(http://www.pieriandata.com)

NumPy Exercises

Now that we've learned about NumPy let's test your knowledge. We'll start off with a few simple tasks, and then you'll be asked some more complicated questions.

Import NumPy as np ¶

```
In [4]: import numpy as np
```

Create an array of 10 zeros

```
In [2]: np.zeros(10)
Out[2]: array([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])
```

Create an array of 10 ones

```
In [3]: np.ones(10)
Out[3]: array([ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.])
```

Create an array of 10 fives

```
In [13]: np.ones(10) * 5
Out[13]: array([ 5.,  5.,  5.,  5.,  5.,  5.,  5.,  5.,  5.,  5.])
```

Create an array of the integers from 10 to 50

In [14]:
```
np.arange(10,51)
```

Out[14]:
```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
       27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
       44, 45, 46, 47, 48, 49, 50])
```

Create an array of all the even integers from 10 to 50

In [15]:
```
np.arange(10,51,2)
```

Out[15]:
```
array([10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42,
       44, 46, 48, 50])
```

Create a 3x3 matrix with values ranging from 0 to 8

In [7]:
```
np.arange(0,9).reshape(3,3)
```

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

Create a 3x3 identity matrix

In [8]:
```
np.eye(3)
```

Out[8]:
```
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

Use NumPy to generate a random number between 0 and 1

In [11]:
```
np.random.rand(1)
```

Out[11]:
```
array([ 0.6205948])
```

Use NumPy to generate an array of 25 random numbers sampled from a standard normal distribution

In [33]:
```
np.random.randn(25).reshape(5,5)
```

Out[33]:
```
array([ 1.32031013,  1.6798602 , -0.42985892, -1.53116655,  0.85753232,
        0.87339938,  0.35668636, -1.47491157,  0.15349697,  0.99530727,
       -0.94865451, -1.69174783,  1.57525349, -0.70615234,  0.10991879,
       -0.49478947,  1.08279872,  0.76488333, -2.3039931 ,  0.35401124,
       -0.45454399, -0.64754649, -0.29391671,  0.02339861,  0.38272124])
```

Create the following matrix:

np.random.rand(100).reshape(10,10)

In [8]:
```
np.random.rand(20).reshape(5,4)
```

Out[8]:
```
array([[ 0.45731013,  0.26211963,  0.39434792,  0.64561167],
       [ 0.92084843,  0.84069706,  0.87552548,  0.81642548],
       [ 0.27858059,  0.59952001,  0.63227753,  0.75526053],
       [ 0.0231313 ,  0.2156344 ,  0.28668429,  0.68872444],
       [ 0.43614747,  0.66520281,  0.31426385,  0.64203766]])
```

Create an array of 20 linearly spaced points between 0 and 1:

Numpy Indexing and Selection

Now you will be given a few matrices, and be asked to replicate the resulting matrix outputs:

In [9]:
```
mat = np.arange(1,26).reshape(5,5)
mat
```

Out[9]:
```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20],
       [21, 22, 23, 24, 25]])
```

In [39]:
```
# WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

In [40]:
```
mat[2:,1:]
```

Out[40]:
```
array([[12, 13, 14, 15],
       [17, 18, 19, 20],
       [22, 23, 24, 25]])
```

In [29]:
```
# WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

In [41]:
```
mat[3,4]
```

Out[41]:
```
20
```

In [30]:
```
# WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

In [67]:
```
mat[:4,1]
```

Out[67]:
```
array([ 2,  7, 12, 17])
```

In [60]:
```
# WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

In [46]:
```
mat[4]
```

Out[46]:
```
array([21, 22, 23, 24, 25])
```

In [32]:
```
# WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

In [41]:
```
mat[3:,:]
```

Out[41]:
```
array([[16, 17, 18, 19, 20],
       [21, 22, 23, 24, 25]])
```

Now do the following

Get the sum of all the values in mat

In [50]:
```
np.sum(mat)
```

Out[50]:
```
325
```

Get the standard deviation of the values in mat

In [22]:
```
np.std(mat)
```

Out[22]:
```
7.2111025509279782
```

Get the sum of all the columns in mat

In [49]:
```
mat[:5,0].sum()
```

Out[49]:
```
55
```

Great Job!

In [ ]: