In [1]:
```
import numpy as np
import timeit
```

In [8]:
```
#generating input and printing it
np.random.seed(24787)
X = np.random.randint(-1000, 1000, size=3000)
Y = np.random.randint(-1000, 1000, size=3000)
X,Y
```

Out[8]:
```
(array([-646, -482,  700, ..., -404,  149, -456]),
 array([-446, -671, -499, ..., -363, -711, -501]))
```

In [9]:
```
def NUMPY_outer(X,Y):
    C = np.full((X.size,Y.size),0)
    for i in range(X.size):
        for j in range(Y.size):
            C[i][j] = X[i] * Y[j]
    return C
```

In [10]:
```
start = timeit.default_timer()
Z = NUMPY_outer(X,Y)
stop = timeit.default_timer()
#pritning time and the outer product we obtained
print('Time: ', stop - start)
print(Z)
```

```
Time:  3.2401320529997975
[[ 288116  433466  322354 ...  234498  459306  323646]
 [ 214972  323422  240518 ...  174966  342702  241482]
 [-312200 -469700 -349300 ... -254100 -497700 -350700]
 ...
 [ 180184  271084  201596 ...  146652  287244  202404]
 [ -66454  -99979  -74351 ...  -54087 -105939  -74649]
 [ 203376  305976  227544 ...  165528  324216  228456]]
```

In [11]:
```
start = timeit.default_timer()
C = np.outer(X,Y)
stop = timeit.default_timer()
#printing time and outer product from numpy
print('Time: ', stop - start)
print(C)
```

```
Time:  0.038213711000025796
[[ 288116  433466  322354 ...  234498  459306  323646]
 [ 214972  323422  240518 ...  174966  342702  241482]
 [-312200 -469700 -349300 ... -254100 -497700 -350700]
 ...
 [ 180184  271084  201596 ...  146652  287244  202404]
```

```
 [ -66454  -99979  -74351 ...  -54087 -105939  -74649]
 [ 203376  305976  227544 ...  165528  324216  228456]]
```

In [7]:

```
#Testing Accuracy
print(Z-C)
```

```
[[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 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

Explanation for the time difference

The numpy library vectorizes the array and can perform operations on multiple array elements at once. But our for loop does the m*n operations separately. So numpy is way faster as shown in the cells above

In [ ]: