Import NumPy

In [1]:
```
import numpy as np
```

Creating Arrays ¶

Create a 1-Dimesnional numpy array

In [2]:
```
score=np.array([70,63,81,58,76])
print(score)
```

```
[70 63 81 58 76]
```

Create a 4*5 1-D array with elements from 0 to 19 sorted ascending

In [3]:
```
np.arange(20).reshape(4,5)
```

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

Create a 2*5 2-Dimesnional numpy array

In [4]:
```
score_height=np.array([([70,63,81,58,76]),(5.1,4.5,6.0,5.3,5.5)])
print(score_height)
```

```
[[70.  63.  81.  58.  76. ]
 [ 5.1  4.5  6.   5.3  5.5]]
```

Create a 4*5 2-Dimesnional 0s numpy array

In [5]:
```
zero=np.zeros((4,5),dtype=np.int16)
print(zero)
```

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

Create a 4*5 2-Dimesnional 1s numpy array

In [6]:
```
one=np.ones((4,5),dtype=np.int16)
print(one)
```

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

Create a 5*5 identity matrix

In [7]:
```
identity=np.eye(5,dtype=np.int16)
print(identity)
```

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

Inspecting Arrays

Find the number of elements in an array

In [8]:
```
print(score_height)
print(score_height.size)
```

```
[[70.  63.  81.  58.  76. ]
 [ 5.1  4.5  6.   5.3  5.5]]
10
```

Find the dimensions of an array

In [9]:
```
print(score_height.shape)
```

```
(2, 5)
```

Find the data type of elements in the array

In [10]:
```
print(score_height.dtype)
```

```
float64
```

Convert array to Python list

In [11]:
```
print("NumPy array : \n ",score_height)
print("\nPython list : \n",score_height.tolist())
```

```
NumPy array :
  [[70.  63.  81.  58.  76. ]
 [ 5.1  4.5  6.   5.3  5.5]]

Python list :
 [[70.0, 63.0, 81.0, 58.0, 76.0], [5.1, 4.5, 6.0, 5.3, 5.5]]
```

Adding Elements to an array

Append new elements at the end of an array

In [12]:
```
score=np.append(score,90)
print(score)
score=np.append(score,[45,59,94])
print(score)
```

```
[70 63 81 58 76 90]
[70 63 81 58 76 90 45 59 94]
```

Insert new element(s) at a specific position in an array

In [13]:
```
score=np.insert(score,3,20)
print(score)
```

```
[70 63 81 20 58 76 90 45 59 94]
```

Removing Elements from an array

Delete row on index 2 from an array

In [14]:
```
print(score)
score=np.delete(score,2,axis=0)
print("\nRemoved 76 from the original array \n",score)
```

```
[70 63 81 20 58 76 90 45 59 94]

Removed 76 from the original array
 [70 63 20 58 76 90 45 59 94]
```

Delete column on index 2 from an array

In [15]:
```
print(score_height)
score_height=np.delete(score_height,2,axis=1)
print("\nRemove element at index 2 \n",score_height)
```

```
[[70.  63.  81.  58.  76. ]
 [ 5.1  4.5  6.   5.3  5.5]]

Remove element at index 2
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]
```

NumPy Statistics functions

Sum all elements in an array using sum() function

sum() function offers better performance than manually iterating through each element

In [16]:
```
print("1. Original array \n",score_height)
print("\n2. Sum of columns \n ",np.sum(score_height,axis=0))
print("\n3. Sum of rows \n ",np.sum(score_height,axis=1))
print("\n4. Sum of all elements \n",np.sum(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Sum of columns
  [75.1 67.5 63.3 81.5]

3. Sum of rows
  [267.   20.4]

4. Sum of all elements
 287.4
```

Find max element in a numpy array

In [17]:
```
print("1. Original array \n",score_height)
print("\n2. Max of columns \n ",np.max(score_height,axis=0))
print("\n3. Max of rows \n ",np.max(score_height,axis=1))
print("\n4. Max of all elements \n",np.max(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Max of columns
  [70. 63. 58. 76.]

3. Max of rows
  [76.   5.5]

4. Max of all elements
 76.0
```

