Importing numpy library

```
In [8]: import numpy as np
```

How to check the version

```
In [11]: np.__version__

Out[11]: '1.26.4'
```

Creating Arrays

```
In [19]: l = [0,1,2,3,4,5] # this also creates an array but in form of list, we want it i
         l

Out[19]: [0, 1, 2, 3, 4, 5]

In [23]: type(l) # as we can see the type is list

Out[23]: list

In [29]: a = np.array(l) # here we are converting the list to an array, using the array f
         a

Out[29]: array([0, 1, 2, 3, 4, 5])

In [34]: type(a) # now we can see that the type says ndarray

Out[34]: numpy.ndarray
```

In-Built Functions

1) Arrange Function

- This function is used to print numbers in-between the given parameter range.

- If there is no start value given, the default value is 0.

- Always the left value should be greater than the right value.

- It can only take 3 parameters

```
In [42]: np.arange(5)
```

```
Out[42]: array([0, 1, 2, 3, 4])
```

```
In [46]: np.arange(3,10)
```

```
Out[46]: array([3, 4, 5, 6, 7, 8, 9])
```

```
In [50]: np.arange(10,20)
```

```
Out[50]: array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

```
In [52]: np.arange(20,10)
```

```
Out[52]: array([], dtype=int32)
```

```
In [54]: np.arange(-20,10)
```

```
Out[54]: array([-20, -19, -18, -17, -16, -15, -14, -13, -12, -11, -10,  -9,  -8,
                 -7,  -6,  -5,  -4,  -3,  -2,  -1,   0,   1,   2,   3,   4,   5,
                  6,   7,   8,   9])
```

```
In [190… # STEP COUNT
```

```
In [64]: np.arange(10,30,5) # this prints the values from starting point 10 to ending poi
```

```
Out[64]: array([10, 15, 20, 25])
```

```
In [68]: np.arange(20,40,3) # this prints the values from starting point 20 to ending poi
```

```
Out[68]: array([20, 23, 26, 29, 32, 35, 38])
```

2) Zeros Function

- This function is used to print '0' values in the form of rows and columns

- By default it prints in float data type

```
In [72]: np.zeros(2)
```

```
Out[72]: array([0., 0.])
```

In [188…
```
# PRINT THE VALUES IN INT DATA TYPE
```

In [74]:
```
np.zeros(5,dtype=int)
```

Out[74]:
```
array([0, 0, 0, 0, 0])
```

In [80]:
```
np.zeros(7,dtype=int)
```

Out[80]:
```
array([0, 0, 0, 0, 0, 0, 0])
```

In [178…
```
# PRINT THE VALUES IN THE FORM OF ROWS AND COLUMNS
```

In [88]:
```
np.zeros((10,20)) # right = rows & left = columns
```

Out[88]:
```
array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.]])
```

In [92]:
```
np.zeros((10,20),dtype=int) # prints the values in int data type
```

Out[92]:
```
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

2) Ones Function

- This function is used to print '1' value in the form of rows and columns

- By default it prints in float data type

```
In [97]: np.ones(5)

Out[97]: array([1., 1., 1., 1., 1.])

In [186…: # PRINT THE VALUES IN INT DATA TYPE

In [101…: np.ones(5,dtype=int)

Out[101…: array([1, 1, 1, 1, 1])

In [176…: # PRINT THE VALUES IN THE FORM OF ROWS AND COLUMNS

In [105…: np.ones((10,20))

Out[105…: array([[1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.],
                [1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
                 1., 1., 1., 1.]])

In [184…: # PRINT THE VALUES IN INT DATA TYPE

In [113…: np.ones((10,20),dtype=int)

Out[113…: array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])
```

4) Rand Function

- This function is used to print random values in the given parameter range

- By default it prints in float data type

In [117…
```
np.random.rand(5)
```
Out[117…
```
array([0.59727119, 0.79196472, 0.47381981, 0.82679948, 0.28165967])
```

In [130…
```
np.random.rand(8)
```
Out[130…
```
array([0.731875  , 0.85412199, 0.13240791, 0.85467504, 0.72098898,
       0.52427139, 0.93812497, 0.88270683])
```

In [174…
```
# PRINT THE VALUES IN THE FORM OF ROWS AND COLUMNS
```

In [136…
```
np.random.rand(5,10)
```
Out[136…
```
array([[0.99701445, 0.604679  , 0.66245259, 0.76780933, 0.0825743 ,
        0.91584385, 0.00923835, 0.47935937, 0.07700348, 0.98416253],
       [0.70528301, 0.65397497, 0.84961914, 0.23702242, 0.55085741,
        0.71423256, 0.57402881, 0.03228762, 0.88177617, 0.18382838],
       [0.07365755, 0.05647315, 0.48040081, 0.86918701, 0.00807736,
        0.31111854, 0.80429396, 0.86191459, 0.99138848, 0.44553371],
       [0.19383207, 0.93800025, 0.14238045, 0.83329656, 0.57726333,
        0.75136829, 0.41668952, 0.4641046 , 0.87975676, 0.42625816],
       [0.86695274, 0.39975061, 0.73886408, 0.13796584, 0.6221803 ,
        0.11741976, 0.87424078, 0.06784886, 0.81362802, 0.09953479]])
```

5) Randint Function

- This function is used to print random values in the given parameter range

In [146…
```
np.random.randint(5) # this prints random values from 0 - 4 (n-1->5-1=4)
```
Out[146…
```
4
```

In [164…
```
# PRINT THE VALUES IN THE GIVEN RANGE
```

In [150…
```
np.random.randint(10,20)
```
Out[150…
```
17
```

In [154…
```
np.random.randint(20,40)
```
Out[154…
```
24
```

In [156…
```
np.random.randint(10,40,4) # prints 4 random values in between the range 10-39 (
Out[156… array([23, 27, 16, 32])
```

```
In [160… np.random.randint(10,21,3) # prints 3 random values in between the range 10-20 (
```

```
Out[160… array([15, 20, 13])
```

```
In [162… # PRINT THE VALUES IN THE FORM OF ROWS AND COLUMNS
```

```
In [170… np.random.randint(10,20,(4,4)) # prints random values in bwetween the range 10-1
```

```
Out[170… array([[15, 14, 15, 18],
                [14, 17, 11, 11],
                [17, 12, 12, 10],
                [14, 14, 19, 19]])
```

```
In [172… np.random.randint(10,40,(10,10)) # prints random values in bwetween the range 10
```

```
Out[172… array([[13, 23, 22, 27, 39, 35, 29, 24, 16, 37],
                [21, 14, 29, 20, 34, 15, 39, 27, 34, 19],
                [17, 14, 16, 10, 16, 36, 12, 27, 11, 26],
                [15, 14, 15, 30, 10, 20, 36, 16, 11, 23],
                [16, 36, 15, 21, 36, 15, 30, 30, 29, 39],
                [35, 27, 34, 28, 17, 32, 20, 20, 22, 35],
                [39, 10, 27, 39, 17, 37, 37, 18, 16, 33],
                [17, 32, 14, 15, 22, 37, 11, 37, 28, 16],
                [25, 23, 21, 24, 25, 13, 21, 29, 18, 32],
                [13, 20, 13, 37, 28, 19, 38, 14, 29, 35]])
```

Indexing and Slicing

1) Slicing

```
In [195… a = np.random.randint(10,20,(5,4))
         a
```

```
Out[195… array([[16, 17, 14, 11],
                [16, 17, 10, 18],
                [10, 17, 14, 14],
                [14, 12, 16, 10],
                [13, 17, 10, 18]])
```

```
In [278… a[:] # prints all the rows and columns
```

```
Out[278… array([[16, 17, 14, 11],
                [16, 17, 10, 18],
                [10, 17, 14, 14],
                [14, 12, 16, 10],
                [13, 17, 10, 18]])
```

```
In [280… a[1:5] # prints the rows and columns from 1 to 4 (n-1->5-1=4)
```

```
Out[280… array([[16, 17, 10, 18],
                [10, 17, 14, 14],
                [14, 12, 16, 10],
                [13, 17, 10, 18]])
```

In [282…
```
a
```

Out[282…
```
array([[16, 17, 14, 11],
       [16, 17, 10, 18],
       [10, 17, 14, 14],
       [14, 12, 16, 10],
       [13, 17, 10, 18]])
```

In [284…
```
a[0:-1] # prints the rows and columns from 0 to -2 (n-1->-1-1=-2)
```

Out[284…
```
array([[16, 17, 14, 11],
       [16, 17, 10, 18],
       [10, 17, 14, 14],
       [14, 12, 16, 10]])
```

In [205…
```
a
```

Out[205…
```
array([[16, 17, 14, 11],
       [16, 17, 10, 18],
       [10, 17, 14, 14],
       [14, 12, 16, 10],
       [13, 17, 10, 18]])
```

In [219…
```
a1 = np.random.randint(0,100,(10,10))
a1
```

Out[219…
```
array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
       [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
       [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
       [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
       [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
       [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
       [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
       [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
       [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
       [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

In [286…
```
a1[:] # prints all the rows and columns
```

Out[286…
```
array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
       [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
       [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
       [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
       [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
       [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
       [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
       [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
       [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
       [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

In [288…
```
a1[0:5] # prints the rows and columns from 1 to 4 (n-1->5-1=4)
```

Out[288…
```
array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
       [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
       [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
       [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
       [65, 84, 45, 96, 63, 45, 31, 76, 80, 26]])
```

In [229…
```
a1
```

```
Out[229… array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
               [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
               [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
               [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
               [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
               [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
               [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
               [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
               [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
               [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

```
In [290… a1[::-1] # prints the rows and columns in reverse form with 1 step
```

```
Out[290… array([[ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17],
               [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
               [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
               [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
               [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
               [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
               [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
               [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
               [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
               [99, 83, 41, 90, 49, 43, 33, 35, 45, 73]])
```

```
In [255… a1
```

```
Out[255… array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
               [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
               [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
               [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
               [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
               [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
               [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
               [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
               [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
               [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

```
In [292… a1[::-2] # prints the rows and columns in reverse form with 2 step
```

```
Out[292… array([[ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17],
               [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
               [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
               [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
               [27, 52,  5, 97, 57, 37, 67, 61, 67, 62]])
```

```
In [259… a1
```

```
Out[259… array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
               [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
               [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
               [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
               [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
               [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
               [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
               [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
               [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
               [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

```
In [294… a1[::-3] # prints the rows and columns in reverse form with 3 step
```

```
Out[294…  array([[ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17],
                 [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
                 [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
                 [99, 83, 41, 90, 49, 43, 33, 35, 45, 73]])
```

2) Indexing

```
In [303…  a1
```

```
Out[303…  array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
                 [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
                 [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
                 [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
                 [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
                 [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
                 [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
                 [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
                 [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
                 [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

```
In [305…  a1[0,2] # returns the value in the oth row and 2nd column
```

```
Out[305…  41
```

```
In [319…  a1[1,5] # returns the value in the 1st row and 5th column
```

```
Out[319…  37
```

```
In [329…  a1[-5,5] # returns the value in the -5th row and 5th column
```

```
Out[329…  22
```

```
In [323…  a1[-5,-5] # returns the value in the -5th row and -5th column
```

```
Out[323…  22
```

```
In [325…  a1
```

```
Out[325…  array([[99, 83, 41, 90, 49, 43, 33, 35, 45, 73],
                 [27, 52,  5, 97, 57, 37, 67, 61, 67, 62],
                 [74, 28, 57, 74, 88, 64, 14, 86, 19, 73],
                 [52, 23, 88, 17, 48, 94, 57,  1, 89, 24],
                 [65, 84, 45, 96, 63, 45, 31, 76, 80, 26],
                 [23, 50, 30, 41, 98, 22, 26,  9,  4,  4],
                 [95, 89, 80, 38, 84, 43, 29, 74, 31, 55],
                 [63, 26, 68, 31, 58, 59, 83, 96, 40, 13],
                 [44, 52, 60, 46, 70, 94, 96, 62, 37, 12],
                 [ 5, 81, 51,  8, 96, 26, 73,  3, 61, 17]])
```

```
In [327…  a1[-1,-2] # returns the value in the -1st row and -2nd column
```

```
Out[327…  61
```

```
In [399…  # INDEXING USING VARIABLES
```

