For Loops!!!!!

Today we are going to learn what is a for loop and how they work. A for loop, goes across every item in a list. then when you have that item you can do something with it. For example if you made a grocery list. You could say for every item in the list I will pick it off the shelf and put it in my cart. we will do slightly different things like math and printing but the idea is similar!

So lets make a list and then loop over it. we will start with a small list we make ourselves.

In [3]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
mylist=[50,30,40,10,100]
mylist
```

Out[2]:
```
[50, 30, 40, 10, 100]
```

In [3]:
```
for i in mylist:
    print (i)
```

```
50
30
40
10
100
```

Okay I admit that wasn't that exciting. It did the same thing as above but it did it one by one. But it gets better

In [4]:
```
# A cool trick I learned recently is that you can force it to the same line.
#use end=" ", usually python has
# a default end which is a new line.  I guess you could add anything!
for i in mylist:
    print (i,end=" ")
```

```
50 30 40 10 100
```

In [4]:
```
for i in mylist:
    print (i,end=" do I really want the line to end\n")
```

```
50 do I really want the line to end
30 do I really want the line to end
40 do I really want the line to end
10 do I really want the line to end
100 do I really want the line to end
```

In [7]:
```
mystrlist=['env','chem','bio','psych']
for dept in mystrlist:
    print (dept)
```

```
env
chem
```

```
bio
psych
```

Exercise 1. Can you make a list and loop over it? Don't forget the colon and the indent. If you put the colon when typing ipyhton will automatically indent.

In [ ]:

```
```

What happens if something is not indented?

In [8]:

```
mystrlist=['env','chem','bio','psych']
for dept in mystrlist:
    print (dept)
print (dept)
```

```
env
chem
bio
psych
psych
```

after the indent the for loop is over!

In [1]:

```
mystrlist=['env','chem','bio','psych']
for dept in mystrlist:
    print('in loop')
    print (dept)
print('out of loop')
print (dept)
```

```
in loop
env
in loop
chem
in loop
bio
in loop
psych
out of loop
psych
```

Exercise 2. Go back to your list. Inside the for loop print dept and mystrlist. Outside the for loop print the same thing.

In [ ]:

```
```

But lets make a loop with numbers. What number is not in the list.

In [20]:

```
for  i in np.arange(0,10):
    print (i)
```

```
0
1
2
3
4
5
6
7
8
9
```

Exercise 3. What if we want 10? Can you do that

In [10]:

```
```

Exercise 4. can you do 0 to 10 by 2's?

In [11]:

```
```

```
0
2
4
6
8
10
```

So now we have two ways to loop. the first is to loop over a list. The second is to loop over a range (arange) of numbers. could we loop over numbers and call a list? We can do this but it is generally not needed in Python

I added the \ as a line continuation so it prints nicer

In [7]:

```
mystrlist=['env','chem','bio','psych']
for i in np.arange(0,4):
    print ('i={}  The item in mystrlist[{}] is {}'\
           .format(i,i,mystrlist[i]))
```

```
i=0  The item in mystrlist[0] is env
i=1  The item in mystrlist[1] is chem
i=2  The item in mystrlist[2] is bio
i=3  The item in mystrlist[3] is psych
```

Instead of knowing that the length of the list is 4 you could use the len function....

In [10]:

```
mystrlist=['env','chem','bio','psych']
for i in np.arange(0,len(mystrlist)):
    print ('i=',i,' The item in mystrlist[',i,'] is',mystrlist[i])
```

```
i= 0  The item in mystrlist[ 0 ] is env
i= 1  The item in mystrlist[ 1 ] is chem
i= 2  The item in mystrlist[ 2 ] is bio
i= 3  The item in mystrlist[ 3 ] is psych
```

Python has a simple trick to count through a list and give you the list. It may not make sense now but it will later in the semester! PAY ATTENTION TO THIS!

THIS IS HELPFUL!

In [11]:

```
mystrlist=['env','chem','bio','psych']
for i,mystr in enumerate(mystrlist):
    print (i,mystr)
```

```
0 env
1 chem
2 bio
3 psych
```

Exercise 5. Can you make a list with linspace and iterate through it?

In [15]:

as we are iterating we could add to a list. Lets start with an empty list. then lets add to the list the square of the value.

In [10]:
```
mylist=[]
for i in np.linspace(0,10,5):
    mylist.append(i**2)
    print ('i={} The value of appended to mylist is i^2 which ={}'\
           .format(i,i**2))

print('We are done outside of the loop and mylist is ',mylist)
print('and remember they are in python order so mylist[0]\
={} since we appended an empty list'.format(mylist[0]))
```

```
i=0.0 The value of appended to mylist is i^2 which =0.0
i=2.5 The value of appended to mylist is i^2 which =6.25
i=5.0 The value of appended to mylist is i^2 which =25.0
i=7.5 The value of appended to mylist is i^2 which =56.25
i=10.0 The value of appended to mylist is i^2 which =100.0
We are done outside of the loop and mylist is  [0.0, 6.25, 25.0, 56.25, 100.0]
and remember they are in python order so mylist[0]=0.0 since we appended an empt
y list
```

Exercise 6.

Can you make two lists. The first is months. The second is days in that month. Now can you loop through and print

January has 31 days.

February has 29 days.

March has.....

In [ ]:

Exercise 7.

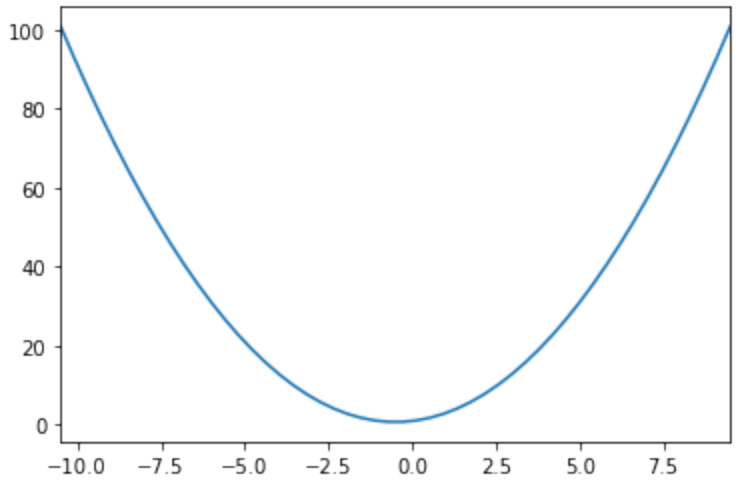
Can you now go back to the earlier assignment of plotting a parabola and do it the long old fashioned way in a for loop? Remember

$$y = ax^2 + bx + c$$

set an a,b,and c. get an x range with linspace or range solve for y plot it.

In [18]:
```
import matplotlib.pylab as plt
a=1
b=1
c=1
```

Out[18]: (-10.5, 9.5)



We can also sum up in a for loop.

In [27]:
```
sum=0
mylist=np.linspace(0.2,0.3,10)
for i in mylist:
    sum+=i   # This is a cool trick and means sum=sum+i
print('the sume is=',sum)
```

```
the sume is= 2.4999999999999996
```

Exercise 8. Now can you determine the average or the mean using a for loop? I would also use the len() function.

In [76]:
```

```

```
the sume is= 2.5
the mean is= 0.25
```

go back to exercise 8 and change the number of items in your list and see if it changes the mean....

In [ ]:
```

```

You can also put for loops inside of for loops

Again This is important! Remember this!

In [6]:
```
for i in np.arange(0,3):
    print('i={}'.format(i))
    for j in np.arange(0,3):
        print('j={}'.format(j))
```

```
i=0
j=0
j=1
j=2
i=1
j=0
j=1
j=2
i=2
j=0
```

```
j=1
j=2
```

You can also pass a parameter to arange in a for loop

In [11]:
```
start=5
stop=12
for i in np.arange(start,stop):
    print('This loop starts at {},\
        is currenly at {}, and ends at {}'.format(start,i,stop))
```

```
This loop starts at 5,        is currenly at 5, and ends at 12
This loop starts at 5,        is currenly at 6, and ends at 12
This loop starts at 5,        is currenly at 7, and ends at 12
This loop starts at 5,        is currenly at 8, and ends at 12
This loop starts at 5,        is currenly at 9, and ends at 12
This loop starts at 5,        is currenly at 10, and ends at 12
This loop starts at 5,        is currenly at 11, and ends at 12
```

Doing the math in for loops like this is "old fashioned" and is how you used to have to program. Having all these built in functions is really nice and does make life so much easier and most of this course is taking advantage of built in functions whenever we can

ANSWERS

Answer Exercise 1. Just make your own list

In [28]:
```
students=['Therese','Colette','Lisa','Hadil']
for student in students:
    print (student)
```

```
Therese
Colette
Lisa
Hadil
```

Answer Exercise 2.

In [30]:
```
students=['Therese','Colette','Lisa','Hadil']
for student in students:
    print (student,students)

print (student,students)
```

```
Therese ['Therese', 'Colette', 'Lisa', 'Hadil']
Colette ['Therese', 'Colette', 'Lisa', 'Hadil']
Lisa ['Therese', 'Colette', 'Lisa', 'Hadil']
Hadil ['Therese', 'Colette', 'Lisa', 'Hadil']
Hadil ['Therese', 'Colette', 'Lisa', 'Hadil']
```

Answer Exercise 3.

In [4]:
```
for  i in np.arange(11):
    print (i)
```

```
0
1
2
3
4
```

```
5
6
7
8
9
10
```

Answer Exercise 4.

In [5]:
```
for  i in np.arange(0,11,2):
    print (i)
```

```
0
2
4
6
8
10
```

Answer Exercise 5

In [33]:
```
new_list=np.linspace(0.1,0.2,20)
for item in new_list:
    print (item)
```

```
0.1
0.10526315789473685
0.1105263157894737
0.11578947368421053
0.12105263157894737
0.12631578947368421
0.13157894736842107
0.1368421052631579
0.14210526315789473
0.1473684210526316
0.15263157894736842
0.15789473684210525
0.1631578947368421
0.16842105263157897
0.17368421052631577
0.17894736842105263
0.1842105263157895
0.18947368421052632
0.19473684210526315
0.2
```

Answer Exercise 6

In [6]:
```
#Setting my lists
Months=['January','February','March','April','May','June','July'\
        ,'August','September','October','November','December']
Days=[31,28,31,30,31,30,31,31,30,31,30,31]

#Method 1
for i in np.arange(12):
    print (Months[i],' has ',Days[i],' days')

#Methods 2 zip
print ('\nI am going to use a trick\n')
#this is a trick for going through 2 lists.
for i,j in zip(Months,Days):
    print (i,' has ',j,' days')

#Method 3, Enumerate
```

```
print ('\nI am going to use a second trick\n')
#this is a trick for going through 2 lists.
for i,j in enumerate(Months):
    print (j,' has ',Days[i],' days')

#Method 3 printed nicer
print ('\nNpw with format to make it look nice\n')
#this is a trick for going through 2 lists.
for i,j in enumerate(Months):
    print( '{:9} has {} days'.format(j,Days[i]))
```

```
January  has  31  days
February  has  28  days
March  has  31  days
April  has  30  days
May  has  31  days
June  has  30  days
July  has  31  days
August  has  31  days
September  has  30  days
October  has  31  days
November  has  30  days
December  has  31  days

I am going to use a trick

January  has  31  days
February  has  28  days
March  has  31  days
April  has  30  days
May  has  31  days
June  has  30  days
July  has  31  days
August  has  31  days
September  has  30  days
October  has  31  days
November  has  30  days
December  has  31  days

I am going to use a second trick

January  has  31  days
February  has  28  days
March  has  31  days
April  has  30  days
May  has  31  days
June  has  30  days
July  has  31  days
August  has  31  days
September  has  30  days
October  has  31  days
November  has  30  days
December  has  31  days

Npw with format to make it look nice

January   has 31 days
February  has 28 days
March     has 31 days
April     has 30 days
May       has 31 days
June      has 30 days
July      has 31 days
August    has 31 days
```

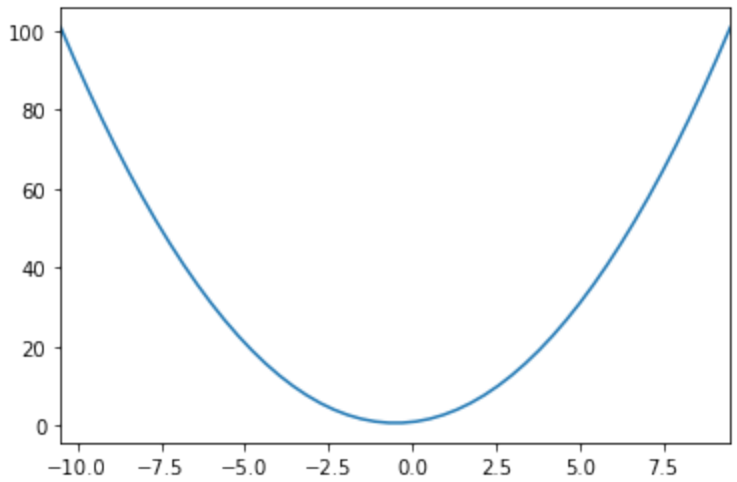
```
September has 30 days
October   has 31 days
November  has 30 days
December  has 31 days
```

Answer exercise 7

In [17]:
```
import matplotlib.pylab as plt
a=1
b=1
c=1
vertex=-b/(2*a) # I got fancy to center the parabola
x_distance=10
x=np.linspace(vertex-x_distance,vertex+x_distance)
y=[]
for i in x:
    y.append(a*i**2+b*i+c)

fig,ax=plt.subplots()
ax.plot(x,y)
ax.set_xlim(vertex-x_distance,vertex+x_distance) #centering the parabolia
```

Out[17]: (-10.5, 9.5)



Answer Exercise 8

In [37]:
```
sum=0
mylist=np.linspace(0.2,0.3,10)
for i in mylist:
    sum+=i
print ('the sume is=',sum)
print ('the mean is=',sum/len(mylist))
```

```
the sume is= 2.4999999999999996
the mean is= 0.24999999999999994
```

In [ ]: