Name: Dhinesh Kandra

University and campus: VIT Vellore

Branch: BTECH CSE with Specialization in Data Science

Registration number: 21bds0030

Course name: Smart Bridge – Artificial Intelligence & Machine Learning in collaboration with Google

Batch: Morning

Exercise 1: Numpy

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 [ ]: import numpy as np
```

Create an array of 10 zeros

```
In [ ]: Array_of_10_zeros = np.zeros(10)
        Array_of_10_zeros

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

Create an array of 10 ones

```
In [ ]: Array_of_10_ones = np.ones(10)
        Array_of_10_ones

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

Create an array of 10 fives

```
In [ ]: Array_of_10_fives = np.ones(10) * 5
        Array_of_10_fives

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

Create an array of the integers from 10 to 50

```
In [ ]: integers_array_from_10_to_50 = np.arange(10, 51)
        integers_array_from_10_to_50

Out[ ]: 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 [ ]: even_integers_array_from_10_to_50 = np.arange(10, 51, 2)
        even_integers_array_from_10_to_50

Out[ ]: 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 [ ]: matrix_3x3_from_0_to_8 = np.arange(9).reshape(3, 3)
        matrix_3x3_from_0_to_8

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

Create a 3x3 identity matrix

```
In [ ]: identity_matrix_3x3 = np.eye(3)
        identity_matrix_3x3

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

Use NumPy to generate a random number between 0 and 1

```
In [ ]: random_number_between_0_and_1 = np.random.rand()
        random_number_between_0_and_1

Out[ ]: 0.593846003640236
```

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

```
In [ ]: random_numbers_generator_as_instructed = np.random.randn(25)
        random_numbers_generator_as_instructed

Out[ ]: array([-0.72819147,  0.03434961,  0.39840005,  0.52330798,  0.24903082,
                0.82236374,  0.1636888 ,  1.15979086,  0.43523016,  0.09733887,
                0.12510476,  1.49162313, -1.07670934, -0.08710647,  1.2326144 ,
               -0.53459582, -0.67154897,  0.5821499 , -1.67920898, -0.08936521,
               -0.69768957,  0.09020073,  0.39467317,  1.04676277,  0.09537276])
```

Create the following matrix:

```
In [ ]: given_matrix = np.arange(1, 101).reshape(10, 10) / 100
        given_matrix

Out[ ]: 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.  ]])
```

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

```
In [ ]: linear_spaced_array_as_instructed = np.linspace(0, 1, 20)
        linear_spaced_array_as_instructed
```

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

```
Out[ ]: 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 [ ]: # 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 [ ]: submatrix = mat[2:, 1:]
        submatrix
```

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

```
In [ ]: # 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 [ ]: nbr_20 = mat[3, 4]
        nbr_20
```

```
Out[ ]: 20
```

```
In [ ]: # 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 [ ]: submatrix = mat[:3, 1:2]
        submatrix
```

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

```
In [ ]: # 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 [ ]: Output_array_2 = mat[4, :]
        Output_array_2
```

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

```
In [ ]: # 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 [ ]: Output_array_3 = mat[3:5, :]
        Output_array_3
```

```
Out[ ]: 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 [ ]: sum_of_values_in_the_matrix = np.sum(mat)
        sum_of_values_in_the_matrix
```

```
Out[ ]: 325
```

Get the standard deviation of the values in mat

```
In [ ]: standard_deviation_in_the_matrix = np.std(mat)
        standard_deviation_in_the_matrix
```

```
Out[ ]: 7.211102550927978
```

Get the sum of all the columns in mat

```
In [ ]: sum_of_columns_in_the_matrix = np.sum(mat, axis=0)
        sum_of_columns_in_the_matrix
```

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