In [405…
```
mat = np.arange(0,100).reshape(10,10)
mat
```

Out[405…
```
array([[ 0,  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, 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, 51, 52, 53, 54, 55, 56, 57, 58, 59],
       [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
       [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
       [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
       [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

In [407…
```
row = 4
col = 5
```

In [409…
```
mat[row,col]
```

Out[409…
```
45
```

In [411…
```
mat[:]
```

Out[411…
```
array([[ 0,  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, 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, 51, 52, 53, 54, 55, 56, 57, 58, 59],
       [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
       [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
       [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
       [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

In [ ]:
```
# HOW TO PRINT ONLY COLUMNS OF A MATRIX
```

In [413…
```
mat[:,col]
```

Out[413…
```
array([ 5, 15, 25, 35, 45, 55, 65, 75, 85, 95])
```

In [415…
```
# HOW TO PRINT ONLY ROWS OF A MATRIX
```

In [417…
```
mat[row,:]
```

Out[417…
```
array([40, 41, 42, 43, 44, 45, 46, 47, 48, 49])
```

In [419…
```
# HOW TO PRINT ROWS AND COLUMNS OF A MATRIX
```

In [421…
```
mat
```

```
Out[421…  array([[ 0,  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, 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, 51, 52, 53, 54, 55, 56, 57, 58, 59],
                 [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
                 [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
                 [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
                 [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

```
In [423…  mat[2:6,2:4]
```

```
Out[423…  array([[22, 23],
                 [32, 33],
                 [42, 43],
                 [52, 53]])
```

```
In [425…  mat[1:2,2:4]
```

```
Out[425…  array([[12, 13]])
```

```
In [427…  mat[2:3,2:3]
```

```
Out[427…  array([[22]])
```

```
In [429…  mat[3:5,2:4]
```

```
Out[429…  array([[32, 33],
                 [42, 43]])
```

```
In [431…  mat[2:3,4:5]
```

```
Out[431…  array([[24]])
```

Operations

```
In [332…  a2 = np.array(l)
          a2
```

```
Out[332…  array([0, 1, 2, 3, 4, 5])
```

1) Maximum Function

```
In [336…  # PRINTS THE MAXIMUM/HIGHEST ELEMENT IN THE ARRAY
```

```
In [338…  a2.max()
```

```
Out[338…  5
```

2) Minimum Function

In [340…
```
# PRINTS THE MINIMUM/LOWEST ELEMENT IN THE ARRAY
```

In [342…
```
a2.min()
```

Out[342…
```
0
```

3) Mean Function

In [344…
```
# PRINTING THE MEAN OF THE ELEMENTS IN THE ARRAY
```

In [346…
```
a2
```

Out[346…
```
array([0, 1, 2, 3, 4, 5])
```

In [348…
```
a2.mean()
```

Out[348…
```
2.5
```

4) Median Function

In [350…
```
# PRINTING THE MEDIAN OF THE ELEMENTS IN THE ARRAY
```

In [352…
```
a2
```

Out[352…
```
array([0, 1, 2, 3, 4, 5])
```

In [354…
```
a2.median()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[354], line 1
----> 1 a2.median()

AttributeError: 'numpy.ndarray' object has no attribute 'median'
```

In [356…
```
from numpy import *
```

In [368…
```
a3 = array([0,1,2,3,4,5])
median(a3)
```

Out[368…
```
2.5
```

5) Reshape Function

In [377…
```
a2
```

Out[377…
```
array([0, 1, 2, 3, 4, 5])
```

In [381…
```
# PRINTS THE ARRAY IN THE GIVEN MATRIX PARAMETER ( GIVEN ROWS & COLUMN PARAMETER
```

In [383…
```
a2.reshape(2,3)
```
Out[383…
```
array([[0, 1, 2],
       [3, 4, 5]])
```
In [385…
```
a2.reshape(6,1)
```
Out[385…
```
array([[0],
       [1],
       [2],
       [3],
       [4],
       [5]])
```
In [387…
```
a2.reshape(1,6)
```
Out[387…
```
array([[0, 1, 2, 3, 4, 5]])
```
In [393…
```
a2.reshape(1,7) # we are getting this error becuase we only have 6 elements so t
```
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[393], line 1
----> 1 a2.reshape(1,7)

ValueError: cannot reshape array of size 6 into shape (1,7)
```
In [397…
```
a2.reshape(3,2,order='A')
```
Out[397…
```
array([[0, 1],
       [2, 3],
       [4, 5]])
```

Masking (or) Filtering

In [435…
```
mat
```
Out[435…
```
array([[ 0,  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, 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, 51, 52, 53, 54, 55, 56, 57, 58, 59],
       [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
       [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
       [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
       [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```
In [437…
```
id(mat) # returns the address of mat
```
Out[437…
```
2915217839792
```
In [439…
```
# LESS THAN OPERATION
```
In [443…
```
mat < 50 # this prints true if values are less than 50 and false if values are g
```

```
Out[443…  array([[ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False]])
```

```
In [447…  mat <= 50 # this prints true if values are less than 50 and false if values are
```

```
Out[447…  array([[ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False]])
```

```
In [445…  # GREATER THAN OPERATION
```

```
In [449…  mat > 50 # this prints true if values are greater than 50 and false if values ar
```

```
Out[449…  array([[False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True]])
```

```
In [451…  mat >= 50 # this prints true if values are greater than 50 and false if values a
```

```
Out[451…  array([[False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [False, False, False, False, False, False, False, False, False,
                  False],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True],
                 [ True,  True,  True,  True,  True,  True,  True,  True,  True,
                   True]])
```

```
In [453…  # DOUBLE EQUAL TO OPERATION
```

```
In [457…  mat == 50 # this prints true if values is equal to 50 and false if values are no
```

Out[457…
```
array([[False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [ True, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False],
        [False, False, False, False, False, False, False, False, False,
         False]])
```

In [459…
```
# HOW TO RETURN THE VALUES IN A MATRIX, WITH USING MASKING OR FILTERING
```

In [465…
```
# PRINTS THE VALUES THAT ARE LESS THAN 50, WITH THE ACTUAL RANGE PARAMETER
```

In [467…
```
a4 = mat[mat<50]
a4
```

Out[467…
```
array([ 0,  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, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49])
```

In [469…
```
# PRINTS THE VALUES THAT ARE GREATER THAN 50, WITH THE ACTUAL RANGE PARAMETER
```

In [471…
```
a4 = mat[mat>50]
a4
```

Out[471…
```
array([51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [473…
```
# PRINTS THE VALUES THAT ARE EQUAL TO 50, WITH THE ACTUAL RANGE PARAMETER
```

In [475…
```
a4 = mat[mat==50]
a4
```

Out[475…
```
array([50])
```

In [481…
```
# PRINTS THE VALUES THAT ARE LESS THAN OR EQUAL TO 50, WITH THE ACTUAL RANGE PAR
```

In [483…
```
a4 = mat[mat<=50]
a4
```

Out[483…
```
array([ 0,  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, 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])
```

In [477…
```
# PRINTS THE VALUES THAT ARE GREATER THAN OR EQUAL TO 50, WITH THE ACTUAL RANGE
```

In [479…
```
a4 = mat[mat>=50]
a4
```

Out[479…
```
array([50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66,
       67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83,
       84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```