In [2]:

```python
# import the numpy library
import numpy as np
print("Numpy lib imported")
```

```
Numpy lib imported
```

In [3]:

```python
#check the version
print(np.__version__)
```

```
1.20.3
```

In [4]:

```python
# Create 1-D Array in np
arr1=np.array([3,4,6,7])
print(arr1)
print(type(arr1))
```

```
[3 4 6 7]
<class 'numpy.ndarray'>
```

In [5]:

```python
# list- create numpy array using list
list1=[2,5,7,8]
arr2=np.array(list1)
print(arr2)
print(type(arr2))
```

```
[2 5 7 8]
<class 'numpy.ndarray'>
```

In [6]:

```python
# create a 1-D array pass float elements
list2=[2.4,4,67,9.5]
arr3=np.array(list2)
print(arr3)
print(type(arr3))
```

```
[ 2.4  4.  67.   9.5]
<class 'numpy.ndarray'>
```

In [9]:

```python
arr1=np.array([3,4.8,6,7],dtype='int')
print(arr1)
print(type(arr1))
```

```
[3 4 6 7]
<class 'numpy.ndarray'>
```

In [10]:

```python
# Create 2-D array
arr2=np.array([[5,6,7,8],[3,5,2,1]])
print(arr2)
print(type(arr2))
```

```
[[5 6 7 8]
 [3 5 2 1]]
<class 'numpy.ndarray'>
```

In [11]:

```python
#Attributes
#Shape
#ndim
#dtype
#itemsize
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
#nbytes
print(arr2.ndim)
print(arr2.shape)
print(arr2.dtype)
print(arr2.itemsize)
print(arr2.nbytes)
```

```
2
(2, 4)
int32
4
32
```

In [14]:

```
# create float type 2-D array of float type 3*3 , print all attributes
arr4=np.array([[4.4,5.6,67.6],[4.3,5.4,6.9],[4.2,5,8]])
print(arr4)
print(arr4.ndim)
print(arr4.shape)
print(arr4.dtype)
print(arr4.itemsize)
print(arr4.nbytes)
```

```
[[ 4.4  5.6 67.6]
 [ 4.3  5.4  6.9]
 [ 4.2  5.   8. ]]
2
(3, 3)
float64
8
72
```

In [15]:

```
arr3=np.array([['banana','apple123'],['pen','pencil']])
print(arr3)
print(arr3.ndim)
print(arr3.shape)
print(arr3.dtype)
print(arr3.itemsize)
print(arr3.nbytes)
```

```
[['banana' 'apple123']
 ['pen' 'pencil']]
2
(2, 2)
<U8
32
128
```

In [2]:

```
import numpy as np
distance=np.array([10,12,15,17,50])
time=np.array([0.3,0.4,0.6,0.8,0.9])
```

In [3]:

```
print(distance)
print(time)
```

```
[10 12 15 17 50]
[0.3 0.4 0.6 0.8 0.9]
```

In [5]:

```
distance+time
```

Out[5]:

```
array([10.3, 12.4, 15.6, 17.8, 50.9])
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

In [6]:
```
distance-time
```
Out[6]:
```
array([ 9.7, 11.6, 14.4, 16.2, 49.1])
```
In [7]:
```
distance*time
```
Out[7]:
```
array([ 3. ,  4.8,  9. , 13.6, 45. ])
```
In [9]:
```
distance/time
```
Out[9]:
```
array([33.33333333, 30.        , 25.        , 21.25      , 55.55555556])
```
In [10]:
```
print(distance*2)
```
```
[ 20  24  30  34 100]
```
In [11]:
```
# Access the array elements
arr3=np.array([[1.5,2,5],[2.5,3.6,4],[1.6,2,5]],dtype='int')
```
In [12]:
```
arr3
```
Out[12]:
```
array([[1, 2, 5],
       [2, 3, 4],
       [1, 2, 5]])
```
In [13]:
```
arr3[0]     # first array
```
Out[13]:
```
array([1, 2, 5])
```
In [16]:
```
arr3[1],arr3[2]
```
Out[16]:
```
(array([2, 3, 4]), array([1, 2, 5]))
```
In [18]:
```
arr3[0,0] # 0 index from 0th array
```
Out[18]:
```
1
```
In [19]:
```
arr3[1,2]
```
Out[19]:
```
4
```
In [21]:
```
#slicing
arr3[0:2,0:2]
```
Out[21]:
```
array([[1, 2],
       [2, 3]])
```
In [22]:
```
# Conditional Selestion
# extract elements of the array based the condition
```
Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[22]:

```
array([[False, False,  True],
       [False, False,  True],
       [False, False,  True]])
```

In [23]:

```
#Universal Functions
#mean
#median
#std
#variance
#sqrt
#min
#max
#floor
#ceil
#count
```

In [24]:

```
arr1=np.array([1,3,5,7,8])
```

In [25]:

```
print(np.mean(arr1))
```

```
4.8
```

In [ ]: