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 [121]:

```
#Uday_21BAI1373

import numpy as np
```

Create an array of 10 zeros ¶

In [122]:

```
zeros_array = np.zeros(10)
zeros_array
```

Out[122]:

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

Create an array of 10 ones

In [123]:

```
ones_array = np.ones(10)
ones_array
```

Out[123]:

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Create an array of 10 fives

In [124]:

```
fives_array = 5 * np.ones(10)
fives_array
```

Out[124]:

```
array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])
```

Create an array of the integers from 10 to 50

In [125]:

```
#Uday_21BAI1373

integers_array = np.arange(10, 51)
integers_array
```

Out[125]:

```
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 [126]:

```
even_integers_array = np.arange(10, 51, 2)
even_integers_array
```

Out[126]:

```
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 [127]:

```
#Uday_21BAI1373

matrix = np.arange(9).reshape(3, 3)
matrix
```

Out[127]:

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

Create a 3x3 identity matrix

In [128]:

```
identity_matrix = np.eye(3)
identity_matrix
```

Out[128]:

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

Use NumPy to generate a random number between 0 and 1

In [129]:

```
#Uday_21BAI1373

random_number = np.random.rand()

print(random_number)
```

```
0.7239331234468815
```

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

In [130]:

```
random_numbers = np.random.randn(25)

random_numbers
```

Out[130]:

```
array([-0.52387687,  0.37539146,  0.27356959,  2.04104
531,  0.57324139,
       -1.70911924, -0.14611303, -0.5965386 ,  0.25641
731,  0.18951955,
        0.61578559, -0.0104123 ,  0.11652538,  0.44129
299,  0.7157297 ,
       -0.50366065,  0.80025118, -1.65916486, -0.72406
63 ,  0.7023415 ,
       -1.13426731, -1.26129678, -3.03469159,  0.06315
02 ,  0.13223111])
```

Create the following matrix:

In [131]:

```
numbers_array = np.arange(0.01, 1.01, 0.01)

matrix = numbers_array.reshape(10, 10)

print(matrix)
```

```
[[0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ]
 [0.11 0.12 0.13 0.14 0.15 0.16 0.17 0.18 0.19 0.2 ]
 [0.21 0.22 0.23 0.24 0.25 0.26 0.27 0.28 0.29 0.3 ]
 [0.31 0.32 0.33 0.34 0.35 0.36 0.37 0.38 0.39 0.4 ]
 [0.41 0.42 0.43 0.44 0.45 0.46 0.47 0.48 0.49 0.5 ]
 [0.51 0.52 0.53 0.54 0.55 0.56 0.57 0.58 0.59 0.6 ]
 [0.61 0.62 0.63 0.64 0.65 0.66 0.67 0.68 0.69 0.7 ]
 [0.71 0.72 0.73 0.74 0.75 0.76 0.77 0.78 0.79 0.8 ]
 [0.81 0.82 0.83 0.84 0.85 0.86 0.87 0.88 0.89 0.9 ]
 [0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.  ]]
```

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

In [132]:

```
linear_space = np.linspace(0, 1, 20)

print(linear_space)
```

```
[0.         0.05263158 0.10526316 0.15789474 0.2105263
2 0.26315789
 0.31578947 0.36842105 0.42105263 0.47368421 0.5263157
9 0.57894737
 0.63157895 0.68421053 0.73684211 0.78947368 0.8421052
6 0.89473684
 0.94736842 1.        ]
```

Numpy Indexing and Selection

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

In [133]:

```
mat = np.arange(1,26).reshape(5,5)
mat
```

Out[133]:

```
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 [134]:

```
# 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 [135]:

```
#Uday_21BAI1373

mat[2:, 1:]
```

Out[135]:

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

In [136]:

```
# 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 [137]:

```
print(mat[3, 4])
```

```
20
```

In [138]:

```
# 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 [139]:

```
mat = np.arange(1,26).reshape(5,5)

output = mat[0:3, 1].reshape(3,1)

output
```

Out[139]:

```
array([[ 2],
       [ 7],
       [12]])
```

In [140]:

```
# 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 [141]:

```
mat[4,:]
```

Out[141]:

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

In [142]:

```
# 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 [143]:

```
mat[3:, 0:]
```

Out[143]:

```
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 [144]:

```
#Uday_21BAI1373

sum_of_all_values = np.sum(mat)

print(sum_of_all_values)
```

```
325
```

Get the standard deviation of the values in mat

In [145]:

```
standard_deviation = np.std(mat)

print(standard_deviation)
```

```
7.211102550927978
```

Get the sum of all the columns in mat

In [146]:

```
#Uday_21BAI1373

mat = np.arange(1,26).reshape(5,5)

sum_of_all_columns = np.sum(mat, axis=0)

print(sum_of_all_columns)
```

```
[55 60 65 70 75]
```

Type *Markdown* and LaTeX: α^2