**Simple Plotting example**

In [113]:

%**matplotlib** inline

**import** **matplotlib.pyplot** **as** **plt** *#importing matplot lib library*

**import** **numpy** **as** **np**

x = range(100)

*#print x, print and check what is x*

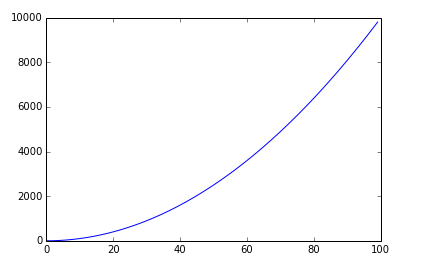
y =[val\*\*2 **for** val **in** x]

*#print y*

plt.plot(x,y) *#plotting x and y*

Out[113]:

[<matplotlib.lines.Line2D at 0x7857bb0>]



**See how [np.linspace](http://docs.scipy.org/doc/numpy/reference/generated/numpy.linspace.html) works.**

**Using Numpy**

In [17]:

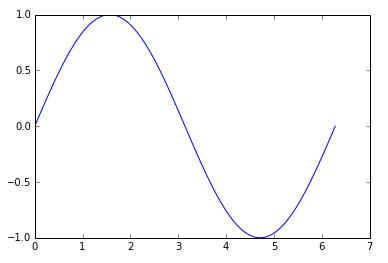
x = np.linspace(0, 2\*np.pi, 100)

y =np.sin(x)

plt.plot(x,y)

Out[17]:

[<matplotlib.lines.Line2D at 0x579aef0>]



In [24]:

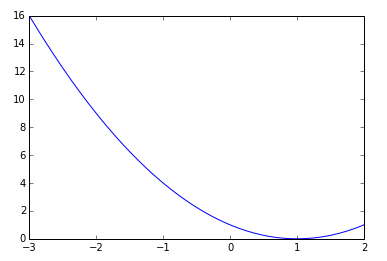
x= np.linspace(-3,2, 200)

Y = x \*\* 2 - 2 \* x + 1.

plt.plot(x,Y)

Out[24]:

[<matplotlib.lines.Line2D at 0x6ffb310>]



In [32]:

*# plotting multiple plots*

x =np.linspace(0, 2 \* np.pi, 100)

y = np.sin(x)

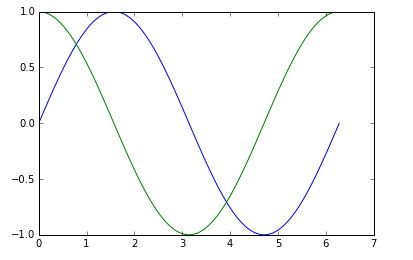
z = np.cos(x)

plt.plot(x,y)

plt.plot(x,z)

plt.show()

*# Matplot lib picks different colors for different plot.*



In [35]:

cd C:\Users\tk\Desktop\Matplot

C:\Users\tk\Desktop\Matplot

In [39]:

data = np.loadtxt('numpy.txt')

plt.plot(data[:,0], data[:,1]) *# plotting column 1 vs column 2*

*# The text in the numpy.txt should look like this*

*# 0 0*

*# 1 1*

*# 2 4*

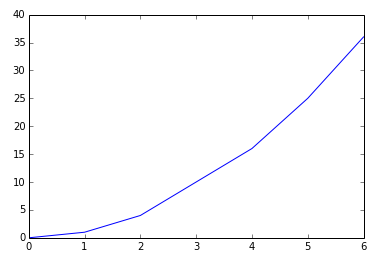
*# 4 16*

*# 5 25*

*# 6 36*

Out[39]:

[<matplotlib.lines.Line2D at 0x740f090>]



In [56]:

data1 = np.loadtxt('scipy.txt') *# load the file*

print data1.T

**for** val **in** data1.T: *#loop over each and every value in data1.T*

plt.plot(data1[:,0], val) *#data1[:,0] is the first row in data1.T*

*# data in scipy.txt looks like this:*

*# 0 0 6*

*# 1 1 5*

*# 2 4 4*

*# 4 16 3*

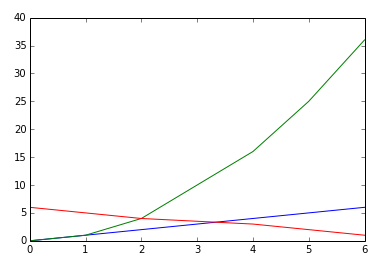
*# 5 25 2*

*# 6 36 1*

[[ 0. 1. 2. 4. 5. 6.]

[ 0. 1. 4. 16. 25. 36.]

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



**Scatter Plots and Bar Graphs**

In [64]:

sct = np.random.rand(20, 2)

print sct

plt.scatter(sct[:,0], sct[:,1]) *# I am plotting a scatter plot.*

[[ 0.51454542 0.61859101]

[ 0.45115993 0.69774873]

[ 0.29051205 0.28594808]

[ 0.73240446 0.41905186]

[ 0.23869394 0.5238878 ]

[ 0.38422814 0.31108919]

[ 0.52218967 0.56526379]

[ 0.60760426 0.80247073]

[ 0.37239096 0.51279078]

[ 0.45864677 0.28952167]

[ 0.8325996 0.28479446]

[ 0.14609382 0.8275477 ]

[ 0.86338279 0.87428696]

[ 0.55481585 0.24481165]

[ 0.99553336 0.79511137]

[ 0.55025277 0.67267026]

[ 0.39052024 0.65924857]

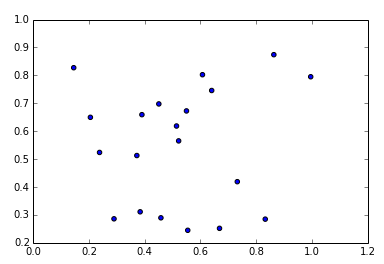
[ 0.66868207 0.25186664]

[ 0.64066313 0.74589812]

[ 0.20587731 0.64977807]]

Out[64]:

<matplotlib.collections.PathCollection at 0x78a7110>



In [65]:

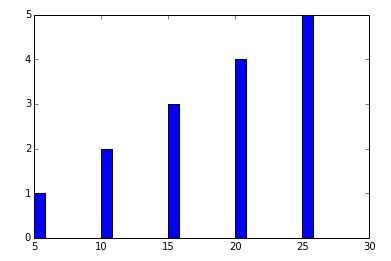
ghj =[5, 10 ,15, 20, 25]

it =[ 1, 2, 3, 4, 5]

plt.bar(ghj, it) *# simple bar graph*

Out[65]:

<Container object of 5 artists>



In [74]:

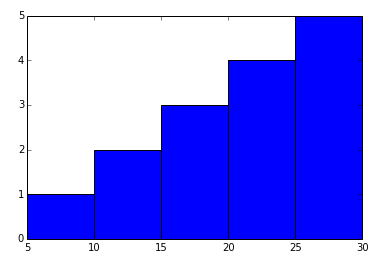
ghj =[5, 10 ,15, 20, 25]

it =[ 1, 2, 3, 4, 5]

plt.bar(ghj, it, width =5)*# you can change the thickness of a bar, by default the bar will have a thickness of 0.8 units*

Out[74]:

<Container object of 5 artists>



In [75]:

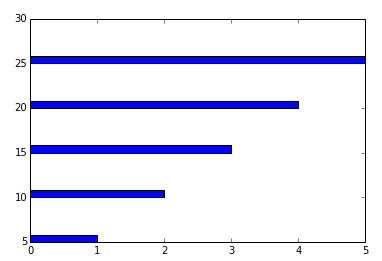
ghj =[5, 10 ,15, 20, 25]

it =[ 1, 2, 3, 4, 5]

plt.barh(ghj, it) *# barh is a horizontal bar graph*

Out[75]:

<Container object of 5 artists>



**<font size = "5">Multiple bar charts </font>**

**In [95]:**

**new\_list = [[5., 25., 50., 20.], [4., 23., 51., 17.], [6., 22., 52., 19.]]**

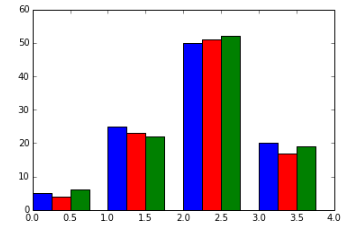
**x = np.arange(4)**

**plt.bar(x + 0.00, new\_list[0], color ='b', width =0.25)**

**plt.bar(x + 0.25, new\_list[1], color ='r', width =0.25)**

**plt.bar(x + 0.50, new\_list[2], color ='g', width =0.25)**

***#plt.show()***

****

**In [100]:**

***#Stacked Bar charts***

**p = [5., 30., 45., 22.]**

**q = [5., 25., 50., 20.]**

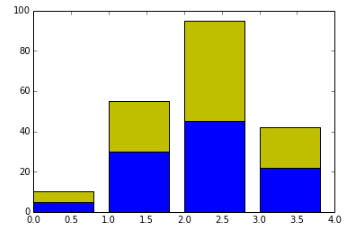
**x =range(4)**

**plt.bar(x, p, color ='b')**

**plt.bar(x, q, color ='y', bottom =p)**

**Out[100]:**

**<Container object of 4 artists>**

****

**In [35]:**

***# plotting more than 2 values***

**A = np.array([5., 30., 45., 22.])**

**B = np.array([5., 25., 50., 20.])**

**C = np.array([1., 2., 1., 1.])**

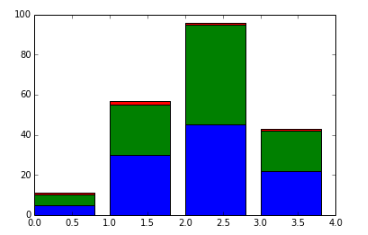
**X = np.arange(4)**

**plt.bar(X, A, color = 'b')**

**plt.bar(X, B, color = 'g', bottom = A)**

**plt.bar(X, C, color = 'r', bottom = A + B) *# for the third argument, I use A+B***

**plt.show()**

****

**In [94]:**

**black\_money = np.array([5., 30., 45., 22.])**

**white\_money = np.array([5., 25., 50., 20.])**

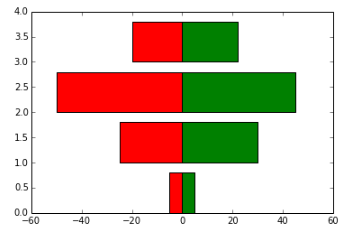
**z = np.arange(4)**

**plt.barh(z, black\_money, color ='g')**

**plt.barh(z, -white\_money, color ='r')*# - notation is needed for generating, back to back charts***

**Out[94]:**

**<Container object of 4 artists>**

****

**Other Plots**

**In [114]:**

***#Pie charts***

**y = [5, 25, 45, 65]**

**plt.pie(y)**

**Out[114]:**

**([<matplotlib.patches.Wedge at 0x7a19d50>,**

**<matplotlib.patches.Wedge at 0x7a252b0>,**

**<matplotlib.patches.Wedge at 0x7a257b0>,**

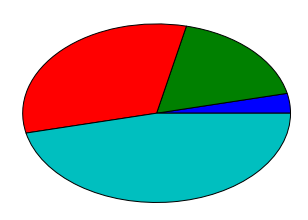
**<matplotlib.patches.Wedge at 0x7a25cb0>],**

**[<matplotlib.text.Text at 0x7a25070>,**

**<matplotlib.text.Text at 0x7a25550>,**

**<matplotlib.text.Text at 0x7a25a50>,**

**<matplotlib.text.Text at 0x7a25f50>])**

****

**In [115]:**

***#Histograms***

**d = np.random.randn(100)**

**plt.hist(d, bins = 20)**

**Out[115]:**

**(array([ 2., 3., 2., 1., 2., 6., 5., 7., 10., 12., 9.,**

**12., 11., 5., 6., 4., 1., 0., 1., 1.]),**

**array([-2.9389701 , -2.64475645, -2.35054281, -2.05632916, -1.76211551,**

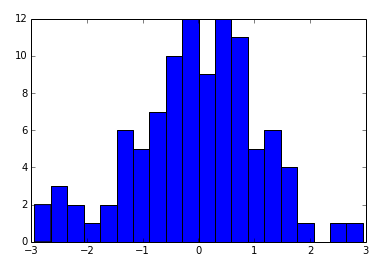
**-1.46790186, -1.17368821, -0.87947456, -0.58526092, -0.29104727,**

**0.00316638, 0.29738003, 0.59159368, 0.88580733, 1.18002097,**

**1.47423462, 1.76844827, 2.06266192, 2.35687557, 2.65108921,**

**2.94530286]),**

**<a list of 20 Patch objects>)**

****

**In [116]:**

**d = np.random.randn(100)**

**plt.boxplot(d)**

***#1) The red bar is the median of the distribution***

***#2) The blue box includes 50 percent of the data from the lower quartile to the upper quartile.***

***# Thus, the box is centered on the median of the data.***

**Out[116]:**

**{'boxes': [<matplotlib.lines.Line2D at 0x7cca090>],**

**'caps': [<matplotlib.lines.Line2D at 0x7c02d70>,**

**<matplotlib.lines.Line2D at 0x7cc2c90>],**

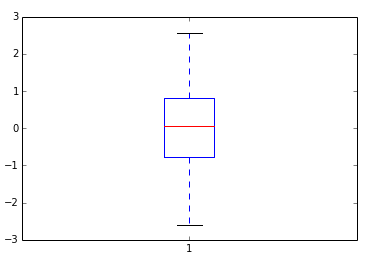
**'fliers': [<matplotlib.lines.Line2D at 0x7cca850>,**

**<matplotlib.lines.Line2D at 0x7ccae10>],**

**'medians': [<matplotlib.lines.Line2D at 0x7cca470>],**

**'whiskers': [<matplotlib.lines.Line2D at 0x7c02730>,**

**<matplotlib.lines.Line2D at 0x7cc24b0>]}**

****

**In [118]:**

**d = np.random.randn(100, 5) *# generating multiple box plots***

**plt.boxplot(d)**

**Out[118]:**

**{'boxes': [<matplotlib.lines.Line2D at 0x7f49d70>,**

**<matplotlib.lines.Line2D at 0x7ea1c90>,**

**<matplotlib.lines.Line2D at 0x7eafb90>,**

**<matplotlib.lines.Line2D at 0x7ebea90>,**

**<matplotlib.lines.Line2D at 0x7ece990>],**

**'caps': [<matplotlib.lines.Line2D at 0x7f2b3b0>,**

**<matplotlib.lines.Line2D at 0x7f49990>,**

**<matplotlib.lines.Line2D at 0x7ea14d0>,**

**<matplotlib.lines.Line2D at 0x7ea18b0>,**

**<matplotlib.lines.Line2D at 0x7eaf3d0>,**

**<matplotlib.lines.Line2D at 0x7eaf7b0>,**

**<matplotlib.lines.Line2D at 0x7ebe2d0>,**

**<matplotlib.lines.Line2D at 0x7ebe6b0>,**

**<matplotlib.lines.Line2D at 0x7ece1d0>,**

**<matplotlib.lines.Line2D at 0x7ece5b0>],**

**'fliers': [<matplotlib.lines.Line2D at 0x7e98550>,**

**<matplotlib.lines.Line2D at 0x7e98930>,**

**<matplotlib.lines.Line2D at 0x7ea8470>,**

**<matplotlib.lines.Line2D at 0x7ea8a10>,**

**<matplotlib.lines.Line2D at 0x7eb6370>,**

**<matplotlib.lines.Line2D at 0x7eb6730>,**

**<matplotlib.lines.Line2D at 0x7ec6270>,**

**<matplotlib.lines.Line2D at 0x7ec6810>,**

**<matplotlib.lines.Line2D at 0x8030170>,**

**<matplotlib.lines.Line2D at 0x8030710>],**

**'medians': [<matplotlib.lines.Line2D at 0x7e98170>,**

**<matplotlib.lines.Line2D at 0x7ea8090>,**

**<matplotlib.lines.Line2D at 0x7eaff70>,**

**<matplotlib.lines.Line2D at 0x7ebee70>,**

**<matplotlib.lines.Line2D at 0x7eced70>],**

**'whiskers': [<matplotlib.lines.Line2D at 0x7f2bb50>,**

**<matplotlib.lines.Line2D at 0x7f491b0>,**

**<matplotlib.lines.Line2D at 0x7e98cf0>,**

**<matplotlib.lines.Line2D at 0x7ea10f0>,**

**<matplotlib.lines.Line2D at 0x7ea8bf0>,**

**<matplotlib.lines.Line2D at 0x7ea8fd0>,**

**<matplotlib.lines.Line2D at 0x7eb6cd0>,**

**<matplotlib.lines.Line2D at 0x7eb6ed0>,**

**<matplotlib.lines.Line2D at 0x7ec6bd0>,**

**<matplotlib.lines.Line2D at 0x7ec6dd0>]}**

