Experiment 5 | NumPy

```
In [2]: import numpy as np
```

```
In [3]: arr = np.arange(1,10).reshape(3,3)
        print(arr)

[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Maximum and Minimum

```
In [5]: arr.max()

Out[5]: 9
```

```
In [6]: arr.argmax()

Out[6]: 8
```

```
In [7]: arr.max(axis=0)

Out[7]: array([7, 8, 9])
```

```
In [8]: arr.max(axis=1)

Out[8]: array([3, 6, 9])
```

Sum of Matrix

```
In [10]: np.sum(arr)

Out[10]: 45
```

```
In [11]: np.sum(arr, axis=0)

Out[11]: array([12, 15, 18])
```

```
In [12]: np.sum(arr, axis=1)

Out[12]: array([ 6, 15, 24])
```

Mean, Median, & Standard Deviation

```
In [14]: np.mean(arr)
```

```
Out[14]: 5.0
```

```
In [15]: np.median(arr)

Out[15]: 5.0
```

```
In [16]: np.std(arr)

Out[16]: 2.581988897471611
```

```
In [17]: np.log(arr)

Out[17]: array([[0.        , 0.69314718, 1.09861229],
                [1.38629436, 1.60943791, 1.79175947],
                [1.94591015, 2.07944154, 2.19722458]])
```

```
In [18]: np.log10(arr)

Out[18]: array([[0.        , 0.30103   , 0.47712125],
                [0.60205999, 0.69897   , 0.77815125],
                [0.84509804, 0.90308999, 0.95424251]])
```

```
In [19]: np.exp(arr)

Out[19]: array([[2.71828183e+00, 7.38905610e+00, 2.00855369e+01],
                [5.45981500e+01, 1.48413159e+02, 4.03428793e+02],
                [1.09663316e+03, 2.98095799e+03, 8.10308393e+03]])
```

Slicing a numpy array

```
In [21]: a = np.arange(1,101).reshape(10,10)
         a

Out[21]: 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,  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, 100]])
```

```
In [22]: a[1,4]

Out[22]: 15
```

```
In [23]: a[:,6:8]
```

```
Out[23]: array([[ 7,  8],
                [17, 18],
                [27, 28],
                [37, 38],
                [47, 48],
                [57, 58],
                [67, 68],
                [77, 78],
                [87, 88],
                [97, 98]])

In [24]: a[1:3,:]

Out[24]: array([[11, 12, 13, 14, 15, 16, 17, 18, 19, 20],
                [21, 22, 23, 24, 25, 26, 27, 28, 29, 30]])

In [25]: a[6:8,:]

Out[25]: array([[61, 62, 63, 64, 65, 66, 67, 68, 69, 70],
                [71, 72, 73, 74, 75, 76, 77, 78, 79, 80]])

In [26]: a[2:4,3:5]

Out[26]: array([[24, 25],
                [34, 35]])

In [27]: a[5:8,4:7]

Out[27]: array([[55, 56, 57],
                [65, 66, 67],
                [75, 76, 77]])

In [28]: mat = np.arange(1,41).reshape(5,8)
         mat

Out[28]: 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, 26, 27, 28, 29, 30, 31, 32],
                [33, 34, 35, 36, 37, 38, 39, 40]])

In [29]: sm1 = mat[:,:4]
         sm2 = mat[:,4:]
         sum_mat = sm1 + sm2
         sum_mat

Out[29]: array([[ 6,  8, 10, 12],
                [22, 24, 26, 28],
                [38, 40, 42, 44],
                [54, 56, 58, 60],
                [70, 72, 74, 76]])

In [30]: def generate_data(data):
             return {"max":data.max(),
                     "min":data.min(),
                     "col_wise_max":data.max(axis=0),
```

```
                     "col_wise_min":data.min(axis=0),
                     "row_wise_max":data.max(axis=1),
                     "row_wise_min":data.min(axis=1)}

In [31]: generate_data(sm1)

Out[31]: {'max': 36,
          'min': 1,
          'col_wise_max': array([33, 34, 35, 36]),
          'col_wise_min': array([1, 2, 3, 4]),
          'row_wise_max': array([ 4, 12, 20, 28, 36]),
          'row_wise_min': array([ 1,  9, 17, 25, 33])}

In [32]: generate_data(sm2)

Out[32]: {'max': 40,
          'min': 5,
          'col_wise_max': array([37, 38, 39, 40]),
          'col_wise_min': array([5, 6, 7, 8]),
          'row_wise_max': array([ 8, 16, 24, 32, 40]),
          'row_wise_min': array([ 5, 13, 21, 29, 37])}
```

Random

```
In [34]: import random

In [35]: np.random.random(1)

Out[35]: array([0.84621876])

In [36]: np.random.random(5)

Out[36]: array([0.60523804, 0.77002022, 0.34487246, 0.32779005, 0.14189571])

In [37]: np.random.random((5,5))

Out[37]: array([[0.01274261, 0.96279378, 0.5875845 , 0.31545778, 0.81238997],
                [0.92198638, 0.79763951, 0.06724639, 0.09951761, 0.60742107],
                [0.36190619, 0.5909866 , 0.95425123, 0.11578233, 0.5381572 ],
                [0.12994701, 0.59872639, 0.54582951, 0.8785299 , 0.73387918],
                [0.45412489, 0.21098579, 0.34398891, 0.1827605 , 0.45072356]])

In [38]: np.random.randint(1)

Out[38]: 0

In [39]: np.random.randint(5)

Out[39]: 4

In [40]: np.random.randint(20,25,12)
```

```
Out[40]:  array([22, 20, 24, 20, 23, 20, 23, 22, 23, 20, 22, 23])

In [41]:  np.random.randint(20,25,(5,8))

Out[41]:  array([[20, 22, 21, 23, 22, 21, 22, 24],
                 [24, 20, 22, 21, 22, 20, 21, 22],
                 [20, 20, 24, 24, 24, 21, 20, 20],
                 [20, 21, 22, 22, 20, 20, 20, 21],
                 [23, 20, 22, 24, 23, 24, 23, 22]])

In [42]:  np.random.randint(20,25,(2,5,5))

Out[42]:  array([[[22, 24, 22, 21, 24],
                  [20, 24, 22, 21, 22],
                  [23, 23, 20, 24, 22],
                  [23, 21, 20, 22, 24],
                  [20, 23, 23, 22, 22]],

                 [[23, 22, 23, 21, 20],
                  [22, 24, 20, 23, 23],
                  [23, 23, 21, 22, 24],
                  [21, 20, 22, 22, 20],
                  [21, 23, 23, 21, 22]]])

In [43]:  # choice
          p = [1,5,8,9,12,4,5,63,17,58]
          np.random.choice(p)

Out[43]:  63

In [44]:  np.random.choice(p,12)

Out[44]:  array([ 5, 63,  9,  1,  5,  5,  9, 63, 63, 12, 12,  9])

In [45]:  np.random.choice(p,(5,6))

Out[45]:  array([[ 5,  9,  8,  5, 58,  9],
                 [58,  4,  9,  4, 17,  5],
                 [17, 58, 63, 17,  5, 63],
                 [ 9, 58, 58, 63,  4, 58],
                 [ 4,  1, 63, 12,  9, 12]])
```

Permutation

```
In [47]:  k = [1,2,3]
          np.random.permutation(k)

Out[47]:  array([3, 2, 1])

In [48]:  m1 = np.arange(1,17).reshape(4,4)
          m2 = np.arange(17,33).reshape(4,4)

In [49]:  m1
```

```
Out[49]:  array([[ 1,  2,  3,  4],
                 [ 5,  6,  7,  8],
                 [ 9, 10, 11, 12],
                 [13, 14, 15, 16]])

In [50]:  m2

Out[50]:  array([[17, 18, 19, 20],
                 [21, 22, 23, 24],
                 [25, 26, 27, 28],
                 [29, 30, 31, 32]])
```

Concatenate()

```
In [52]:  np.concatenate((m1,m2))

Out[52]:  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, 26, 27, 28],
                 [29, 30, 31, 32]])

In [53]:  np.concatenate((m1,m2), axis=1)

Out[53]:  array([[ 1,  2,  3,  4, 17, 18, 19, 20],
                 [ 5,  6,  7,  8, 21, 22, 23, 24],
                 [ 9, 10, 11, 12, 25, 26, 27, 28],
                 [13, 14, 15, 16, 29, 30, 31, 32]])
```

Split()

```
In [55]:  np.split(m1,2)

Out[55]:  [array([[1, 2, 3, 4],
                  [5, 6, 7, 8]]),
           array([[ 9, 10, 11, 12],
                  [13, 14, 15, 16]])]
```

Generate NumPy Data for House Price

```
In [57]:  # area, valcanie, bhk, houseprice

          valcanie = np.random.randint(1,6,100)
          area = np.random.randint(600,2000,100)
          bhk = np.random.randint(1,6,100)
          price = np.random.randint(423000,2500000,100)
          data = np.zeros((100,4),dtype=int)
```

```
In [58]: data[:,1] = valcanie
         data[:,0] = area
         data[:,2] = bhk
         data[:,3] = price
```

```
In [110… data[:10]
```

```
Out[110… array([[   1680,       4,       1,  790255],
                [   1584,       4,       4, 1479742],
                [   1668,       3,       4, 2236801],
                [   1724,       4,       3, 1813506],
                [   1207,       4,       2, 1161492],
                [   1004,       1,       3, 2053083],
                [   1613,       3,       5, 2119885],
                [   1374,       3,       1,  581890],
                [    965,       3,       2,  715463],
                [   1935,       3,       4, 2411882]])
```

```
In [ ]:
```