Ipong, Jose Martin R.

Exercise: Simple arrays

In [44]:
```
import numpy as np
```

In [45]:
```
a = np.array([4, 5, 6, 7])
```

In [46]:
```
a.ndim
```

Out[46]: 1

In [47]:
```
len(a)
```

Out[47]: 4

In [48]:
```
a.shape
```

Out[48]: (4,)

In [49]:
```
b = np.array([[6, 7, 8], [9, 10, 11]])
```

In [50]:
```
b.ndim
```

Out[50]: 2

In [51]:
```
len(b)
```

Out[51]: 2

In [52]:
```
b.shape
```

Out[52]: (2, 3)

Exercise: Creating arrays using functions

In [53]:
```
c = np.arange(6,22,3)
```

In [54]:
```
c = np.linspace(5,10,8)
```

In [55]:
```
c = np.ones(2,2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)

<ipython-input-55-2c1b760c424a> in <module>()
----> 1 c = np.ones(2,2)

/usr/lib/python2.7/dist-packages/numpy/core/numeric.pyc in ones(shape, dtype, order)
    176
    177     """
--> 178     a = empty(shape, dtype, order)
    179     multiarray.copyto(a, 1, casting='unsafe')
    180     return a

TypeError: data type not understood
```

In [56]:
```
c = np.zeros(2,2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
```

```
<ipython-input-56-c5a7ec536c53> in <module>()
----> 1 c = np.zeros(2,2)

TypeError: data type not understood
```

In [57]:
```
c = np.eye(4)
```

In [58]:
```
c = np.diag(np.array([5,6,7,8]))
```

In [59]:
```
c = np.random.rand(4)
```

In [60]:
```
c = np.random.rand(5)*100
```

In [61]:
```
np.random.seed(3444)
```

In [62]:
```
c = np.random.rand(5)*100
```

In [63]:
```
print c
```

```
[ 87.91850915  17.47777221  41.82172511  88.02620437  53.84828808]
```

np.empty() returns an empty array. It is useful for generating arrays that would be automatically filled as the program runs

Exercise: Simple Visualizations

In [64]:
```
import matplotlib.pyplot as plt
```

In [65]:
```
x = np.linspace(0, 5, 5)
y = np.linspace(0, 5, 5)
plt.plot(x, y)
plt.show()
```

In [66]:
```
image = np.random.rand(50, 50)
plt.imshow(image, cmap=plt.cm.hot)
plt.colorbar()
plt.show()
```

In [67]:
```
image = np.random.rand(50, 50)
plt.imshow(image, cmap=plt.cm.binary)
plt.colorbar()
plt.show()
```

Exercise: Indexing and slicing

In [68]:
```
a = np.arange(20)
a
```

Out[68]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

In [69]:
```
a[11:19:2] # [start:end:step]
```

Out[69]:
```
array([11, 13, 15, 17])
```

In [70]:
```
a[5:6]
```

Out[70]:
```
array([5])
```

In [71]:
```
a[::5]
```

Out[71]:
```
array([ 0,  5, 10, 15])
```

In [72]:
```
a[17:]
```

Out[72]:
```
array([17, 18, 19])
```

In [73]:
```
import numpy as np
np.arange(6) + np.arange(0, 51, 10)[:, np.newaxis]
```

Out[73]:
```
array([[ 0,  1,  2,  3,  4,  5],
       [10, 11, 12, 13, 14, 15],
       [20, 21, 22, 23, 24, 25],
       [30, 31, 32, 33, 34, 35],
       [40, 41, 42, 43, 44, 45],
       [50, 51, 52, 53, 54, 55]])
```

In [74]:
```
b = np.array([[1,2],[3,4]])
b
```

Out[74]:
```
array([[1, 2],
       [3, 4]])
```

In [75]:
```
c = np.array([[5,6,7],[8,9,10],[11,12,13]])
c
```

Out[75]:
```
array([[ 5,  6,  7],
       [ 8,  9, 10],
       [11, 12, 13]])
```

In [76]:
```
b[2::] = c[::-1]
b
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-76-6239593788e8> in <module>()
----> 1 b[2::] = c[::-1]
      2 b

ValueError: could not broadcast input array from shape (3,3) into shape (0,2)
```

In []:
```
b = np.arange(5)
b = b[::-2]
b
```

Exercise: Array creation

In []:
```
a = np.array([[1,1,1,1],[1,1,1,2],[1,1,1,2],[1,6,1,1]])
a
```

In []:
```
c = np.array([[0.,0.,0.,0.,0.,0.],[0.,2.,0.,0.,0.,0.],[0.,0.,3.,0.,0.,0.],[0.,0.,0.,4.,0.,0.],[0.,0.,0.,0.,5.,0.],[0.,0.,0.,0.,0.,6.]])
c
```

Exercise: Tiling for array creation

In []:
```
d = np.array([[4,3],[2,1]])
np.tile(d,(2,3))
```

Exercise: Fancy indexing

In []:
```
a = np.arange(10)
idx = np.array([[3, 4], [9, 7]])
idx.shape
a[idx]
```