Numpy - 3

Prereqs:

① You have attended and fully understood the ideas and concepts discussed in BOTH prev sessions

② You have:

(a) Pen + Paper (or equivalent)

(b) Colab ready to run (once link shared)

Agenda:

→ Aggregate functions ✓✓

→ Sorting ✓✓

→ Multiplication scalar, vectors, matrix

→ Vectorization

→ Broadcasting

Indexes: 0 1 2 3 4 5

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

o/p → 1, 3, 5

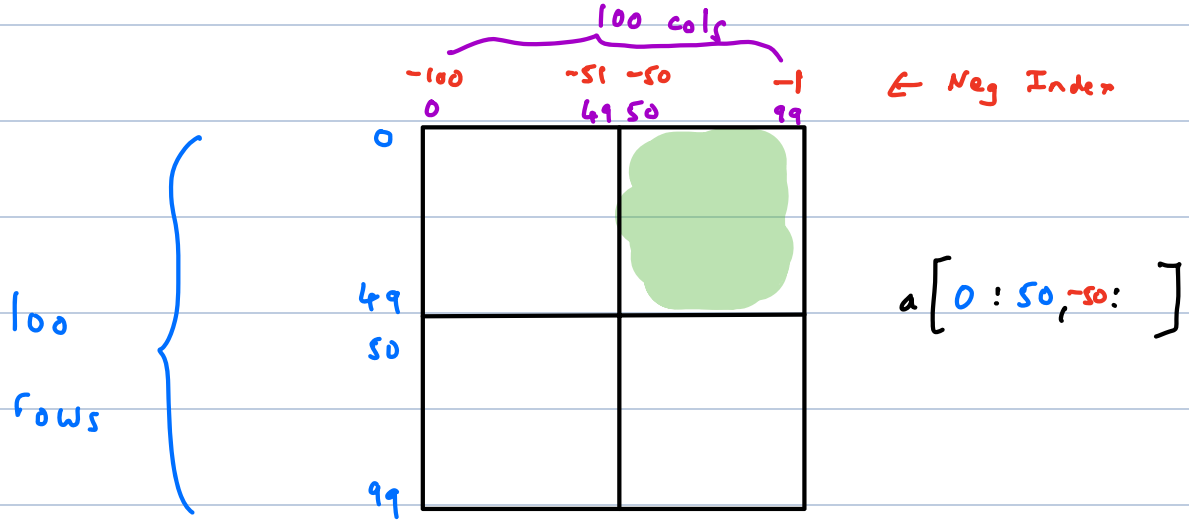
```
a[start : stop : step]
```

start: 1 — included in result

stop: 5+1 — not included

step: +2

If array "a" has an "100 x 100" pixel image, then, which of these will plot only the "top right" quarter of that image?



