In [ ]:
```
PrajyotPatil
```

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.

In [ ]:
```

```

Import NumPy as np

In [1]:
```
import numpy as np
```

Create an array of 10 zeros

In [24]:
```
import numpy as np
np.zeros(10)
```

Out[24]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

Create an array of 10 ones

In [23]:
```
import numpy as np
np.ones(10)
```

Out[23]:
```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Create an array of 10 fives

In [22]:
```
import numpy as np
np.ones(10) * 5
```

Out[22]:
```
array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])
```

Create an array of the integers from 10 to 50

In [25]:
```
import numpy as np
np.arange(10,51)
```

Out[25]:
```
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 [7]:
```
array = np.arange(10,51,2),
print(array)
```

```
(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 [26]:
```
import numpy as np
np.arange(0,9).reshape((3,3))
```

Out[26]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

Create a 3x3 identity matrix

In [27]:
```
import numpy as np
np.eye(3)
```

Out[27]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

Use NumPy to generate a random number between 0 and 1

In [10]:
```
random_numbers_array = np.random.rand()
print(random_numbers_array)
```

```
0.8420385292387754
```

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

In [28]:
```
import numpy as np
np.linspace(0,1,20)
```

Out[28]:
```
array([0.        , 0.05263158, 0.10526316, 0.15789474, 0.21052632,
       0.26315789, 0.31578947, 0.36842105, 0.42105263, 0.47368421,
       0.52631579, 0.57894737, 0.63157895, 0.68421053, 0.73684211,
       0.78947368, 0.84210526, 0.89473684, 0.94736842, 1.        ])
```

Create the following matrix:

In [12]:
```
import numpy as np
values = np.arange(0.01, 1.01, 0.01)
matrix = values.reshape(10, 10),
print(matrix)
```

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

In [13]:
```
import numpy as np
values = np.arange(0, 1.00000001, 0.01010101)
matrix = values.reshape(10, 10)
print(matrix)
```

```
[[0.         0.01010101 0.02020202 0.03030303 0.04040404 0.05050505
  0.06060606 0.07070707 0.08080808 0.09090909]
 [0.10101010  0.11111111 0.12121212 0.13131313 0.14141414 0.15151515
  0.16161616 0.17171717 0.18181818 0.19191919]
 [0.2020202  0.21212121 0.22222222 0.23232323 0.24242424 0.25252525
  0.26262626 0.27272727 0.28282828 0.29292929]
 [0.3030303  0.31313131 0.32323232 0.33333333 0.34343434 0.35353535
  0.36363636 0.37373737 0.38383838 0.39393939]
 [0.4040404  0.41414141 0.42424242 0.43434343 0.44444444 0.45454545
  0.46464646 0.47474747 0.48484848 0.49494949]
 [0.5050505  0.51515151 0.52525252 0.53535353 0.54545454 0.55555555
  0.56565656 0.57575757 0.58585858 0.59595959]
 [0.6060606  0.61616161 0.62626262 0.63636363 0.64646464 0.65656565
  0.66666666 0.67676767 0.68686868 0.69696969]
 [0.7070707  0.71717171 0.72727272 0.73737373 0.74747474 0.75757575
  0.76767676 0.77777777 0.78787878 0.79797979]
 [0.8080808  0.81818181 0.82828282 0.83838383 0.84848484 0.85858585
  0.86868686 0.87878787 0.88888888 0.89898989]
 [0.9090909  0.91919191 0.92929292 0.93939393 0.94949494 0.95959595
  0.96969696 0.97979797 0.98989898 0.99999999]]
```

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

In [14]:
```
linear_spaced_array = np.linspace(0, 1, 20),
print(linear_spaced_array)
```

```
(array([0.        , 0.05263158, 0.10526316, 0.15789474, 0.21052632,
       0.26315789, 0.31578947, 0.36842105, 0.42105263, 0.47368421,
       0.52631579, 0.57894737, 0.63157895, 0.68421053, 0.73684211,
       0.78947368, 0.84210526, 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 [32]:
```
import numpy as np
mat = np.arange(1,26).reshape(5,5)
mat
```

Out[32]:
```
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]:
```
import numpy as np
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 [33]:
```
import numpy as np
mat[3,4]
```

Out[33]:
```
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 [34]:
```
import numpy as np
mat[0:3,1:2]
```

Out[34]:
```
array([[ 2],
       [ 7],
       [12]])
```

In [31]:
```
# 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 [35]:
```
import numpy as np
mat[4]
```

Out[35]:
```
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 [36]:
```
import numpy as np
mat[3:]
```

Out[36]:
```
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 [37]:
```
mat.sum()
```

Out[37]:
```
325
```

Get the standard deviation of the values in mat

In [38]:
```
mat.std()
```

Out[38]:
```
7.211102550927978
```

Get the sum of all the columns in mat

In [39]:
```
mat.sum(axis=0)
```

Out[39]:
```
array([55, 60, 65, 70, 75])
```

In [ ]:
```
PrajyotPatil
```

Great Job!