

PERECHARLA BHAVANA

21BCE9184

bhavana.21bce9184@vitapstudent.ac.in

#ASSIGNMENT-01

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

```
import numpy as np
```

Create an array of 10 zeros

```
a1=np.zeros(10)
a1
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]
```

Create an array of 10 ones

```
a2=np.ones(10)
a2
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1.]
```

Create an array of 10 fives

```
a3=np.full(10,5.0)
a3
array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.]
```

Create an array of the integers from 10 to 50

```
a4=np.arange(10,51)
a4
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

```
a5=np.arange(10,51,2)
a5
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

```
a6=np.array([[0,1,2],[3,4,5],[6,7,8]])
a6
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

Create a 3x3 identity matrix

```
a7=np.eye(3)
a7
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

Use NumPy to generate a random number between 0 and 1

```
a8=np.random.rand()
a8
0.46020269143777315
```

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

```
a9=np.random.randn(25)
a9
array([ 0.35337472, -0.61735381,  0.91874252, -1.28820112,
        1.36947164,  0.92876771, -0.68645589,  0.94001046,  1.29781284,
        1.71179631, -0.49081021, -0.2658486 , -0.11079775,  0.00758062, -
        1.50186169, -1.56060843,  1.00522642, -0.33542725, -0.63625249,
        1.48159084,
```

```
0.06441293, 1.34651447, 1.4128547 , -0.43377127, -  
0.80344635])
```

Create the following matrix:

```
a10=np.arange(0.01,1.0,0.01)  
a10  
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])
```

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

```
b1=np.linspace(0,1,20)  
b1  
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:

```
b2 = np.arange(1,26).reshape(5,5)  
b2  
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
# BE CAREFUL NOT TO RUN THE CELL BELOW, OTHERWISE YOU WON'T
# BE ABLE TO SEE THE OUTPUT ANY MORE
```

```
b2[2:6,1:6]
```

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

```
b2[3:4,4:6]
```

```
array([[20]])
```

```
# 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
```

```
b2[0:3,1:2]
```

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

```
# 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
```

```
b2[4:6,0:6]
```

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

```
b2[3:6,0:6]
```

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

Now do the following Get the sum of all the values in mat

```
s1=np.sum(b2)
```

```
s1
```

```
325
```

```
sd=np.std(b2)
```

```
sd
```

```
7.211102550927978
```

Get the sum of all the columns in mat

```
c1=np.sum(b2,axis=0)
c1
array([55, 60, 65, 70, 75])
```