Find min element in a numpy array

In [18]:
```
print("1. Original array \n",score_height)
print("\n2. Min of columns \n ",np.min(score_height,axis=0))
print("\n3. Min of rows \n ",np.min(score_height,axis=1))
print("\n4. Min of all elements \n",np.min(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Min of columns
  [5.1 4.5 5.3 5.5]

3. Min of rows
  [58.   4.5]

4. Min of all elements
 4.5
```

Find mean of a numpy array

In [19]:
```
print("1. Original array \n",score_height)
print("\n2. Mean of columns \n ",np.mean(score_height,axis=0))
print("\n3. Mean of rows \n ",np.mean(score_height,axis=1))
print("\n4. Mean of all elements \n",np.mean(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Mean of columns
  [37.55 33.75 31.65 40.75]

3. Mean of rows
  [66.75  5.1 ]

4. Mean of all elements
 35.925
```

Find variance of a 1-Dimensional numpy array

Variance refers to a statistical measure of the spread between values in a data set. It shows how values/variables varies with respect to each other. It is the square of Standard Deviation

In [20]:
```
print("1. Original array \n",score_height)
print("\n2. Variance of columns \n ",np.var(score_height,axis=0))
print("\n3. Variance of rows \n ",np.var(score_height,axis=1))
print("\n4. Variance of all elements \n",np.var(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Variance of columns
  [1053.0025  855.5625  694.3225 1242.5625]

3. Variance of rows
  [46.6875  0.14  ]

4. Variance of all elements
 973.594375
```

Find standard deviation of a numpy array

Standard Deviation is a statistic that measures the dispersion/spread of a value with respect to the mean. A low standard deviation shows that the values are closser to the mean of the dataset, while a high standard deviation shows that the values are widely spread out.

In [21]:
```
print("1. Original array \n",score_height)
print("\n2. Standard deviation of columns \n ",np.std(score_height,axis=0))
print("\n3. Standard deviation  of rows \n ",np.std(score_height,axis=1))
print("\n4. Standard deviation  of all elements \n",np.std(score_height))
```

```
1. Original array
 [[70.  63.  58.  76. ]
 [ 5.1  4.5  5.3  5.5]]

2. Standard deviation of columns
  [32.45 29.25 26.35 35.25]

3. Standard deviation  of rows
  [6.83282518 0.37416574]

4. Standard deviation  of all elements
 31.202473860256656
```

Find correlation coefficient of a numpy array

Correlation coefficients is a measure of the strength of relationship between two variables. It shows the magnitude and direction of the relationship between two variables.

In [22]:
```
print(np.corrcoef(score_height))
```

```
[[1.         0.41070025]
 [0.41070025 1.        ]]
```

Read Data to NumPy

Read data from text file

In [23]:
```
students_data=np.loadtxt('students.txt')
print(students_data)
```

```
[[78.   5.1]
 [56.   4.5]
 [67.   5.3]
 [73.   6. ]]
```

Read data from csv file

In [24]:
```
titanic_data=np.genfromtxt('titanic.csv',delimiter=',')
print(titanic_data)
```

```
[[    nan     nan     nan ...     nan     nan     nan]
 [ 0.      3.      nan ...  1.      0.      7.25  ]
 [ 1.      1.      nan ...  1.      0.     71.2833]
 ...
 [ 0.      3.      nan ...  1.      2.     23.45  ]
 [ 1.      1.      nan ...  0.      0.     30.    ]
 [ 0.      3.      nan ...  0.      0.      7.75  ]]
```

Store Data to File

Save data to a text file

In [25]:
```
np.savetxt('students_saved_array.txt',students_data,delimiter=' ')
```

Save data to a csv file

In [26]:
```
np.savetxt('titanic_saved_array.csv',titanic_data,delimiter=',')
```

@Sammy Ongaya https://data2ml.com (https://data2ml.com) https://github.com/SammyOngaya (https://github.com/SammyOngaya) sammiongaya@gmail.com