MatriX/ Arthimatic Operations on NDARRAY of Numpy

On ndarray object, we can perform mentioned ndarray Operation:

1) add()---------->numpy.add(ndarrayobj1, ndarrayobj2)----------(+)

2)subtract()---------->numpy.subtract(ndarrayobj1, ndarrayobj2)----------(-)

3)multiply()---------->numpy.multiply(ndarrayobj1, ndarrayobj2)----------(*)

4)matmul

5) dot()

6) divide() ---------->numpy.divide(ndarrayobj1, ndarrayobj2)----------(/)

7) floor_divide() --------> numpy.floor_divide(ndarrayobj1, ndarrayobj2)----------(//)

8) mod() ----------> numpy.mod(ndarrayobj1, ndarrayobj2)---------- (%)

9)power()---------->numpy.power(ndarrayobj1, ndarrayobj2)----------(**)

Note:In Option(3), we do the element wise matrix multiplication. Actual Matrix Multiplication can be done by using (4) and (5)

In [1]:
```
# We can perform above arthimatic operation either by suing functions or by using operators
```

In [7]:
```
import numpy as np
```

In [9]:
```
a=np.array([[10,20],[30,40]])
print(a)
```

```
[[10 20]
 [30 40]]
```

In [5]:
```
b=np.array([[1,2],[3,4]])
print(b)
```

```
[[1 2]
 [3 4]]
```

In [10]:
```
c=np.add(a,b)  #USING add()  function
print(c)
```

```
[[11 22]
 [33 44]]
```

In [13]:
```
c1=a+b   #Using + operator
print(c1)
```

```
[[11 22]
 [33 44]]
```

In [14]:
```
d=np.subtract(a,b) #using Subtract function
print(d)
```

```
[[ 9 18]
 [27 36]]
```

In [15]:
```
d1= a-b
d1
```

Out[15]:
```
array([[ 9, 18],
       [27, 36]])
```

In [16]:
```
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [22]:
```
print(d1)
```

```
[[ 9 18]
 [27 36]]
```

In [23]:
```
a.ndim
```

Out[23]:
```
2
```

In [24]:
```
d1.dtype
```

Out[24]:
```
dtype('int32')
```

In [29]:
```
print(d1,object)
```

```
[[ 9 18]
 [27 36]] <class 'object'>
```

In [30]:
```
d1.dtype
```

Out[30]:
```
dtype('int32')
```

In [34]:
```
# Element wise Matrux Multiplication
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [36]:
```
c= np.array(a*b)
print(c)
```

```
[[ 10  40]
 [ 90 160]]
```

In [38]:
```
c=a*b
print(c)
```

```
[[ 10  40]
 [ 90 160]]
```

In [39]:
```
c=np.multiply(a,b)
c
```

Out[39]:
```
array([[ 10,  40],
       [ 90, 160]])
```

In [40]:
```
#Actual Matrix Multiplication
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [42]:
```
c=np.matmul(a,b)
print(c,type(c))
```

```
[[ 70 100]
 [150 220]] <class 'numpy.ndarray'>
```

In [45]:
```
c=np.dot(a,b)
print(c,type(c))
```

```
[[ 70 100]
 [150 220]] <class 'numpy.ndarray'>
```

In [47]:
```
a.dtype
```

Out[47]:
```
dtype('int32')
```

In [48]:
```
c=np.dot(a,b)
print(c,object)
```

```
[[ 70 100]
 [150 220]] <class 'object'>
```

In [56]:
```
"""Note: We can use matmul() or Dot() for same multiplication purpose. Anything we can use.
dot()/ matmul()-----for actual matrix multiplication
multiple()-----For elememy wise matrix Multiplication."""
```

Out[56]:
```
'Note: We can use matmul() or Dot() for same multiplication purpose. Anything we can use.\ndot()/ matmul()-----for actual matrix multiplication\nmultiple()-----For elememy wise matrix Multiplication.'
```

In [57]:
```
# Element wise division
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [60]:
```
c=np.divide(a,b)
print(c,type(c))
```

```
[[10. 10.]
 [10. 10.]] <class 'numpy.ndarray'>
```

In [61]:
```
d=a/b
print(d)
```

```
[[10. 10.]
 [10. 10.]]
```

In [65]:
```
# Element wise floor_ Division by using function and also operator separately
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [63]:
```
c=np.floor_divide(a,b)
print(c,type(c))
```

```
[[10 10]
 [10 10]] <class 'numpy.ndarray'>
```

In [64]:
```
c= a//b
print(c,type(c))
```

```
[[10 10]
 [10 10]] <class 'numpy.ndarray'>
```

In [66]:
```
# Element wise modulo Division by using function and also operator separately
```

In [67]:
```
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [68]:
```
c=np.mod(a,b)
print(c,type(c))
```

```
[[0 0]
 [0 0]] <class 'numpy.ndarray'>
```

In [69]:
```
c=a%b
print(c,type(c))
```

```
[[0 0]
 [0 0]] <class 'numpy.ndarray'>
```

In [ ]:
```
# Element wise power (exponentiation) operation by using function and also operator separately
```

In [70]:
```
print(a)
print(b)
```

```
[[10 20]
 [30 40]]
[[1 2]
 [3 4]]
```

In [71]:
```
c=np.power(a,b)
print(c,type(c))
```

```
[[     10      400]
 [  27000 2560000]] <class 'numpy.ndarray'>
```

In [72]:
```
c=(a**b)
print(c,type(c))
```

```
[[     10      400]
 [  27000 2560000]] <class 'numpy.ndarray'>
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: