Welcome to an introductory Jupyter notebook.

We'll use Jupyter notebooks for simple, interactive database examples.

This sample illustrates what you can do with a Jupyter notebook and checks that you've installed everything correctly.

In [70]:
```
from numpy import *
import scipy.linalg
import scipy
```

In [74]:
```
a=array([1,2,3])
a.ndim
```

Out[74]: 1

In [75]:
```
size(a)
```

Out[75]: 3

In [76]:
```
shape(a)
```

Out[76]: (3,)

In [77]:
```
a.shape[1-1]
```

Out[77]: 3

In [78]:
```
array([[1.,2.,3.], [4.,5.,6.]])
```

Out[78]:
```
array([[ 1.,  2.,  3.],
       [ 4.,  5.,  6.]])
```

In [79]:
```
b=a
c=a
d=a
vstack([hstack([a,b]), hstack([c,d])])
```

Out[79]:
```
array([[1, 2, 3, 1, 2, 3],
       [1, 2, 3, 1, 2, 3]])
```

In [80]:
```
a[-1]
```

Out[80]: 3

In [81]:
```
a=vstack([hstack([a,b]), hstack([c,d])])
a[0,2]
```

Out[81]: 3

In [82]:
```
a=array([1,2,3,4,5,6])
a[1] or a[1,:]
```

Out[82]:
```
2
```

In [83]:
```
a[0:5]
```

Out[83]:
```
array([1, 2, 3, 4, 5])
```

In [84]:
```
        a[-5:]
```

Out[84]:
```
array([2, 3, 4, 5, 6])
```

In [85]:
```
a=array([1,2,3,4,5,6])
b=a
c=a
d=a
a=vstack([hstack([a,b]), hstack([c,d]),hstack([c,d]),hstack([c,d])])
print a
a[0:3][:,1:2]
```

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

Out[85]:
```
array([[2],
       [2],
       [2]])
```

In [86]:
```
        a[ix_([1,2,3],[0,2])]
```

Out[86]:
```
array([[1, 3],
       [1, 3],
       [1, 3]])
```

In [87]:
```
a[ 2:21:2,:]
```

Out[87]:
```
array([[1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]])
```

In [88]:
```
a[ ::2,:]
```

Out[88]:
```
array([[1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]])
```

In [89]:
```
a[ ::-1,:]
```

Out[89]:
```
array([[1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]])
```

In [90]:
```
a[r_[:len(a),0]]
```

Out[90]:
```
array([[1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6],
       [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]])
```

In [91]:
```
        a.transpose()
```

Out[91]:
```
array([[1, 1, 1, 1],
       [2, 2, 2, 2],
       [3, 3, 3, 3],
       [4, 4, 4, 4],
       [5, 5, 5, 5],
       [6, 6, 6, 6],
       [1, 1, 1, 1],
       [2, 2, 2, 2],
       [3, 3, 3, 3],
       [4, 4, 4, 4],
       [5, 5, 5, 5],
       [6, 6, 6, 6]])
```

In [92]:
```
a.conj().transpose()
```

Out[92]:
```
array([[1, 1, 1, 1],
       [2, 2, 2, 2],
       [3, 3, 3, 3],
       [4, 4, 4, 4],
       [5, 5, 5, 5],
       [6, 6, 6, 6],
       [1, 1, 1, 1],
       [2, 2, 2, 2],
       [3, 3, 3, 3],
       [4, 4, 4, 4],
       [5, 5, 5, 5],
       [6, 6, 6, 6]])
```

In [93]:
```
a.dot(a.T)
```

Out[93]:
```
array([[182, 182, 182, 182],
       [182, 182, 182, 182],
       [182, 182, 182, 182],
       [182, 182, 182, 182]])
```

In [94]:
```
b=a
a * b
```

Out[94]:
```
array([[ 1,  4,  9, 16, 25, 36,  1,  4,  9, 16, 25, 36],
       [ 1,  4,  9, 16, 25, 36,  1,  4,  9, 16, 25, 36],
       [ 1,  4,  9, 16, 25, 36,  1,  4,  9, 16, 25, 36],
       [ 1,  4,  9, 16, 25, 36,  1,  4,  9, 16, 25, 36]])
```

In [95]:
```
a/b
```

Out[95]:
```
array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])
```

In [96]:
```
a**3
```

Out[96]:
```
array([[  1,   8,  27,  64, 125, 216,   1,   8,  27,  64, 125, 216],
       [  1,   8,  27,  64, 125, 216,   1,   8,  27,  64, 125, 216],
       [  1,   8,  27,  64, 125, 216,   1,   8,  27,  64, 125, 216],
       [  1,   8,  27,  64, 125, 216,   1,   8,  27,  64, 125, 216]])
```

In [97]:
```
(a>0.5)
```

Out[97]:
```
array([[ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True,  True,  True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True,  True,  True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True,  True,  True],
       [ True,  True,  True,  True,  True,  True,  True,  True,  True,
         True,  True,  True]], dtype=bool)
```

In [98]:
```
nonzero(a>0.5)
```

Out[98]:
```
(array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
 1, 1,
        1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3,
        3, 3]),
 array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11,  0,  1,  2,  3,
  4,
         5,  6,  7,  8,  9, 10, 11,  0,  1,  2,  3,  4,  5,  6,  7,  8,
  9,
        10, 11,  0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]))
```

In [99]:
```
v=a-1
a[:,nonzero(v>0.5)[0]]
```

Out[99]:
```
array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
 3,
        3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
 3,
        3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
 3,
        3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4],
       [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
 3,
        3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4]])
```

In [104]:
```
a=array([0,2,3,4,5,6])
#v=array([[0.1],[0.6],[0.7],[0.7],[0.7],[0.7]])
v=array([0,2,3,4,5,5])
a[:,v.T>0.5]
```

```
---------------------------------------------------------------------
----
IndexError                                Traceback (most recent call l
ast)
<ipython-input-104-123a716c8bd4> in <module>()
      2 #v=array([[0.1],[0.6],[0.7],[0.7],[0.7],[0.7]])
      3 v=array([0,2,3,4,5,5])
----> 4 a[:,v.T>0.5]

IndexError: too many indices for array
```

In [101]:
```
a[a<0.5]=0
```

In [102]:
```
a * (a>0.5)
```

Out[102]:
```
array([0, 2, 3, 4, 5, 6])
```

In [83]:
```
a[:] = 3
```

In [84]:
```
x=a
y = x.copy()
```

In [85]:
```
y = x[1,:].copy()
```

In [86]:
```
y = x.flatten()
```

In [87]:
```
arange(1.,11.)
```

Out[87]:
```
array([  1.,   2.,   3.,   4.,   5.,   6.,   7.,   8.,   9.,  10.])
```

In [88]:
```
arange(10.)
```

Out[88]:
```
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.])
```

In [89]:
```
arange(1.,11.)[:, newaxis]
```

Out[89]:
```
array([[  1.],
       [  2.],
       [  3.],
       [  4.],
       [  5.],
       [  6.],
       [  7.],
       [  8.],
       [  9.],
       [ 10.]])
```

In [90]:
```
zeros((3,4))
```

Out[90]:
```
array([[ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.]])
```

In [91]:
```
zeros((3,4,5))
```

Out[91]:
```
array([[[ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.]],

       [[ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.]],

       [[ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.],
        [ 0.,  0.,  0.,  0.,  0.]]])
```

In [92]:
```
ones((3,4))
```

Out[92]:
```
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
```

In [93]:
```
eye(3)
```

Out[93]:
```
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

In [94]:
```
diag(a)
```

Out[94]:
```
array([3, 3, 3])
```

In [95]:
```
diag(a,0)
```

Out[95]:
```
array([3, 3, 3])
```

In [96]:
```
random.rand(3,4)
```

Out[96]:
```
array([[ 0.56588064,  0.63953079,  0.53923796,  0.88577848],
       [ 0.71038199,  0.03737614,  0.24133596,  0.66912884],
       [ 0.47153706,  0.97800525,  0.69947967,  0.41360096]])
```

In [97]:
```
linspace(1,3,4)
```

Out[97]:
```
array([ 1.        ,  1.66666667,  2.33333333,  3.        ])
```

In [98]:
```
mgrid[0:9.,0:6.]
```

Out[98]:
```
array([[[ 0.,  0.,  0.,  0.,  0.,  0.],
        [ 1.,  1.,  1.,  1.,  1.,  1.],
        [ 2.,  2.,  2.,  2.,  2.,  2.],
        [ 3.,  3.,  3.,  3.,  3.,  3.],
        [ 4.,  4.,  4.,  4.,  4.,  4.],
        [ 5.,  5.,  5.,  5.,  5.,  5.],
        [ 6.,  6.,  6.,  6.,  6.,  6.],
        [ 7.,  7.,  7.,  7.,  7.,  7.],
        [ 8.,  8.,  8.,  8.,  8.,  8.]],

       [[ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.],
        [ 0.,  1.,  2.,  3.,  4.,  5.]]])
```

In [99]:
```
ogrid[0:9.,0:6.]
```

Out[99]:
```
[array([[ 0.],
        [ 1.],
        [ 2.],
        [ 3.],
        [ 4.],
        [ 5.],
        [ 6.],
        [ 7.],
        [ 8.]]), array([[ 0.,  1.,  2.,  3.,  4.,  5.]])]
```

In [100]:
```
meshgrid([1,2,4],[2,4,5])
```

Out[100]:
```
[array([[1, 2, 4],
        [1, 2, 4],
        [1, 2, 4]]), array([[2, 2, 2],
        [4, 4, 4],
        [5, 5, 5]])]
```

In [101]:
```
ix_([1,2,4],[2,4,5])
```

Out[101]:
```
(array([[1],
        [2],
        [4]]), array([[2, 4, 5]]))
```

In [102]:
```
m=2
n=3
tile(a, (m, n))
```

Out[102]:
```
array([[3, 3, 3, 3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3, 3, 3, 3]])
```

In [107]:
```
b=a
concatenate((a,b),1)
```

Out[107]:
```
array([[3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3],
       [3, 3, 3, 3, 3, 3]])
```

In [108]:
```
concatenate((a,b))
```

Out[108]:
```
array([[3, 3, 3],
       [3, 3, 3],
       [3, 3, 3],
       [3, 3, 3],
       [3, 3, 3],
       [3, 3, 3]])
```

In [109]:
```
a.max()
```

Out[109]:
```
3
```

In [110]:
```
        a.max(0)
```

Out[110]:
```
array([3, 3, 3])
```

In [111]:
```
        a.max(1)
```

Out[111]:
```
array([3, 3, 3])
```

In [112]:
```
        maximum(a, b)
```

Out[112]:
```
array([[3, 3, 3],
       [3, 3, 3],
       [3, 3, 3]])
```

In [113]:
```
        sqrt(dot(v,v))
```

Out[113]:
```
array([[ 1.5       ,  2.59807621,  3.35410197],
       [ 1.5       ,  2.59807621,  3.35410197],
       [ 1.5       ,  2.59807621,  3.35410197]])
```

In [114]:
```
logical_and(a,b)
```

Out[114]:
```
array([[ True,  True,  True],
       [ True,  True,  True],
       [ True,  True,  True]], dtype=bool)
```

In [115]:
```
        logical_or(a,b)
```

Out[115]:
```
array([[ True,  True,  True],
       [ True,  True,  True],
       [ True,  True,  True]], dtype=bool)
```

In [116]:
```
a & b
```

Out[116]:
```
array([[3, 3, 3],
       [3, 3, 3],
       [3, 3, 3]])
```

In [117]:
```
a | b
```

Out[117]:
```
array([[3, 3, 3],
       [3, 3, 3],
       [3, 3, 3]])
```

In [120]:
```
a=array([[1,2,3],[2,3,4],[3,4,5]])
linalg.inv(a)
```

Out[120]:
```
array([[  1.35107989e+16,  -2.70215978e+16,   1.35107989e+16],
       [ -2.70215978e+16,   5.40431955e+16,  -2.70215978e+16],
       [  1.35107989e+16,  -2.70215978e+16,   1.35107989e+16]])
```

In [121]:
```
        linalg.pinv(a)
```

Out[121]:
```
array([[ -1.08333333e+00,  -1.66666667e-01,   7.50000000e-01],
       [ -1.66666667e-01,   3.26128013e-16,   1.66666667e-01],
       [  7.50000000e-01,   1.66666667e-01,  -4.16666667e-01]])
```

In [122]:
```
linalg.matrix_rank(a)
```

Out[122]:
```
2
```

In [123]:
```
linalg.solve(a,b)
```

Out[123]:
```
array([[ 3.,  3.,  3.],
       [-9., -9., -9.],
       [ 6.,  6.,  6.]])
```

In [38]:
```
a=array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
U, S, Vh = linalg.svd(a)
V = Vh.T
print v
```

```
[[ 0.1]
 [ 0.6]
 [ 0.7]
 [ 0.7]
 [ 0.7]
 [ 0.7]]
```

In [39]:
```
a=array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
linalg.cholesky(a).T
```

Out[39]:
```
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

In [45]:
```
b=array([[ 3.,  3.,  3.],
       [-9., -9., -9.],
       [ 6.,  6.,  6.]])
a=array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
V,D = linalg.eig(a,b)
```

```
---------------------------------------------------------------------
----
TypeError                                 Traceback (most recent call l
ast)
<ipython-input-45-c4cb723e08a1> in <module>()
      5        [0, 1, 0],
      6        [0, 0, 1]])
----> 7 V,D = linalg.eig(a,b)

TypeError: eig() takes exactly 1 argument (2 given)
```

In [47]:
```
Q,R = scipy.linalg.qr(a)
```

In [48]:
```
L,U = scipy.linalg.lu(a)
```

```
---------------------------------------------------------------------
----
ValueError                                Traceback (most recent call l
ast)
<ipython-input-48-6f2aa08dcebc> in <module>()
----> 1 L,U = scipy.linalg.lu(a)

ValueError: too many values to unpack
```

In [49]:
```
```

```
---------------------------------------------------------------------
----
AttributeError                            Traceback (most recent call l
ast)
<ipython-input-49-8717a5e877a7> in <module>()
----> 1 scipy.sparse.linalg.cg

AttributeError: 'module' object has no attribute 'sparse'
```

In [52]:
```
a=[1,2,3]
fft.fft(a)
```

Out[52]:
```
array([ 6.0+0.j       , -1.5+0.8660254j, -1.5-0.8660254j])
```

In [53]:
```
fft.ifft(a)
```

Out[53]:
```
array([ 2.0+0.j        , -0.5-0.28867513j, -0.5+0.28867513j])
```

In [54]:
```
sort(a)
```

Out[54]:
```
array([1, 2, 3])
```

In [59]:
```
i=1
a=array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
I = argsort(a[:,i])
b=a[I,:]
```

In [62]:
```
X=array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
y=X
linalg.lstsq(X,y)
```

Out[62]:
```
(array([[ 1.,  0.,  0.],
        [ 0.,  1.,  0.],
        [ 0.,  0.,  1.]]), array([], dtype=float64), 3, array([ 1.,
 1.,  1.]))
```

In [66]:
```
scipy.signal.resample(x, len(x)/q)
```

```
---------------------------------------------------------------------
----
AttributeError                            Traceback (most recent call l
ast)
<ipython-input-66-e1b412d4ce6c> in <module>()
----> 1 scipy.signal.resample(x, len(x)/q)

AttributeError: 'module' object has no attribute 'signal'
```

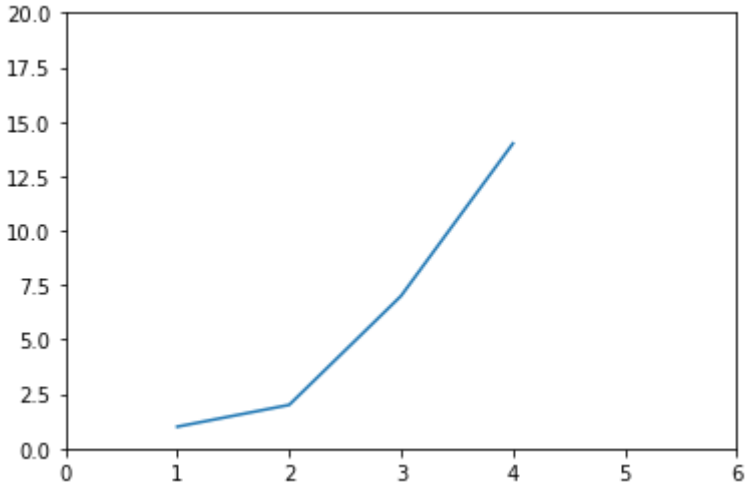
In [24]:
```
unique(a)
```

Out[24]:
```
array([0, 2, 3, 4, 5, 6])
```

In [25]:
```
a.squeeze()
```

Out[25]:
```
array([0, 2, 3, 4, 5, 6])
```

In [28]:
```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,2,7,14])
plt.axis([0, 6, 0, 20])
plt.show()
```

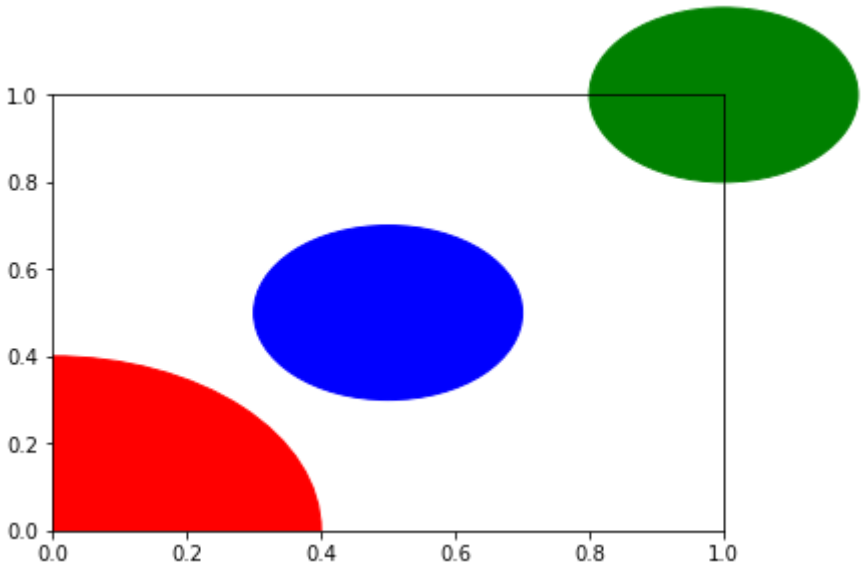


In [32]:
```
import matplotlib.pyplot as plt

circle1 = plt.Circle((0, 0), 0.4, color='r')
circle2 = plt.Circle((0.5, 0.5), 0.2, color='blue')
circle3 = plt.Circle((1, 1), 0.2, color='g', clip_on=False)

fig, ax = plt.subplots() # note we must use plt.subplots, not plt.subplo
t
# (or if you have an existing figure)
# fig = plt.gcf()
# ax = fig.gca()

ax.add_artist(circle1)
ax.add_artist(circle2)
ax.add_artist(circle3)
plt.show()
```



In [ ]:
```
Task 5: Paste your VCS account into your report.

https://github.com/xun6000
```