Ipong, Jose Martin R.

Exercise: Elementwise operations

In [2]:
```
import numpy as np
```

In [7]:
```
a = np.array([3,4,5,6,7])
b = np.ones(5) + 1
a - b
```

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

In [10]:
```
a = np.arange(100000)
%timeit a + 5
```

```
10000 loops, best of 3: 158 µs per loop
10000 loops, best of 3: 142 µs per loop
```

In [9]:
```
l = range(100000)
%timeit [i+5 for i in l]
```

```
100 loops, best of 3: 5.45 ms per loop
100 loops, best of 3: 5.97 ms per loop
```

In [11]:
```
a = np.array([[1,2],[3,4]])
b = np.array([[2,2],[2,2]])
a.dot(b)
```

Out[11]:
```
array([[ 6,  6],
       [14, 14]])
```

In [13]:
```
a = np.array([2**0, 2**1, 2**2, 2**3, 2**4])
a
```

Out[13]:
```
array([ 1,  2,  4,  8, 16])
```

In [15]:
```
a_j = 2^(3*a) - a
```

Exercise other operations

numpy.allclose() is useful for comparing two numpy arrays.

Transpose is a view. An implication would be that a symmetric array would be the same after transposition.

Exercise: Reductions

I would expect to see a difference function.

Sum adds numbers in an array and returns only a single value which is the total sum while cumsum returns a cumulative sum, it returns an array of the cumulative sums of the array that is entered as a parameter.

Exercise: Shape manipulations

Ravel returns a copy while flatten returns a view.

In [16]:
```
a = np.array([[1,2],[3,4],[5,6]])
a
```

Out[16]:
```
array([[1, 2],
```

```
       [3, 4],
       [5, 6]])
```

In [20]:
```
b = np.transpose(a)
b
```

Out[20]:
```
array([[1, 3, 5],
       [2, 4, 6]])
```

Exercise: Sorting

In [23]:
```
a = np.array([2,6,4,5,8,3])
a.sort()
a
```

Out[23]:
```
array([2, 3, 4, 5, 6, 8])
```

In [24]:
```
b = np.sort(a)
b
```

Out[24]:
```
array([2, 3, 4, 5, 6, 8])
```

In [33]:
```
c = np.array(['c','d','b','a'])
c.sort()
c
```

Out[33]:
```
array(['a', 'b', 'c', 'd'],
      dtype='|S1')
```

In [34]:
```
d = np.array([5.67,2.83,2.45,8.23])
d.sort()
d
```

Out[34]:
```
array([ 2.45,  2.83,  5.67,  8.23])
```

In [36]:
```
a = np.array([[2,1,3], [5,4,6]])
b = a.ravel()
b.sort()
b = b.reshape((2, 3))
b
```

Out[36]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [38]:
```
a = np.array([[2,1,3], [5,4,6]])
b = np.sort(a, axis=1)
b
```

Out[38]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In []: