Participant

Title= "Matric Student"

Name= "Muhammad Bin Saqib Ali"

email = "muhammad.saqib8761@gmail.com"

whatsapp = "00923470159155"

NUMPY

Importing Numpy Library

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

Creating an Array

In [ ]:
```
# Creating an Array
a= np.array(["Pizza", "Burger","Shawarma"])
a
```

Out[ ]:
```
array(['Pizza', 'Burger', 'Shawarma'], dtype='<U8')
```

In [ ]:
```
# Creating an Array
price= np.array ([100,200,300])
price
```

Out[ ]:
```
array([100, 200, 300])
```

In [ ]:
```
# checking type of variable
type(a)
```

Out[ ]:
```
numpy.ndarray
```

In [ ]:
```
# checking type of variable
type (price)
```

Out[ ]:
```
numpy.ndarray
```

In [ ]:
```
# checking length of variable
len (price)
```

Out[ ]:
```
3
```

In [ ]:
```
# checking length of variable
len (a)
```

Out[ ]:
```
3
```

In [ ]:
```
# Checking element on index 2
a [2]
```

Out[ ]:
```
'Shawarma'
```

In [ ]:
```
# Taking mean of variable
price.mean()
```

Out[ ]:
```
200.0
```

In [ ]:
```
# printing line of zero
np.zeros(6)
```

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

In [ ]:
```
# printing line of one
np.ones (8)
```

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

In [ ]:
```
np.empty(5)
```

Out[ ]:
```
array([ 1.  ,  2.75,  6.  , 10.75, 17.  ])
```

In [ ]:
```
# specifying the number of range which we have to print
np.arange (10)
```

Out[ ]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [ ]:
```
# specify range with specific jumpy or gap
np.arange (3,33,3)
```

Out[ ]:
```
array([ 3,  6,  9, 12, 15, 18, 21, 24, 27, 30])
```

In [ ]:
```
# giving line space between specific number to specific limit
np.linspace (2,10, num=5)
```

Out[ ]:
```
array([ 2.,  4.,  6.,  8., 10.])
```

In [ ]:
```
# array with specific data type (Float)
np.ones (14, dtype=np.float32)
```

Out[ ]:
```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.],
      dtype=float32)
```

In [ ]:
```
# array with specific data type (Int)
np.ones (23, dtype=np.int32)
```

Out[ ]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1], dtype=int32)
```

Array Function

In [ ]:
```
# Creating an Array
a= np.array ([10,20,30,70,90,23.7,90.3,90.38,908,983.67])
a
```

Out[ ]:
```
array([ 10.  ,  20.  ,  30.  ,  70.  ,  90.  ,  23.7 ,  90.3 ,  90.38,
       908.  , 983.67])
```

In [ ]:
```
# Checking the dimension of variable
np.ndim (a)
```

Out[ ]:
```
1
```

In [ ]:
```
# sorting a Variable in Assending order
a.sort()
a
```

Out[ ]:
```
array([ 10.  ,  20.  ,  23.7 ,  30.  ,  70.  ,  90.  ,  90.3 ,  90.38,
       908.  , 983.67])
```

In [ ]:
```
# creating an Array
b =np.array ([36,28,389,273,27,372,18.74,283.2,374,32.26])
b
```

Out[ ]:
```
array([ 36.  ,  28.  , 389.  , 273.  ,  27.  , 372.  ,  18.74, 283.2 ,
       374.  ,  32.26])
```

In [ ]:
```
# Adding 2 Arrays
c=np.concatenate((a,b))
c
```

Out[ ]:
```
array([ 10.  ,  20.  ,  23.7 ,  30.  ,  70.  ,  90.  ,  90.3 ,  90.38,
       908.  , 983.67,  36.  ,  28.  , 389.  , 273.  ,  27.  , 372.  ,
        18.74, 283.2 , 374.  ,  32.26])
```

In [ ]:
```
# sorting a Variable in Assending order
c.sort ()
c
```

Out[ ]:
```
array([ 10.  ,  18.74,  20.  ,  23.7 ,  27.  ,  28.  ,  30.  ,  32.26,
        36.  ,  70.  ,  90.  ,  90.3 ,  90.38, 273.  , 283.2 , 372.  ,
       374.  , 389.  , 908.  , 983.67])
```

2D Array

In [ ]:
```
# Creating an 2D Array
a= np.array ([[1,2,3,4],[5,6,7,8]])
a
```

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

In [ ]:
```
# Creating an 2D Array
b= np.array ([[5,6,7,8],[1,2,3,4]])
b
```

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

In [ ]:
```
# Checking dimension of an Array
np.ndim (b)
```

Out[ ]:
```
2
```

In [ ]:
```
# Adding 2 Array together
np.concatenate ((a,b))
```

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

3D Array

In [ ]:
```
# Creating 3D Array
a= np.array ([[[1,2,3,4],
               [5,6,7,8]],


