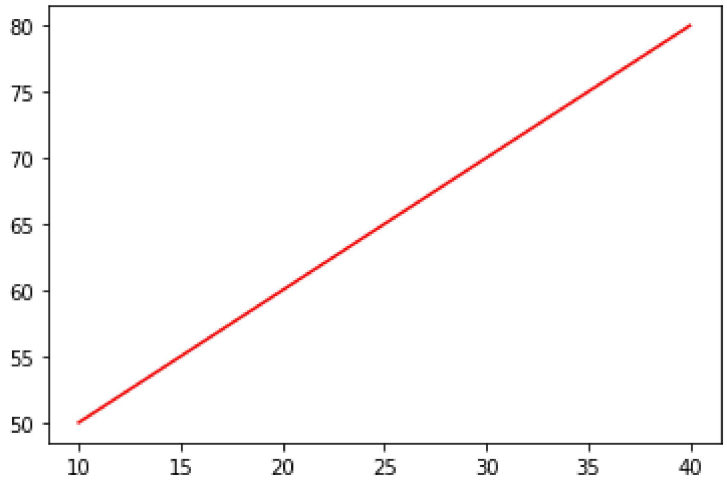
In [5]:
```
plt.plot(x,y,color='g')
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x13ca72a0588>]
```



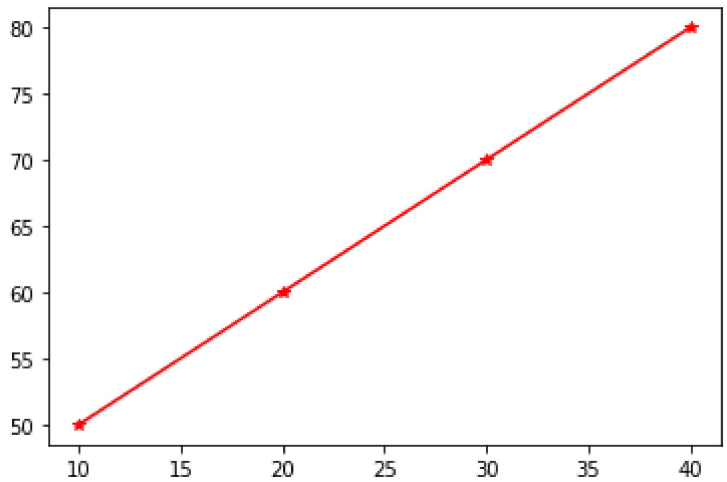
In [6]:
```
plt.plot(x,y,'r')
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x13ca732b390>]
```



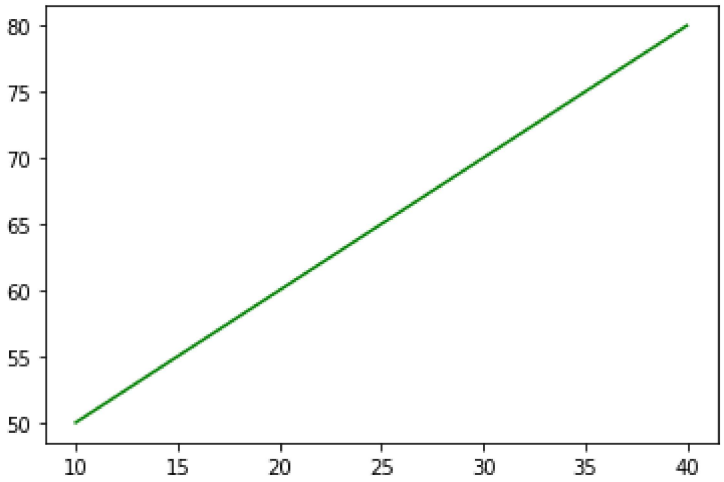
In [10]:
```
plt.plot(x,y,'r-*')
```

Out[10]:
```
[<matplotlib.lines.Line2D at 0x13ca73ee048>]
```



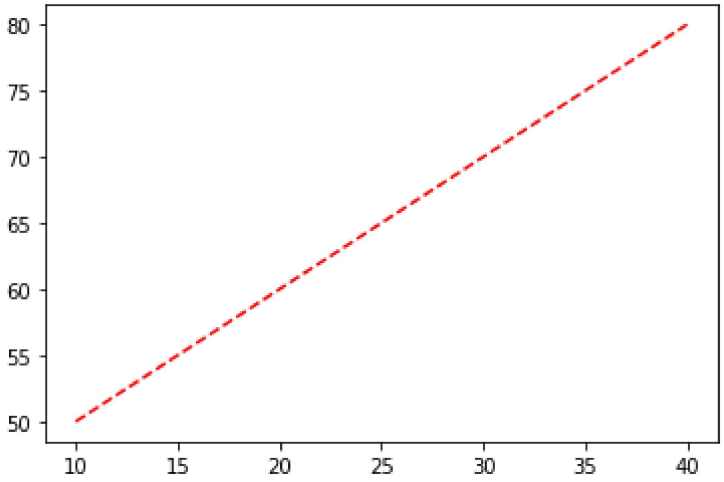
In [11]:
```
plt.plot(x,y,color='#008000')
```

Out[11]:
```
[<matplotlib.lines.Line2D at 0x13ca7447e10>]
```



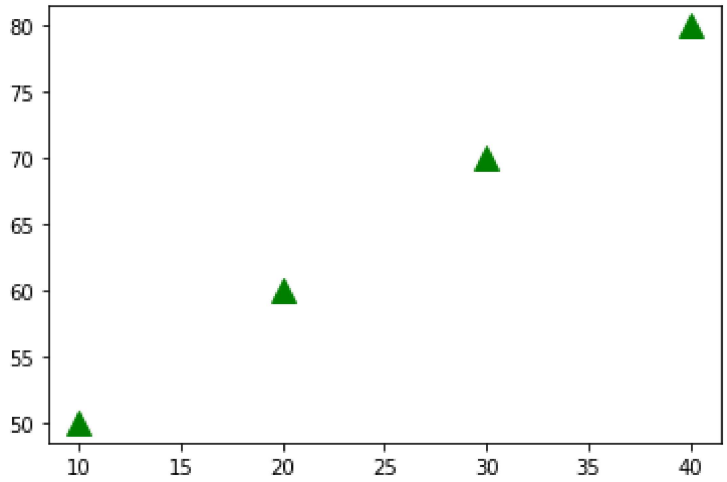
In [12]:
```
plt.plot(x,y,'r',linestyle='dashed')
```

Out[12]:
```
[<matplotlib.lines.Line2D at 0x13ca94bd5f8>]
```

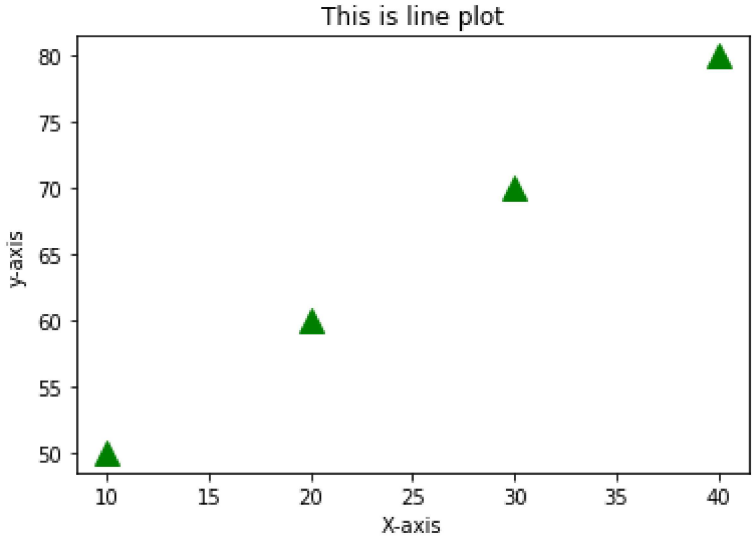


In [14]:
```
plt.plot(x,y,'g^',markersize=12)
```

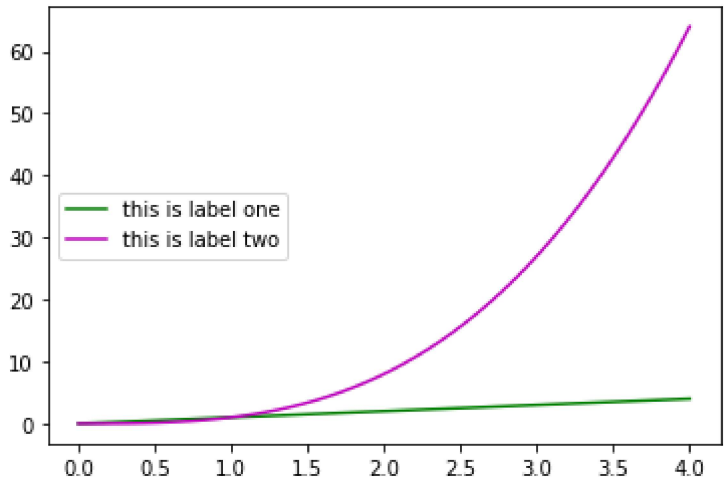
Out[14]:
```
[<matplotlib.lines.Line2D at 0x13ca96f7160>]
```



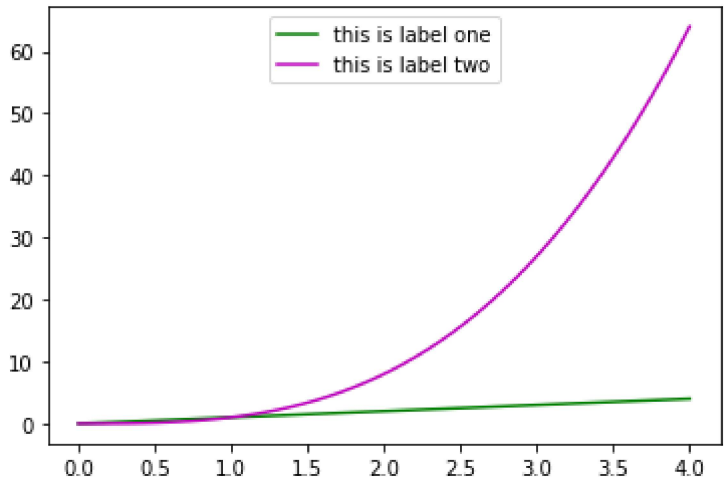
In [16]:
```
plt.plot(x,y,'g^',markersize=12)
plt.title("This is line plot")
plt.xlabel("X-axis")
plt.ylabel("y-axis")
plt.show()
```



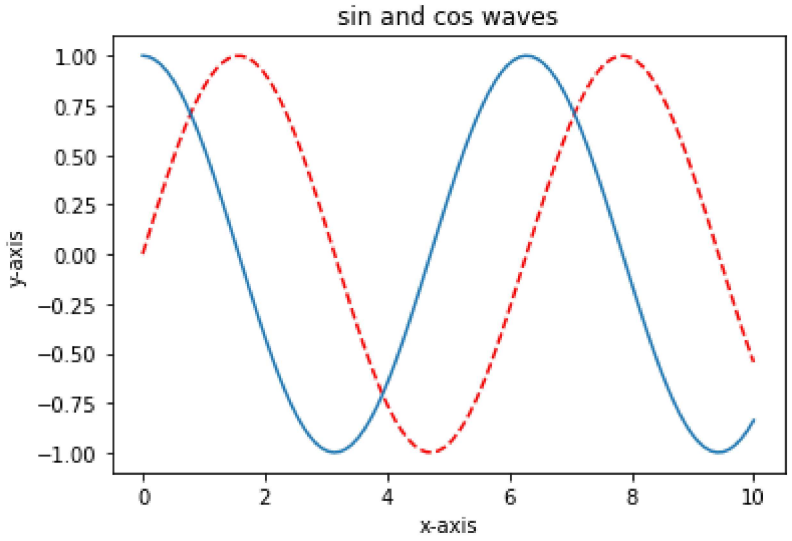
In [26]:
```
x=np.linspace(0,4,50)
plt.plot(x,x,'g',label="this is label one")
plt.plot(x,x**3,'m',label="this is label two")
#plt.legend(loc='upper right')
plt.legend(loc=6)
plt.show()
```



In [27]:
```
x=np.linspace(0,4,50)
plt.plot(x,x,'g',label="this is label one")
plt.plot(x,x**3,'m',label="this is label two")
plt.legend(loc=9)
plt.show()
```



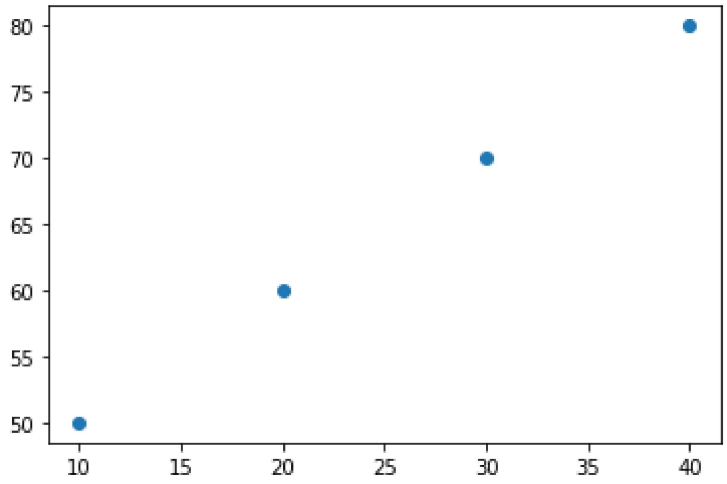
In [30]:
```
x=np.linspace(0,10,100)
plt.plot(x,np.sin(x),'r',linestyle='--')
plt.plot(x,np.cos(x))
plt.title("sin and cos waves")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.show()
```



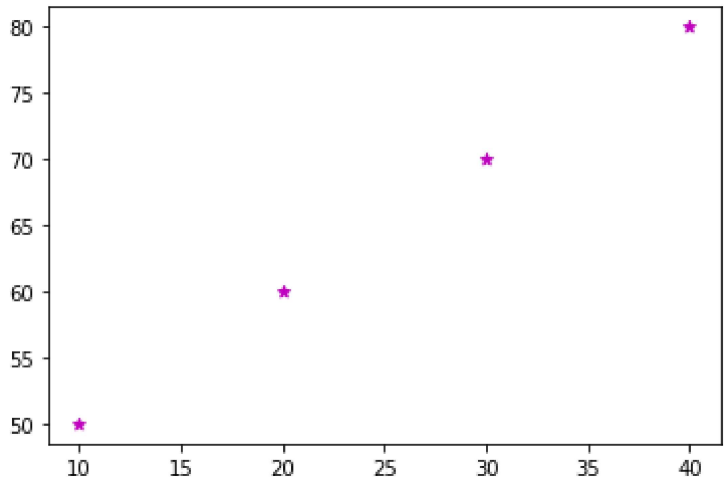
Scatter plots:

- Display the data in points

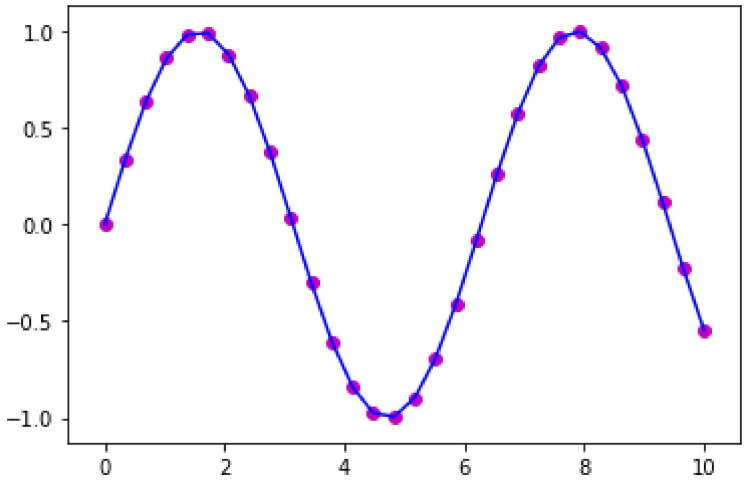
In [31]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.scatter(x,y)
plt.show()
```



In [32]:
```
plt.scatter(x,y,color='m',marker='*')
plt.show()
```



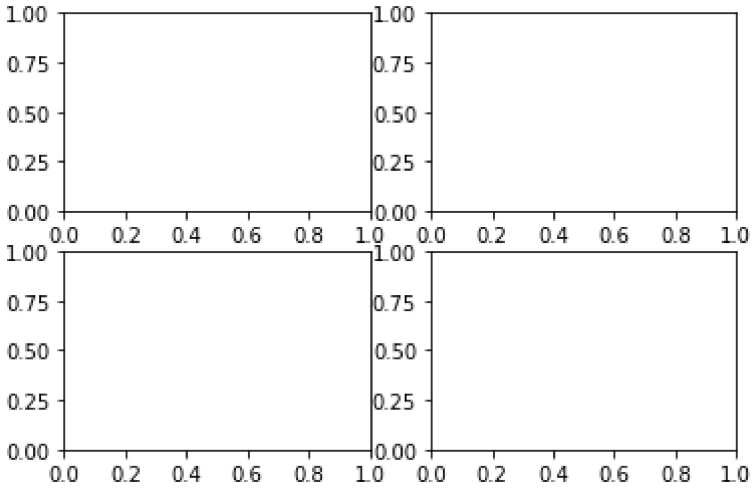
In [34]:
```
x=np.linspace(0,10,30)
y=np.sin(x)
plt.plot(x,y,c='b')
plt.scatter(x,y,c='m')
plt.show()
```



subplots:

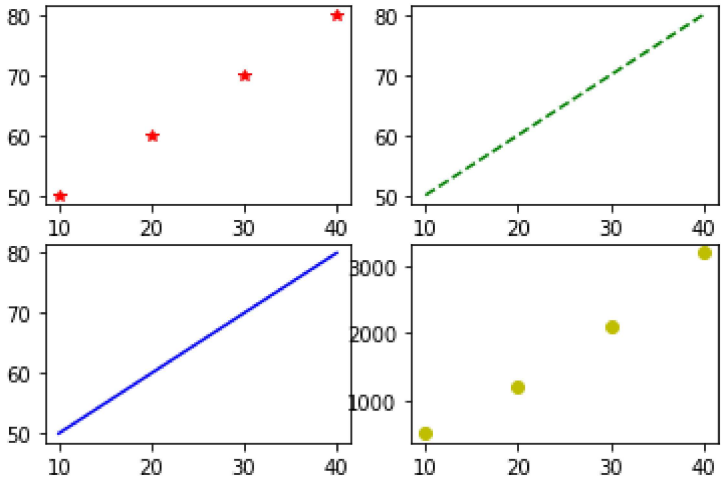
syntax: plt.subplots(row,column,pos)

In [36]:
```
plt.subplots(2,2) # 2 rows and 2 columns
plt.show()
```



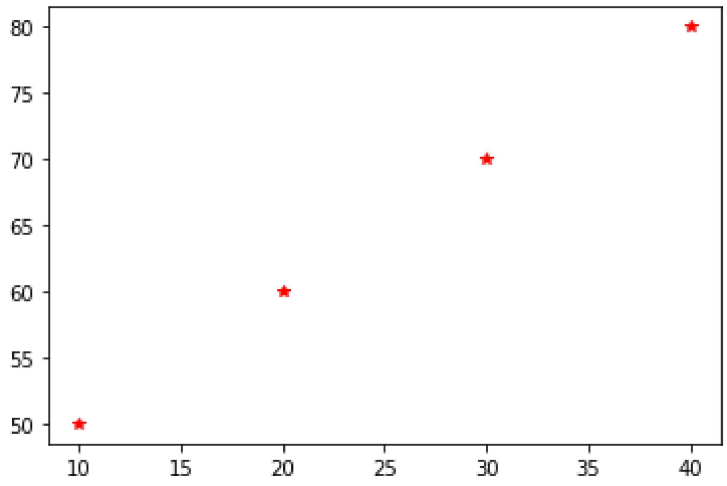
In [44]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.subplot(2,2,1)  # 2 rows,2 columns and 1 position
plt.plot(x,y,'r*')

plt.subplot(2,2,2)
plt.plot(x,y,'g--')

plt.subplot(2,2,3)
plt.plot(x,y,'b-')

plt.subplot(2,2,4)
plt.plot(x,x*y,'yo')
```

Out[44]: [<matplotlib.lines.Line2D at 0x13ca99d6588>]

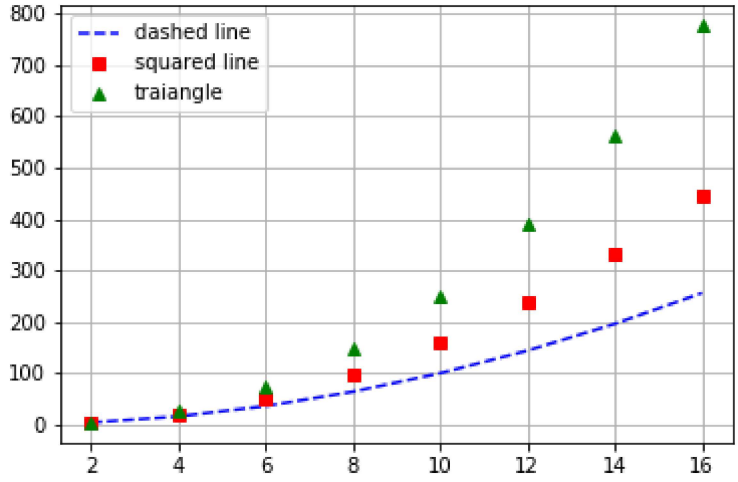


In [106]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.subplot(1,1,1)  # 2 rows,2 columns and 1 position
plt.plot(x,y,'r*')

# plt.subplot(1,2,2)  # 2 rows,2 columns and 1 position
# plt.plot(x,y,'r*')
```

Out[106]: [<matplotlib.lines.Line2D at 0x13cae3d85f8>]

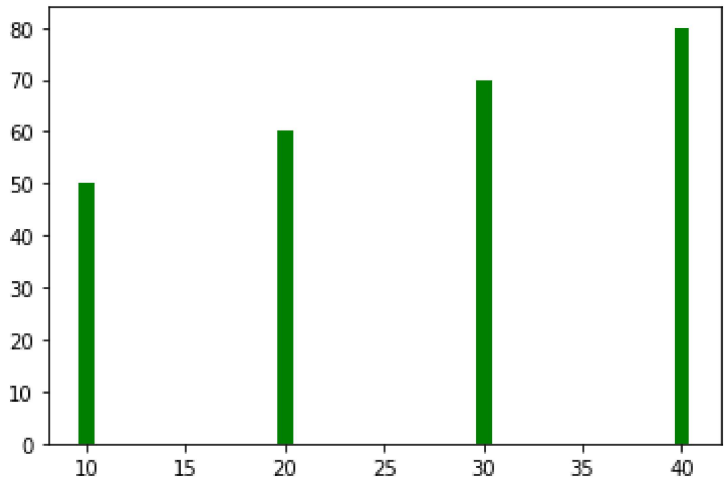


In [46]:
```
x=np.array([2,4,6,8,10,12,14,16])
plt.plot(x,x**2,'b--',label="dashed line")
plt.plot(x,x**2.2,'rs',label='squared line')
plt.plot(x,x**2.4,'g^',label='traiangle')
plt.legend()
plt.grid()
plt.show()
```

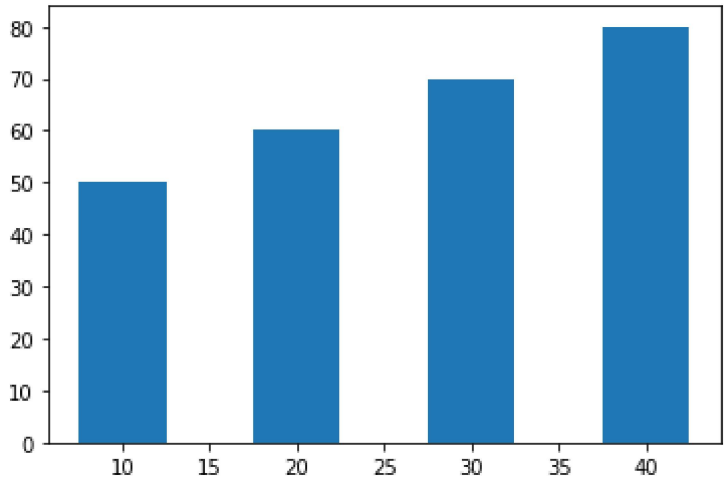


Bar plot

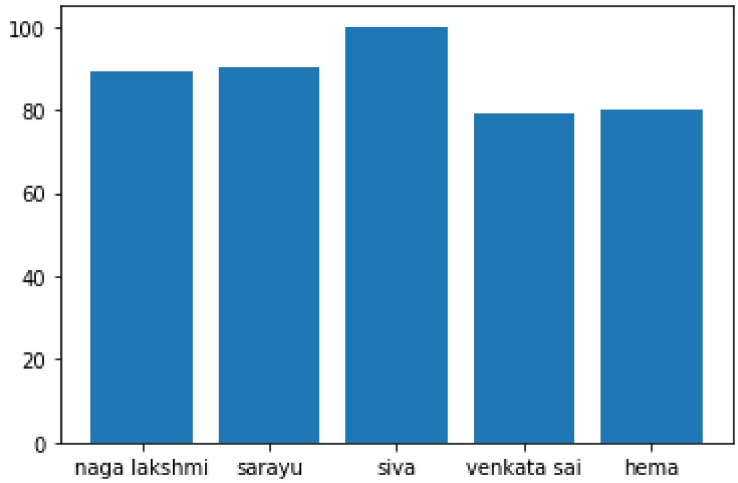
In [47]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.bar(x,y,color='g')
plt.show()
```



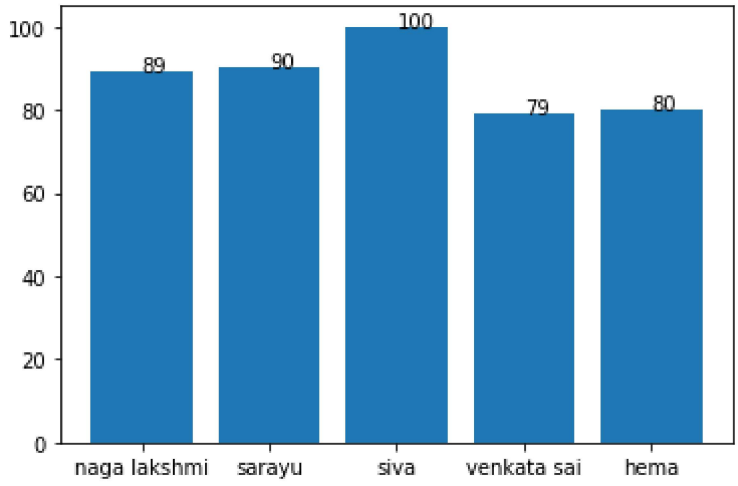
In [49]:
```
x=np.array([10,20,30,40])
y=np.array([50,60,70,80])
plt.bar(x,y,width=5)
plt.show()
```



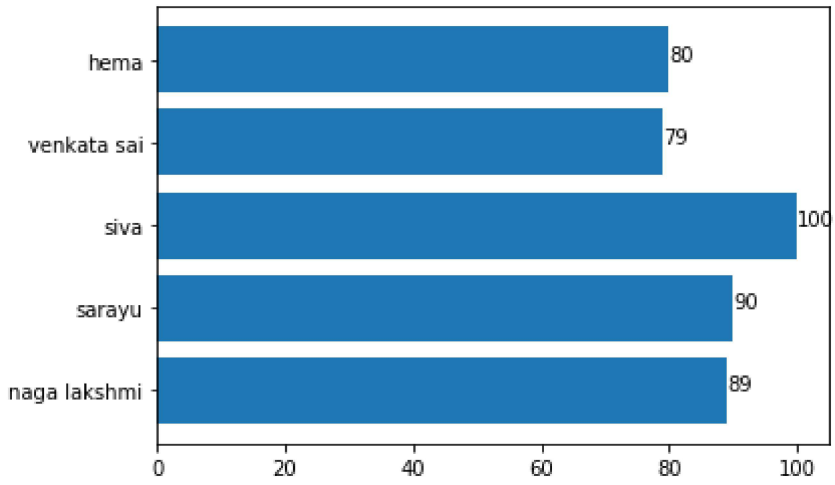
In [50]:
```
x=['naga lakshmi','sarayu','siva','venkata sai','hema']
y=[89,90,100,79,80]

plt.bar(x,y)
plt.show()
```



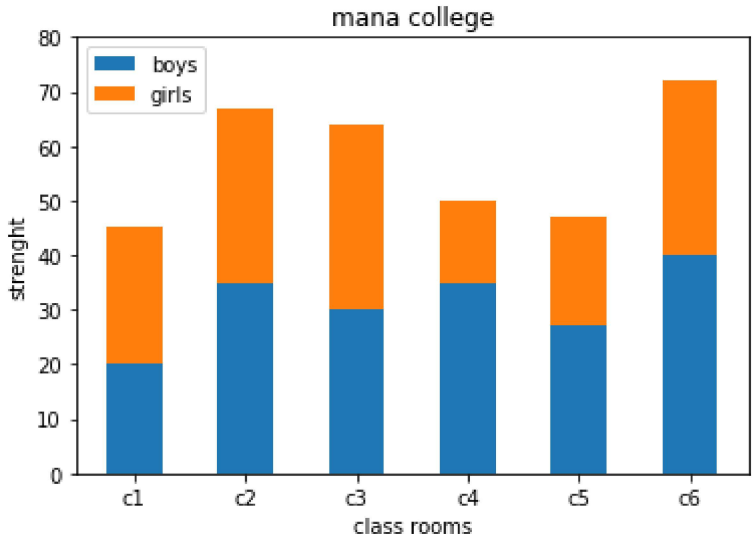
In [59]:
```
x=['naga lakshmi','sarayu','siva','venkata sai','hema']
y=[89,90,100,79,80]

plt.bar(x,y)

for a,b in zip(x,y):
    #plt.text(a,b,str(b))
    #print(b)
    plt.text(a,b,str(b))

plt.show()
```



In [65]:
```
# display the bar graph in horizontal
x=['naga lakshmi','sarayu','siva','venkata sai','hema']
y=[89,90,100,79,80]
plt.barh(x,y)
for a,b in zip(y,x):
    plt.text(a,b,str(a))

plt.show()
```



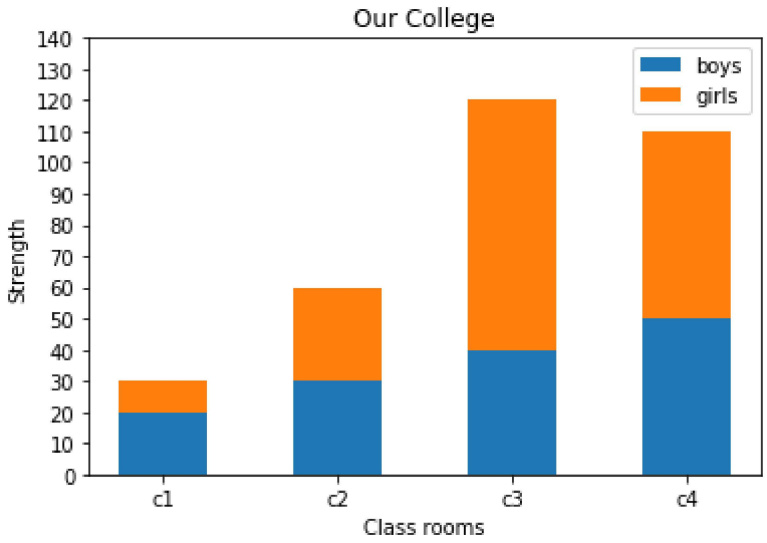
In [68]:
```
b=[20,35,30,35,27,40]
g=[25,32,34,15,20,32]

i=np.arange(0,6)

p1=plt.bar(i,b,width=0.5)
p2=plt.bar(i,g,width=0.5,bottom=b)

plt.xlabel("class rooms")
plt.ylabel("strenght")
plt.title("mana college")
plt.xticks(i,('c1','c2','c3','c4','c5','c6'))
plt.yticks(np.arange(0,90,10))

plt.legend((p1[0],p2[0]),('boys','girls'))
plt.show()
```



In [69]:
```
np.arange(0,5)
```

Out[69]:
```
array([0, 1, 2, 3, 4])
```

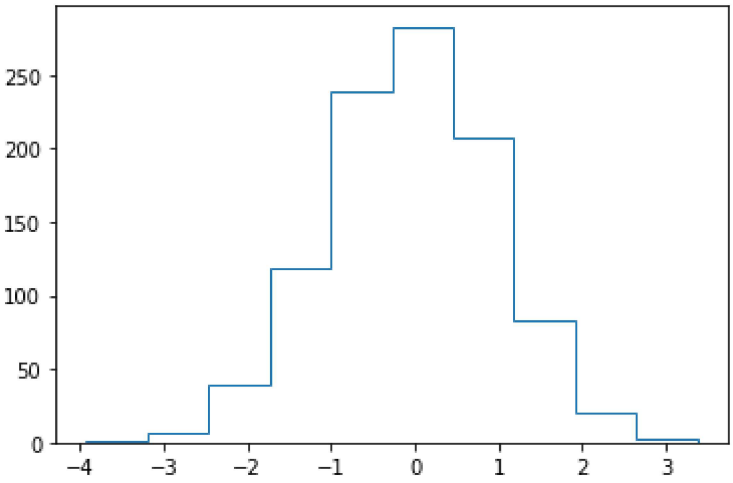
In [75]:
```
### let's take 5 min break

b=[20,30,40,50]
g=[10,30,80,60]

i=np.arange(0,4)

p1=plt.bar(i,b,width=0.5)
p2=plt.bar(i,g,width=0.5,bottom=b)


plt.xlabel("Class rooms")
plt.ylabel("Strength")
plt.title("Our College")


plt.xticks(i,('c1','c2','c3','c4'))
plt.yticks(np.arange(0,150,10))

plt.legend((p1[0],p2[0]),("boys","girls"))
plt.show()
```



Hist plot:

- for 1D array visulization we can you hist plots
- Frequency distribution data

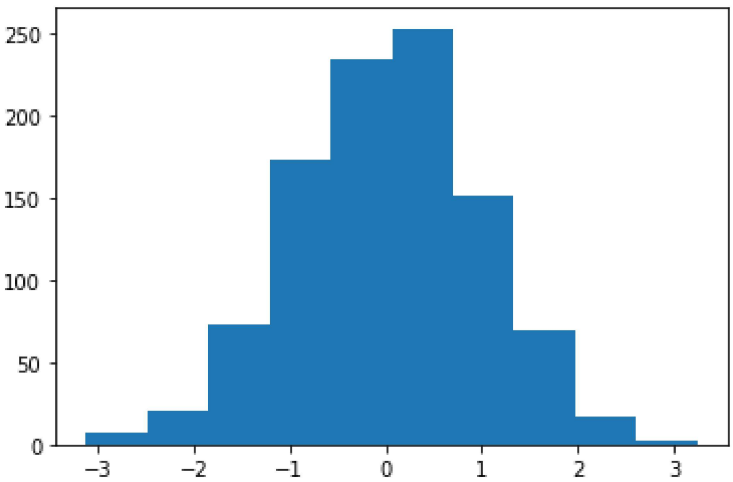
In [80]:
```
x=np.random.randn(1000)
plt.hist(x,histtype='step')
plt.show()
```



In [78]:
```
help(plt.hist)
```

...

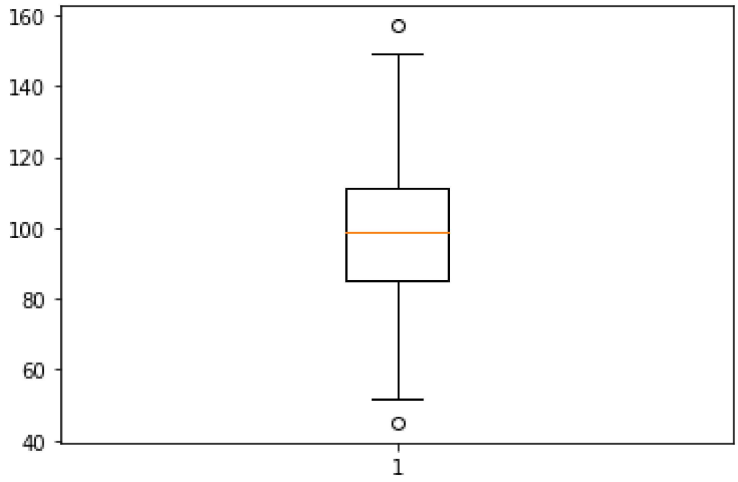
In [81]:
```
x=np.random.randn(1000)
plt.hist(x)
plt.show()
```



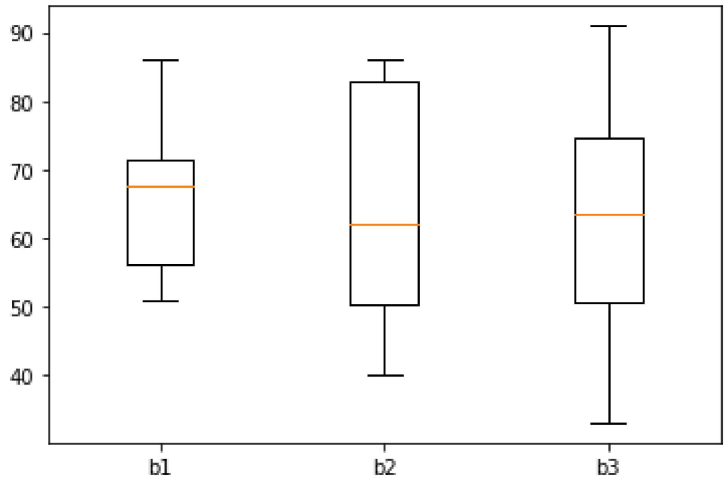
Box plot:

- this is also knows as whisker plot
- it contains min value,max value, first quartile, median, third quartaile

In [83]:
```
data=np.random.normal(100,20,200)
plt.boxplot(data)
plt.show()
```

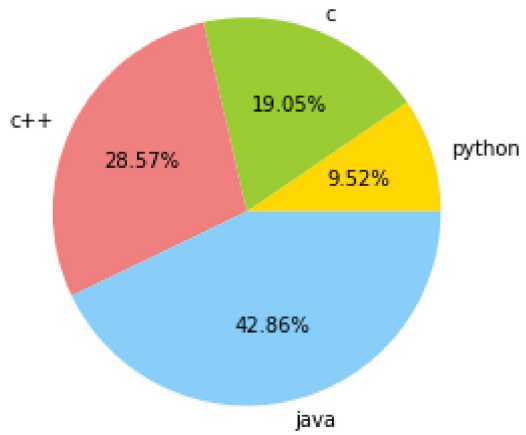


In [85]:
```
a=np.random.randint(50,100,10)
b=np.random.randint(40,90,10)
c=np.random.randint(30,100,10)

plt.boxplot([a,b,c],labels=['b1','b2','b3'])
plt.show()
```

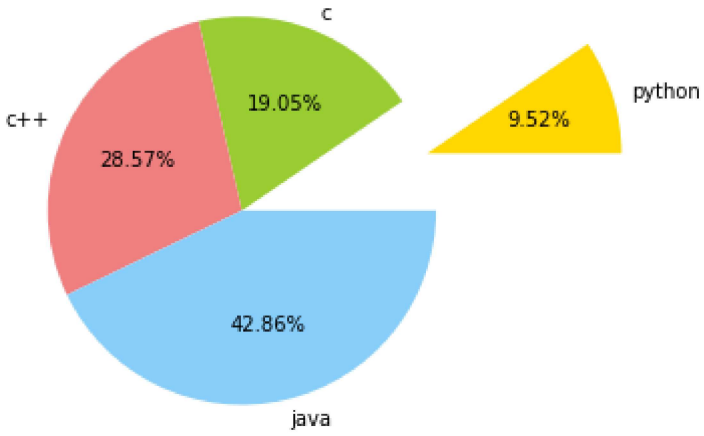


Pie plot

In [93]:
```
labels=['python','c','c++','java']
size=[100,200,300,450]
colors=['gold','yellowgreen','lightcoral','lightskyblue']
plt.pie(size,labels=labels,colors=colors,autopct="%1.2f%%")
plt.axis('equal')
plt.show()
```

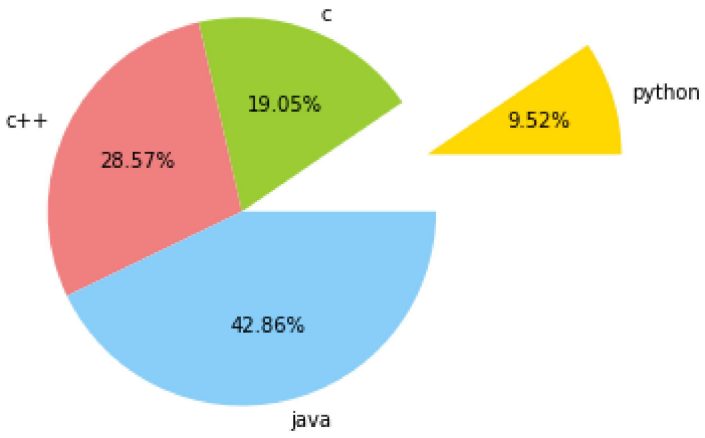


In [96]:
```
labels=['python','c','c++','java']
size=[100,200,300,450]
colors=['gold','yellowgreen','lightcoral','lightskyblue']
explode=(1,0,0,0)
plt.pie(size,labels=labels,explode=explode,colors=colors,autopct="%1.2f%%")
plt.axis('equal')
plt.show()
```



save plot

In [99]:
```
labels=['python','c','c++','java']
size=[100,200,300,450]
colors=['gold','yellowgreen','lightcoral','lightskyblue']
explode=(1,0,0,0)
plt.pie(size,labels=labels,explode=explode,colors=colors,autopct="%1.2f%%")  #ex
plt.axis('equal')
plt.savefig("pieplot2.jpg") #d://folder/filename.png
plt.show()
```

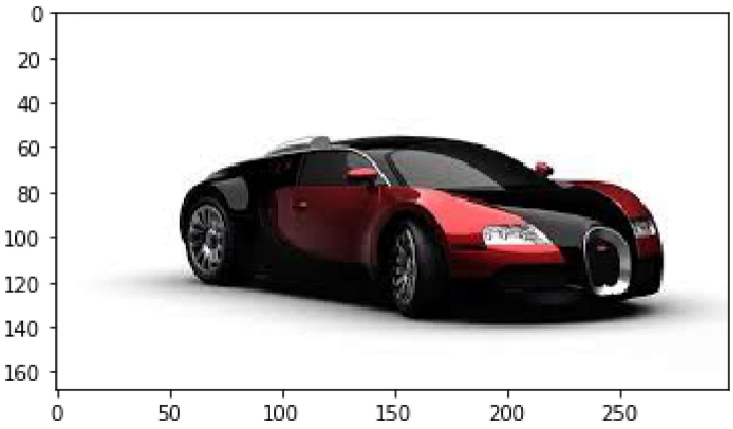


Read the image

In [100]:
```
a=plt.imread("mycar.jpg")
a
```

...

In [107]:
```
plt.imshow(a)
plt.figure(figsize=(12,8))
plt.show()
```




```
<Figure size 864x576 with 0 Axes>
```