#reverse rows
arr2[::-1]

o/p: array([[4., 6., 2., 1.],
[9., 0., 4., 6.]])

#reverse rows and columns both

arr2[::-1, ::-1]

o/p: array([[1., 2., 6., 4.], [6., 4., 0., 9.]])

L4: Specific element extraction

arr2

o/p: array([[9., 0., 4., 6.],
[4., 6., 2., 1.]])

arr2[0, :]

o/p: array([9., 0., 4., 6.],

arr2[0:1, :]

o/p: array([[9., 0., 4., 6.]])

arr2[:2, :]

o/p: arr([9., 0., 4., 6.],
[4., 6., 2., 1.]])

```
arr2[:2,:]

o/p: array([[9., 0., 4., 6.],
            [4., 6., 2., 1.]])

arr2[:-1,:]
o/p: array([[9., 0., 4., 6.]])

# last column
    arr2[:,:3]

array([[9., 0., 4.],
       [4., 6., 2.]])

arr2[:,:-1]

array([[9., 0., 4.],
       [4., 6., 2.]])

arr2[:1,:3] #1,2

array([[9., 0., 4.]])
```

5 Basic Statistics

arr2

o/p: array([[9., 0., 4., 6.],
[4., 6., 2., 1.]])

arr2.min()
o/p: 0.0

arr2.max()
o/p: 9.0

arr2.sum()
o/p: 32.0

arr2.mean()
o/p: 4.0

np.median(arr2)
o/p: 4.0

np.average(arr2)
o/p: 4.0

# Variance
np.var(arr2)
o/p: 7.75

6. Reshaping and Flattening

arr2

o/p: array([[9., 0., 4., 6.],
[4., 6., 2., 1.]])

arr2.shape
o/p: (2,4)

arr2.reshape(4,2)
o/p: array([[9., 0.],
[4., 6.],

[4. , 6.]
[2. , 1.]])

arr2.reshape (1,8)

O/P: array ([[9., 0., 4., 6., 4., 6., 2., 1.]])

I/P: arr2.reshape (8,1)

array ([9.],
[0.],
[4.],
[6.],
[4.],
[6],
[2],
[1]])

I/P: arr.2reshape(2,2)

O/P:
Can't reshape array of size 8 into shape (2,2)

#single dimension

fl=arr2.flatten()

fl

O/P: array ([9. , 0., 4., 6., 4., 6. , 2., 1.])

I/P: fl.ndim

O/P 1

7: Random Arrays and Sequences

```
np.arange(10)
o/p: array([0,1,2,3,4,5,6,7,8,9])

np.arange(10)
o/p: array([0,1,2,3,4,5,6,7,8,9])

np.arange(2,10)
o/p: array([2,3,4,5,6,7,8,9])

np.arange(0,10,2)
o/p: array([0,2,4,6,8])

np.arange(10,0)
o/p: array([], dtype=int32)

# starting from 10 not 0 in descending order

np.arange(10,0,-1)

o/p: array([10,9,8,7,6,5,4,3,2,1])

# mention start, stop, total no of elements
# Gives equalspaces
np.linspace(1,10,3)
```

compiler don't understand

```
o/p: array([1, 5.5, 10.])

np.linspace(1,10,3)

array([1, 5.5, 10])

np.zeros([2,3,4])
o/p: array([[[0.,0.,0.,0.]
             [0.,0.,0.,0.]],

            [[0.,0.,0.,0.],
             [0.,0.,0.,0.],
             [0.,0.,0.,0.]]])
```

8 Unique Items and count

```
arr = [[1,4,5,2,2,5],
       [4,4,1,7,4,5]]

arr
o/p [[1,4,5,2,2,5], [4,4,1,7,4,5]]

I/p: u_val, count = np.unique(arr, return_counts=True)

u_val
o/p: array([1,2,4,5,7])

count
o/p: array([2,2,4,3,1], dtype=int64)
```