              [[5,6,7,8],
              [1,2,3,4]],

              [[1,2,3,4]
               ,[5,6,7,8]]])
a
```

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

       [[5, 6, 7, 8],
        [1, 2, 3, 4]],

       [[1, 2, 3, 4],
        [5, 6, 7, 8]]])
```

In [ ]:
```
# Checking Dimension of an Array
np.ndim (a)
```

Out[ ]:
```
3
```

In [ ]:
```
# Checking a size if an Array (Number of elements in an Array)
a.size
```

Out[ ]:
```
24
```

In [ ]:
```
# Checking the shape of an Array
a.shape
```

Out[ ]:
```
(3, 2, 4)
```

In [ ]:
```
# 2 Dimensional Array
b= np.array ([[1,2,3,4],
              [5,6,7,8],
              [9,10,11,12]])
b
```

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

In [ ]:
```
# Checking Dimension of an array
np.ndim(b)
```

Out[ ]:
```
2
```

In [ ]:
```
# Checking that how many elements available in an Array
b.size
```

Out[ ]:
```
12
```

In [ ]:
```
# Checking shape of an Array
b.shape
```

Out[ ]:
```
(3, 4)
```

In [ ]:
```
# Creating an Array to specific number
a=np.arange (16)
a
```

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

In [ ]:
```
# Shaping an Array to 4 columns and 4 rows
b=a.reshape (4,4)
b
```

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

Convert 1D into 2D

In [ ]:
```
# Creating an Array to specific number
a=np.arange(9)
a
```

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

In [ ]:
```
# Checking a shape of an Array
a.shape
```

Out[ ]:
```
(9,)
```

In [ ]:
```
# Row wise 2Dimension
b=a[np.newaxis, :]
b
```

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

In [ ]:
```
# Checking a shape of an Array
b.shape
```

Out[ ]:
```
(1, 9)
```

In [ ]:
```
# Column Wise 2Dimension
c=a[:, np.newaxis,]
c
```

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

In [ ]:
```
# Checking a shape of an Array
c.shape
```

Out[ ]:
```
(9, 1)
```

In [ ]:
```
# To show some specific range of index
a[2:6]
```

Out[ ]:
```
array([2, 3, 4, 5])
```

In [ ]:
```
# Adding 8 to every element available in an Array
a+8
```

Out[ ]:
```
array([ 8,  9, 10, 11, 12, 13, 14, 15, 16])
```

In [ ]:
```
# Multiplying 2 to every element available in an Array
a*2
```

Out[ ]:
```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16])
```

In [ ]:
```
# dividing 2 to every element available in an Array
a/2
```

Out[ ]:
```
array([0. , 0.5, 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. ])
```

In [ ]:
```
# Adding all value available in an Array
a.sum ()
```

Out[ ]:
```
36
```

In [ ]:
```
# Checking mean of an whole Array
a.mean ()
```

Out[ ]:
```
4.0
```