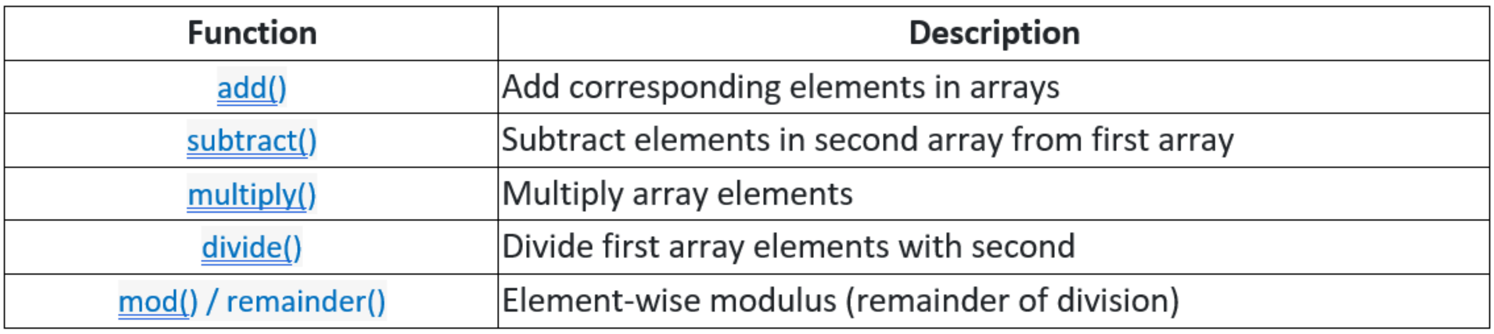
Arithmetic with NumPy Arrays

- There are specific functions in NumPy for performing arithmetic operations - addition, subtraction, multiplication, divisions and remainder.



Function	Description
add()	Add corresponding elements in arrays
subtract()	Subtract elements in second array from first array
multiply()	Multiply array elements
divide()	Divide first array elements with second
mod() / remainder()	Element-wise modulus (remainder of division)

1. **Addition Operation**

> The **add() function** sums the content of two arrays, and return the results in a new array.

2. **Subtraction Operation**

> The **subtract() function** subtracts the values from one array with the values from another array, and return the results in a new array.

3. **Multiplication Operation**

> The **multiply() function** multiplies the values from one array with the values from another array, and return the results in a new array.

4. **Division Operation**

> The **divide() function** divides the values from one array with the values from another array, and return the results in a new array.

5. **Remainder Operation**

> Both the **mod()** and **the remainder()** functions return the remainder of the values in the first array corresponding to the values in the second array, and return the results in a new array.

```
Note:
>We could use arithmetic operators + , - , * , / directly between NumPy arrays.

>But this section discusses an extension of the same where we have functions that can take any
>array-like objects e.g. lists, tuples etc.
```

Addition Operation

```
#Add the values in arr1 to the values in arr2:
import numpy as np

arr1 = np.array([10, 11, 12, 13, 14, 15])
arr2 = np.array([20, 21, 22, 23, 24, 25])

newarr = np.add(arr1, arr2)

print(newarr)
```

```
[30 32 34 36 38 40]
```

Subtraction Operation

```
#Subtract the values in arr2 from the values in arr1:
import numpy as np

arr1 = np.array([10, 20, 30, 40, 50, 60])
arr2 = np.array([20, 21, 22, 23, 24, 25])

newarr = np.subtract(arr1, arr2)

print(newarr)
```

```
[-10  -1   8  17  26  35]
```

Multiplication Operation

```
#Multiply the values in arr1 with the values in arr2:
import numpy as np

arr1 = np.array([10, 20, 30, 40, 50, 60])
arr2 = np.array([20, 21, 22, 23, 24, 25])

newarr = np.multiply(arr1, arr2)

print(newarr)
```

```
[ 200  420  660  920 1200 1500]
```

Division Operation

```
#Divide the values in arr1 with the values in arr2:
```

```
import numpy as np

arr1 = np.array([10, 20, 30, 40, 50, 60])
arr2 = np.array([3, 5, 10, 8, 2, 33])

newarr = np.divide(arr1, arr2)

print(newarr)
```

```
[ 3.33333333  4.          3.          5.         25.          1.81818182]
```

Remainder Operation

```
#Return the remainders:

import numpy as np

arr1 = np.array([10, 20, 30, 40, 50, 60])
arr2 = np.array([3, 7, 9, 8, 2, 33])

newarr = np.mod(arr1, arr2)

print(newarr)
```

```
[ 1  6  3  0  0 27]
```

You get the same result when using the remainder() function:

```
#Return the remainders:
import numpy as np

arr1 = np.array([10, 20, 30, 40, 50, 60])
arr2 = np.array([3, 7, 9, 8, 2, 33])

newarr = np.remainder(arr1, arr2)

print(newarr)
```

```
[ 1  6  3  0  0 27]
```

Using +, -, *, / operators

```
# Defining both the matrices
a = np.array([5, 72, 13, 100])
b = np.array([2, 5, 10, 30])

# Performing addition using arithmetic operator
add_ans = a+b
print(add_ans)
print("\n")


# Performing subtraction using arithmetic operator
sub_ans = a-b
print(sub_ans)
print("\n")


# Performing multiplication using arithmetic operator
mul_ans = a*b
print(mul_ans)
print("\n")

# Performing division using arithmetic operators
div_ans = a/b
print(div_ans)
print("\n")

# Performing mod
mod_ans = np.mod(a, b)
print(mod_ans)
print("\n")

#Performing remainder
rem_ans=np.remainder(a,b)
print(rem_ans)
print("\n")
```

```
    [   7  77  23 130]


    [ 3 67  3 70]


    [ 2 66  2 69]


    [  10  360  130 3000]


    [ 2.5        14.4         1.3         3.33333333]


    [ 1  2  3 10]


    [ 1  2  3 10]
```

Example: Write a Program to demonstrate arithmetic operations on NumPy arrays

```
import numpy as np
a = np.arange(9).reshape(3,3)
b = np.array([10,10,10])
```

```
print('First array:')
print(a)
print('\n')

print('Second array:' )
print(b)
print('\n')

print('Add the two arrays:' )
print(np.add(a,b))
print('\n')

print('Subtract the two arrays:' )
print(np.subtract(a,b) )
print('\n'  )

print('Multiply the two arrays:' )
print(np.multiply(a,b) )
print('\n'  )

print('Divide the two arrays:' )
print(np.divide(a,b))
```

```
First array:
[[0 1 2]
 [3 4 5]
 [6 7 8]]


Second array:
[10 10 10]


Add the two arrays:
[[10 11 12]
 [13 14 15]
 [16 17 18]]


Subtract the two arrays:
[[-10  -9  -8]
 [ -7  -6  -5]
 [ -4  -3  -2]]


Multiply the two arrays:
[[ 0 10 20]
 [30 40 50]
 [60 70 80]]


Divide the two arrays:
[[0.  0.1 0.2]
 [0.3 0.4 0.5]
 [0.6 0.7 0.8]]
```

Example: Write a Program to demonstrate arithmetic operations on NumPy arrays

```
import numpy as np
a = np.array([10,20,30])
b= np.array([1,2,3])

print("addition of a and b :",np.add(a,b))
print("multiplication of a and b :",np.multiply(a,b))
print("subtraction of a and b :",np.subtract(a,b))
print("a raised to b is:",np.power(a,b))
```

```
addition of a and b : [11 22 33]
multiplication of a and b : [10 40 90]
subtraction of a and b : [ 9 18 27]
a raised to b is: [   10   400 27000]
```

Let us take 5 arrays of different dimensions

```
a = np.array([10, 50, 100, 150, 250])

b = np.array([6, 5, 4, 3, 2])


c = np.array([[26,  48,  91,  57, 120], [33,  95,  68, 109, 155], [111, 194,   7,  22, 124], [ 82, 1

d = np.array([[12, 11,  0,  9,  7], [10,  4, 11,  6,  9], [ 9,  2, 10,  9, 11], [ 5, 14,  0, 11,  8]

e = np.array([11, 22, 33, 44, 55])

print(a)
print(a.ndim)
print("\n")
print(b)
print(b.ndim)
print("\n")
print(c)
print(c.ndim)
print("\n")
print(d)
print(d.ndim)
print("\n")
print(e)
print(e.ndim)
print("\n")
```

```
[ 10  50 100 150 250]
1


[6 5 4 3 2]
1


[[ 26  48  91  57 120]
 [ 33  95  68 109 155]
 [111 194   7  22 124]
 [ 82 119  18 156  81]
 [ 38  10 151  24  14]]
2


[[12 11  0  9  7]
 [10  4 11  6  9]
 [ 9  2 10  9 11]
 [ 5 14  0 11  8]
 [ 5 12  5  5 11]]
2


[11 22 33 44 55]
1
```

add()

```
print(np.add(a, b))
print("\n")

print(np.add(c, d))
print("\n")

print(np.add(a, c))
print("\n")

print(np.add(a, b, c))
print("\n")
```

```
[ 16  55 104 153 252]


[[ 38  59  91  66 127]
 [ 43  99  79 115 164]
 [120 196  17  31 135]
 [ 87 133  18 167  89]
 [ 43  22 156  29  25]]


[[ 36  98 191 207 370]
 [ 43 145 168 259 405]
 [121 244 107 172 374]
 [ 92 169 118 306 331]
 [ 48  60 251 174 264]]


[[ 16  55 104 153 252]
 [ 16  55 104 153 252]
 [ 16  55 104 153 252]
 [ 16  55 104 153 252]
 [ 16  55 104 153 252]]
```

subtract()

```
print(np.subtract(a, b))
print("\n")

print(np.subtract(c, d))
print("\n")

print(np.subtract(d, c))
print("\n")

print(np.subtract(a, c))
print("\n")

print(np.subtract(a, b, c))
print("\n")
```

```
[  4  45  96 147 248]


[[  4  44 104 144 245]
 [  6  51  93 147 243]
 [  7  53  94 144 241]
 [ 11  41 104 142 244]
 [ 11  43  99 148 241]]


[[  -4  -44 -104 -144 -245]
 [  -6  -51  -93 -147 -243]
 [  -7  -53  -94 -144 -241]
 [ -11  -41 -104 -142 -244]
 [ -11  -43  -99 -148 -241]]


[[-6 -5 -4 -3 -2]
 [-6 -5 -4 -3 -2]
 [-6 -5 -4 -3 -2]
 [-6 -5 -4 -3 -2]
 [-6 -5 -4 -3 -2]]


[[  4  45  96 147 248]
 [  4  45  96 147 248]
 [  4  45  96 147 248]
 [  4  45  96 147 248]
 [  4  45  96 147 248]]
```

multiply()

```
print(np.multiply(a, b))
print("\n")

print(np.multiply(c, d))
print("\n")

print(np.multiply(d, c))
print("\n")

print(np.multiply(a, c))
print("\n")

print(np.multiply(a, b, c))
print("\n")
```

```
[ 60 250 400 450 500]


[[  48  495    0 1323 1736]
 [  40  180 1056  882 2232]
 [  36   90  960 1323 2728]
 [  20  630    0 1617 1984]
 [  20  540  480  735 2728]]


[[  48  495    0 1323 1736]
 [  40  180 1056  882 2232]
 [  36   90  960 1323 2728]
 [  20  630    0 1617 1984]
 [  20  540  480  735 2728]]


[[   40  2250  9600 22050 62000]
 [   40  2250  9600 22050 62000]
 [   40  2250  9600 22050 62000]
 [   40  2250  9600 22050 62000]
 [   40  2250  9600 22050 62000]]


[[ 60 250 400 450 500]
 [ 60 250 400 450 500]
 [ 60 250 400 450 500]
 [ 60 250 400 450 500]
 [ 60 250 400 450 500]]
```

divide()

```
print(np.divide(a, b))
print("\n")

print(np.divide(d, c))
print("\n")
```

```
[  1.66666667  10.          25.          50.         125.        ]


[[0.2        0.044      0.         0.02       0.014     ]
 [0.16666667 0.016      0.0275     0.01333333 0.018     ]
 [0.15       0.008      0.025      0.02       0.022     ]
 [0.08333333 0.056      0.         0.02444444 0.016     ]
 [0.08333333 0.048      0.0125     0.01111111 0.022     ]]
```

mod()

```
print(np.mod(a, 3))
print("\n")

print(np.mod(a, 6))
print("\n")

print(np.mod(c, 4))
print("\n")

print(np.mod(d, 2))
print("\n")


print(np.mod(a, b))
print("\n")

print(np.mod(a, e))
print("\n")
```

```
[1 2 1 0 1]


[4 2 4 0 4]


[[0 2 0 2 0]
 [0 2 0 2 0]
 [0 2 0 2 0]
 [0 2 0 2 0]
 [0 2 0 2 0]]


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


[4 0 0 0 0]
```

```
[10  6  1 18 30]
```

remainder()

```
print(np.remainder(a, 4))
print("\n")

print(np.remainder(a, 7))
print("\n")

print(np.remainder(c, 5))
print("\n")

print(np.remainder(d, 9))
print("\n")
```

```
[2 2 0 2 2]


[3 1 2 3 5]


[[0 0 0 0 0]
 [0 0 0 0 0]
 [0 0 0 0 0]
 [0 0 0 0 0]
 [0 0 0 0 0]]


[[3 2 0 0 7]
 [1 4 2 6 0]
 [0 2 1 0 2]
 [5 5 0 2 8]
 [5 3 5 5 2]]
```

```
print(np.remainder(a, b))
print("\n")

print(np.remainder(a, e))
print("\n")

print(np.remainder(c, d))
print("\n")

print(np.remainder(d, b))
print("\n")
```

```
[4 0 0 0 0]


[10  6  1 18 30]
```

```
[[ 0  8  0  0  3]
 [ 0  2  4  0  5]
 [ 6  0  0  0  5]
 [ 0 12  0 10  4]
 [ 0 10  0  0  5]]
```

```
[[0 1 0 0 1]
 [4 4 3 0 1]
 [3 2 2 0 1]
 [5 4 0 2 0]
 [5 2 1 2 1]]
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: RuntimeWarning: divide by zero
  import sys
```