Matrix-Vector/Matrix Multiplication

When multiplying a Matrix and a Vector together, you should be thinking about this as simply Matrix-Matrix multiplication. After all, vectors and matrices are one in the same: you may have noticed by now that n-vectors are merely $n \times 1$ matrices! This should simplify much of how we think about resultant matrix outputs.

We've read that matrix multiplication *is not commutative*. This is because the dimensions of both matrices determine the dimensions of the output matrix. As you've read, to multiply two matrices together, the inner dimensions must match. In other words:

A matrix, A, with dimensions $m \times p$ can only be multiplied by another matrix, B, with dimensions $p \times n$. Again, in other words: *The first matrix must have the same number of columns as rows of the second matrix.*

Take the expression $y = Ax$.

- A is an $(m \times n)$ matrix.
- x is an n-vector, or an $n \times 1$ matrix.

We can now see this as a legal operation: We are multiplying an $m \times n$ matrix with an $n \times 1$ matrix:

$m \times (n * n) \times 1$ --> inner dimensions are equal, and we will output a matrix with the outer dimensions.

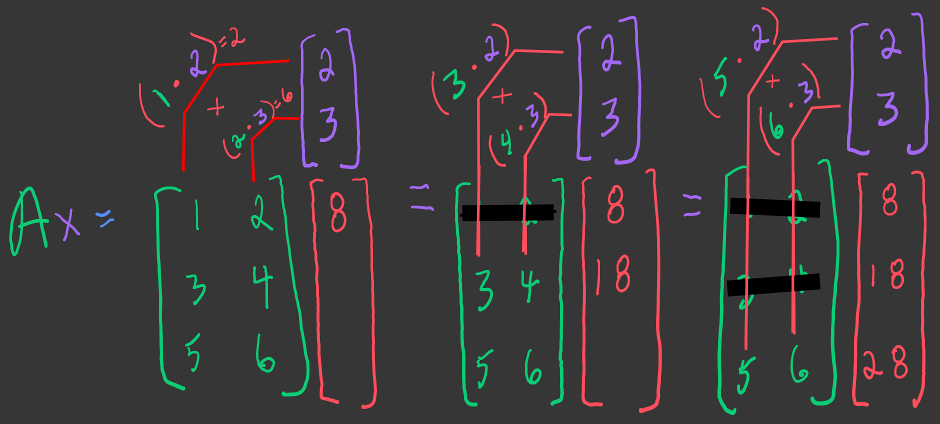
The output, y, is an $m \times 1$ matrix, or an m-vector.

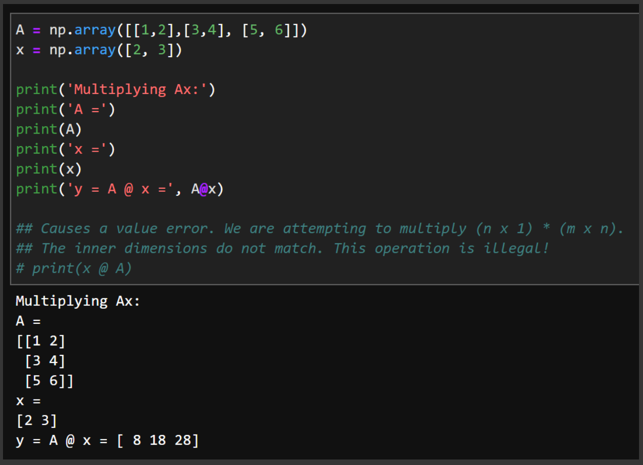
$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \quad x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

A is a $\overset{(m}{3} \times \overset{n)}{2}$ matrix

x is a $\overset{(n)}{2}$-vector.

From 6.4, y will be a $\overset{(m)}{3}$-vector;



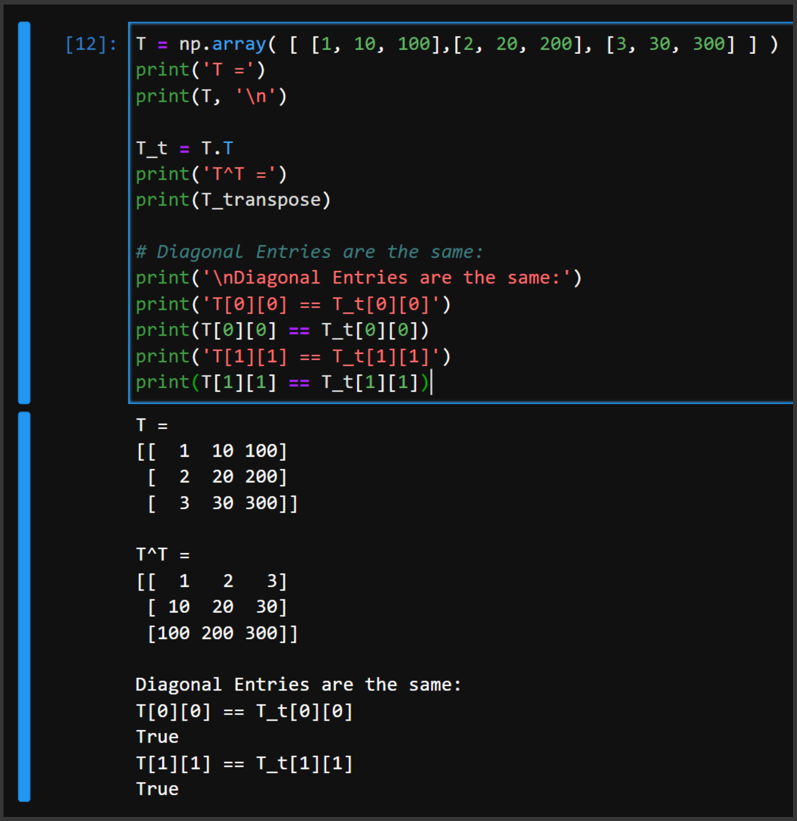

```
A = np.array([[1,2],[3,4], [5, 6]])
x = np.array([2, 3])

print('Multiplying Ax:')
print('A =')
print(A)
print('x =')
print(x)
print('y = A @ x =', A@x)

## Causes a value error. We are attempting to multiply (n x 1) * (m x n).
## The inner dimensions do not match. This operation is illegal!
# print(x @ A)
```

```
Multiplying Ax:
A =
[[1 2]
 [3 4]
 [5 6]]
x =
[2 3]
y = A @ x = [ 8 18 28]
```

Visualizing the Transpose

$$T = \begin{bmatrix} 1 & 10 & 100 \\ 2 & 20 & 200 \\ 3 & 30 & 300 \end{bmatrix}$$

$$T^T = \begin{bmatrix} 1 & 2 & 3 \\ 10 & 20 & 30 \\ 100 & 200 & 300 \end{bmatrix}$$



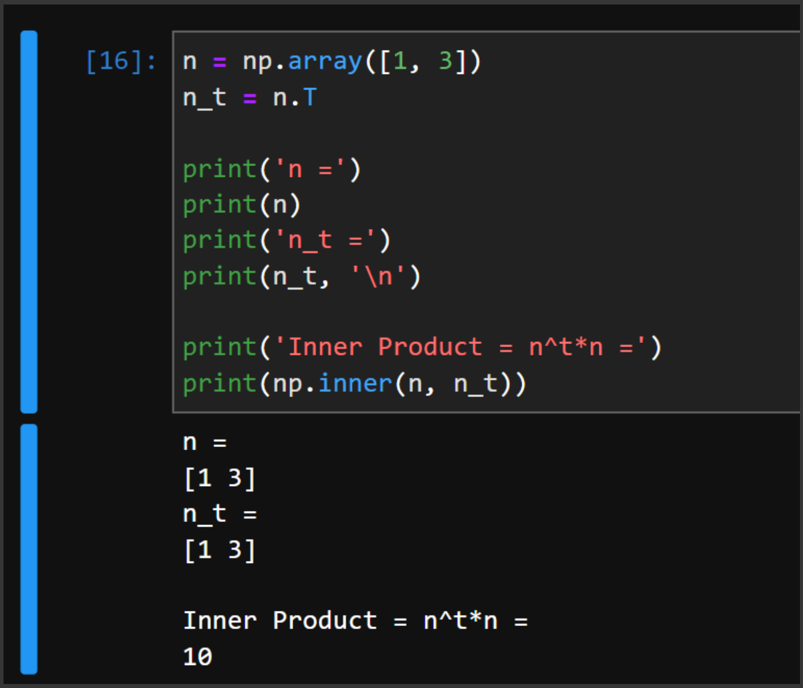
```
[12]: T = np.array( [ [1, 10, 100],[2, 20, 200], [3, 30, 300] ] )
      print('T =')
      print(T, '\n')

      T_t = T.T
      print('T^T =')
      print(T_transpose)

      # Diagonal Entries are the same:
      print('\nDiagonal Entries are the same:')
      print('T[0][0] == T_t[0][0]')
      print(T[0][0] == T_t[0][0])
      print('T[1][1] == T_t[1][1]')
      print(T[1][1] == T_t[1][1])
```

```
T =
[[  1  10 100]
 [  2  20 200]
 [  3  30 300]]

T^T =
[[  1   2   3]
 [ 10  20  30]
 [100 200 300]]

Diagonal Entries are the same:
T[0][0] == T_t[0][0]
True
T[1][1] == T_t[1][1]
True
```

$$X = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

$$X^T = \begin{bmatrix} 1 & 3 \end{bmatrix}$$

$$X^T X = \overset{1\cdot 1 \quad 3\cdot 3}{\begin{bmatrix} 1 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix}} = \begin{bmatrix} 10 \end{bmatrix} = 10$$



```
[16]: n = np.array([1, 3])
      n_t = n.T

      print('n =')
      print(n)
      print('n_t =')
      print(n_t, '\n')

      print('Inner Product = n^t*n =')
      print(np.inner(n, n_t))
```

```
n =
[1 3]
n_t =
[1 3]

Inner Product = n^t*n =
10
```