In [1]:
```
#broadcasting is the concept of applying ufuncs in numpy to array of different sizes
```

In [4]:
```
import numpy as np
#additon of arrays of different sizes
a=np.arange(0,5)
print("a=",a)
b=np.arange(5,10)
print("b=",b)
print("a+b=",a+b)
```

```
a= [0 1 2 3 4]
b= [5 6 7 8 9]
a+b= [ 5  7  9 11 13]
```

In [7]:
```
#adding a value(scalar) to an array using broadcasting
print("adding 5 to each element using broadcasting:",a+5)#actually happening is [0 1 2 3 4]+[5 5 5 5 5]
```

```
adding 5 to each element using broadcasting: [5 6 7 8 9]
```

In [15]:
```
a=np.array([1,2,3])
b=np.random.randint(0,10,(3,3))
print("a=",a)
print("b=",b)
print("adding a and b using broadcasting:")
print(a+b)#a being the samller array is streched so that it can be added with c
#braodcasting is considered as stretching the smaller array to match the size of the bigger array
```

```
a= [1 2 3]
b= [[7 8 8]
 [3 4 4]
 [5 2 9]]
adding a and b using broadcasting:
[[ 8 10 11]
 [ 4  6  7]
 [ 6  4 12]]
```

In [20]:
```
c=np.arange(5)[:,np.newaxis]
d=np.arange(5)[np.newaxis,:]
print("c=")
print(c)
print("d=")
print(d)
print("c+d=")
print(c+d)
```

```
c=
[[0]
 [1]
 [2]
 [3]
 [4]]
d=
[[0 1 2 3 4]]
c+d=
[[0 1 2 3 4]
 [1 2 3 4 5]
 [2 3 4 5 6]
 [3 4 5 6 7]
 [4 5 6 7 8]]
```

In [25]:
```
#rules of broadcasting arrays
#rule 1:if number of dimensions doesnt match,the sahpe with fewer dimensions is padded to the left
#rule 2:if shape of two arrays does not match in any dimension,the array with shape equal to 1 in that dimension is stretched
#rule 3:if in any dimension size does not match and is neither equal to 1 error is raised
```

In [26]:
```
x= np.ones((2, 3))
y = np.arange(3)
print("x=")
print(x)
print("y=")
print(y)
print("x+y=")
print(x+y)
```

```
x=
[[1. 1. 1.]
 [1. 1. 1.]]
y=
[0 1 2]
x+y=
[[1. 2. 3.]
 [1. 2. 3.]]
```

In [43]:
```
#centering an array
e=np.random.randint(0,10,(3,3))
print("Given array=")
print(e)
#mean along 1st row
mean_e=e.mean(axis=0)
print("Mean array=")
print(mean_e)
#creating a centered array
centered_e=e-mean_e
print("Centered array=")
print(centered_e)#mena of centered array is close to zero
```

```
Given array=
[[6 6 3]
 [9 6 2]
 [1 2 0]]
Mean array=
[5.33333333 4.66666667 1.66666667]
Centered array=
[[ 0.66666667  1.33333333  1.33333333]
 [ 3.66666667  1.33333333  0.33333333]
 [-4.33333333 -2.66666667 -1.66666667]]
```