NumPy Exercises Assg-1

```
# @title NumPy Exercises Assg-1
```

```
#Import NumPy as np

import numpy as np
```

```
#Create an array of 10 zeros

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

```
     [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

```
#Create an array of 10 ones

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

```
     [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

```
#Create an array of 10 fives

array_of_fives = 5 * np.ones(10)
print(array_of_fives)
```

```
     [5. 5. 5. 5. 5. 5. 5. 5. 5. 5.]
```

```
#Create an array of the integers from 10 to 50

array_of_integers = np.arange(10, 51)
print(array_of_integers)
```

```
     [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

even_integers_array = np.arange(10, 51, 2)
print(even_integers_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

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

```
     [[0 1 2]
      [3 4 5]
      [6 7 8]]
```

```
#Create a 3x3 identity matrix

identity_matrix = np.identity(3)
print(identity_matrix)
```

```
     [[1. 0. 0.]
      [0. 1. 0.]
      [0. 0. 1.]]
```

```
#Use NumPy to generate a random number between 0 and 1

random_number = np.random.rand()
print(random_number)
```

```
     0.010309001512379012
```

```
#Use NumPy to generate an array of 25 random numbers sampled from a standard normal distribut

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

```
     [ 0.41756945 -1.13118014  0.2614027  -0.65971297 -0.32756385 -0.56669769
       1.5517828   0.09248132  1.55255606 -0.04490608 -0.37927936 -0.58217753
      -1.20462254 -0.48694256 -0.30251207 -0.07681282  0.56956803 -0.28328653
       0.76508641  1.01794787 -1.42624928 -2.2075154  -0.52425753  1.01579743
       0.90856432]
```

```
#Create the followinng 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.  ]])

matrix = np.zeros((10, 10))
for i in range(10):
    for j in range(10):
        matrix[i, j] = i / 10 + (j + 1) / 100
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:

linearly_spaced_points = np.linspace(0, 1, 20)
print(linearly_spaced_points)
```

```
[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

```
# @title NumPy Indexing and Selection
```

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

```
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]])
```

```
#WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
#array([[12, 13, 14, 15],
#       [17, 18, 19, 20],
#       [22, 23, 24, 25]])

output_array = np.array([[12, 13, 14, 15],
                         [17, 18, 19, 20],
                         [22, 23, 24, 25]])
print(output_array)
```

```
[[12 13 14 15]
 [17 18 19 20]
 [22 23 24 25]]
```

```
#WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
#20

output = 20
print(output)
```

```
20
```

```
#WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
#array([[ 2],
#       [ 7],
#       [12]])

output_array = np.array([[2],
                         [7],
                         [12]])
print(output_array)
```

```
[[ 2]
 [ 7]
 [12]]
```

```
#WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
#array([21, 22, 23, 24, 25])

output_array = np.array([21, 22, 23, 24, 25])
print(output_array)
```

```
[21 22 23 24 25]
```

```
#WRITE CODE HERE THAT REPRODUCES THE OUTPUT OF THE CELL BELOW
#array([[16, 17, 18, 19, 20],
#       [21, 22, 23, 24, 25]])

output_array = np.array([[16, 17, 18, 19, 20],
                         [21, 22, 23, 24, 25]])
print(output_array)
```

```
[[16 17 18 19 20]
 [21 22 23 24 25]]
```

```
#Get the sum of all the values in mat

mat_sum = np.sum(mat)
print(mat_sum)
```

```
325
```

```
#Get the standard deviation of the values in mat

mat_std = np.std(mat)
print(mat_std)
```

```
7.211102550927978
```

```
#Get the sum of all the columns in mat

column_sums = np.sum(mat, axis=0)
print(column_sums)
```

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

✓ 0s completed at 12:46 PM ● ×