10 15 25

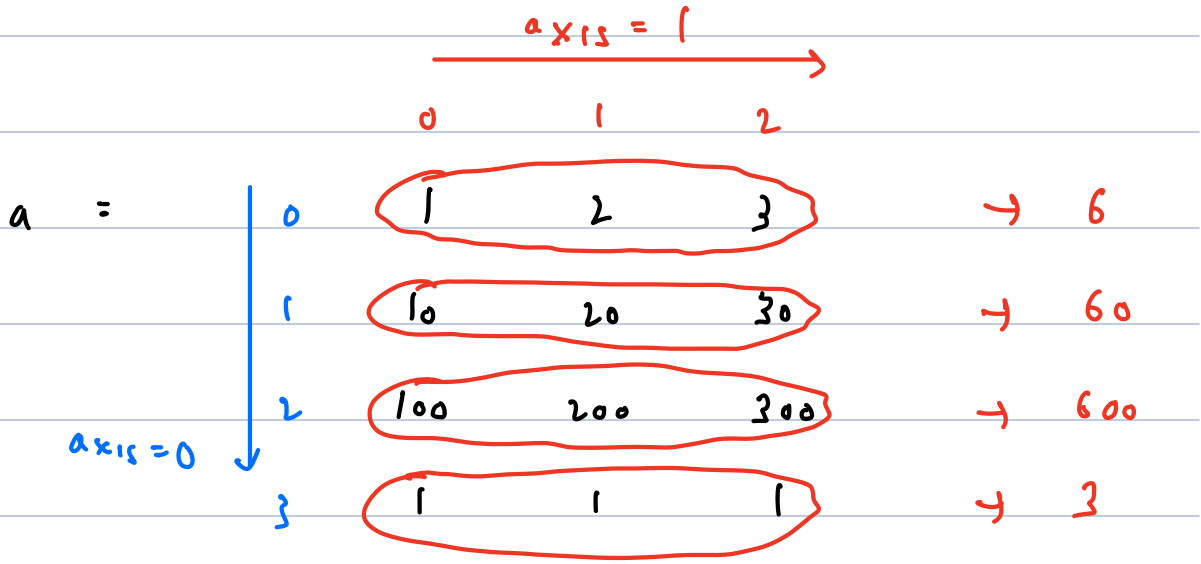
sum = 50

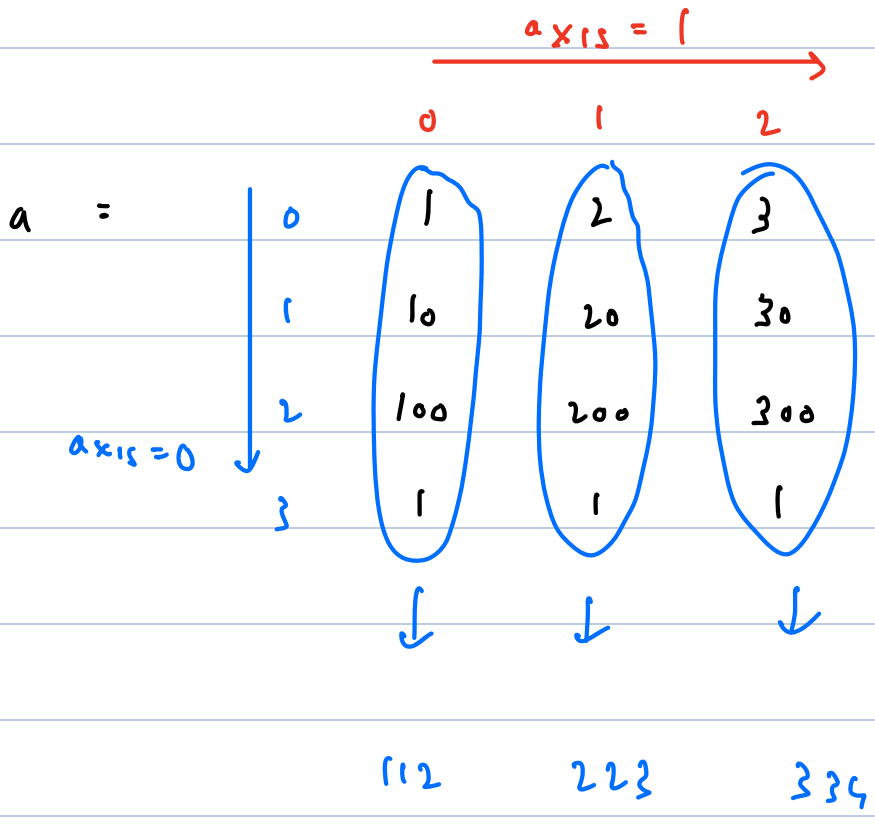
mean = 16.67

max = 25

min = 10

Sum of 2D array





Break! Resuming at 10:22 PM IST

In Linear Algebra

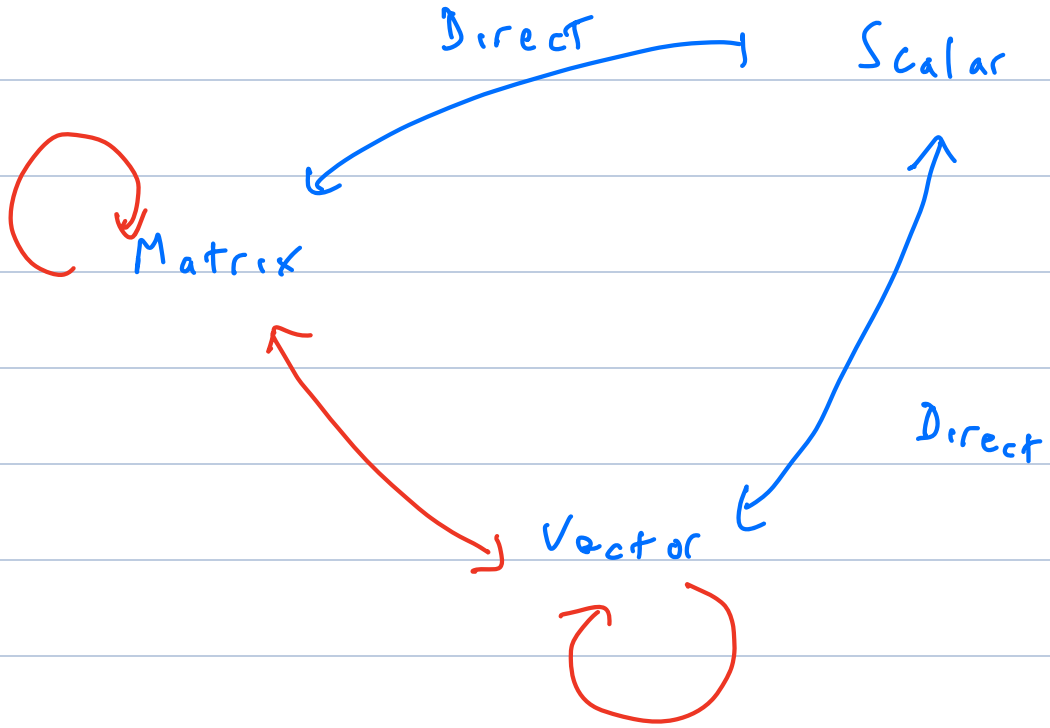
Scalar value : 6 / -5 / 27.48

Vector → 1D array of values

$$\begin{bmatrix} 1 \\ 2 \\ 10 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 10 \end{bmatrix}$$

Matrix → 2d array

$$\begin{bmatrix} 5 & 10 \\ -1 & 16 \\ 0 & 8 \\ 0 & 0 \end{bmatrix}$$



shape_1 = shape_2

np.dot (Vector, Vector)

np.dot (Matrix, Vector)

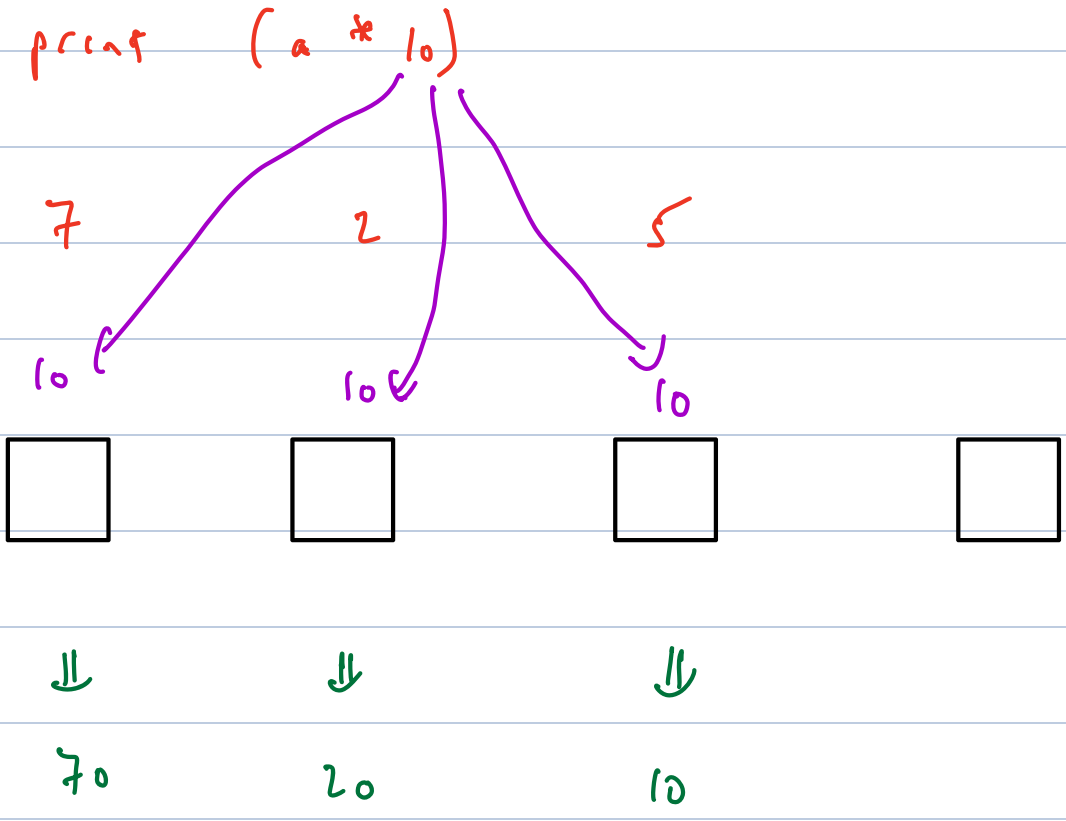
np.dot (Matrix, Matrix)

Result

(M_1, n_1) (m_2, n_2) → (M_1, n_2)

Equal

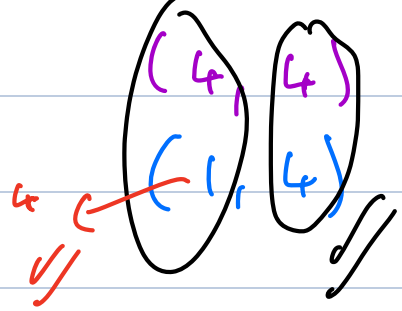
Vectorization




```
a = np.array([7, 2, 5])

b = np.array([10, 10, 15])

print (a+b)
```



(4, 4)

arr-2

arr-1

$$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 1 & 7 & 3 & 5 \\ 2 & 8 & 6 & 9 \\ 11 & 23 & 12 & 19 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$

arr-1

$$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 1 & 7 & 3 & 5 \\ 2 & 8 & 6 & 9 \\ 11 & 23 & 12 & 19 \end{bmatrix}$$

arr-3

$$\begin{bmatrix} 5 & 6 & 8 \end{bmatrix}$$

(4, 4)

(1, 3)

(4, 4)

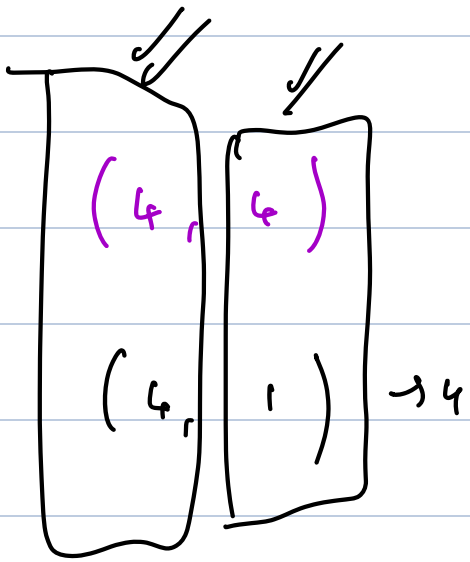
arr-1

(4, 1)

arr-4

2	3	4	5
1	7	3	5
2	8	6	9
11	23	12	19

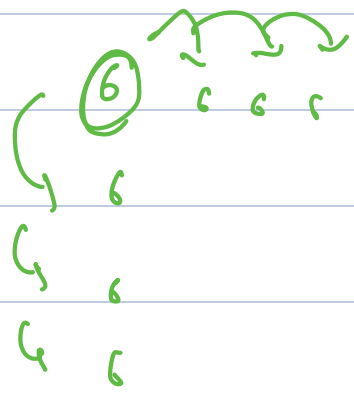
[2 2 2 2]



arr-1

2	3	4	5
1	7	3	5
2	8	6	9
11	23	12	19

n



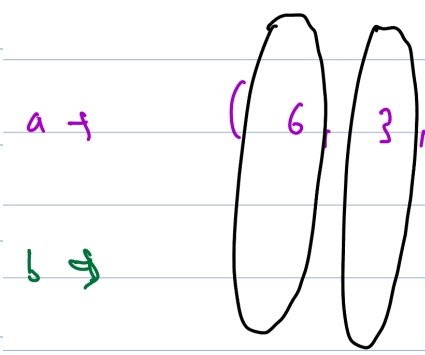
(4, 4)

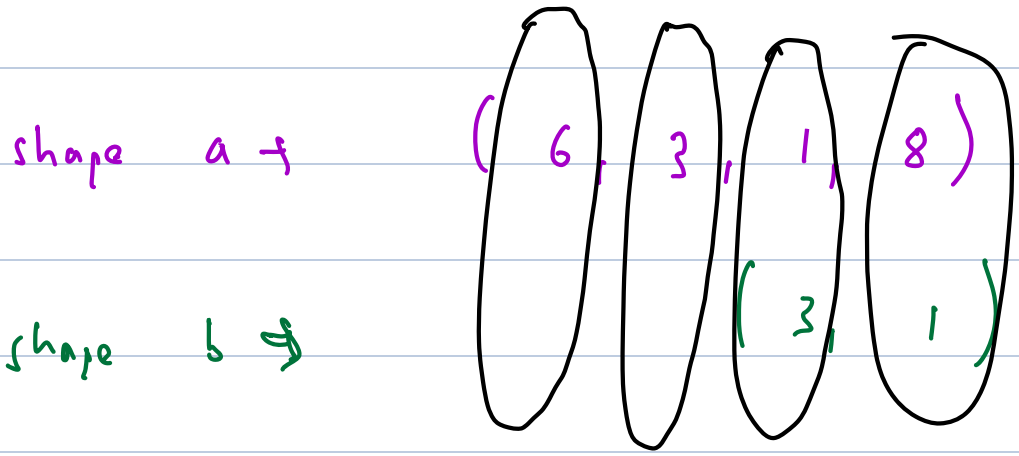
4, 4

(,)

3, 2

Num, 2,





① Write the two shapes one below the other & RIGHT-ALIGNED

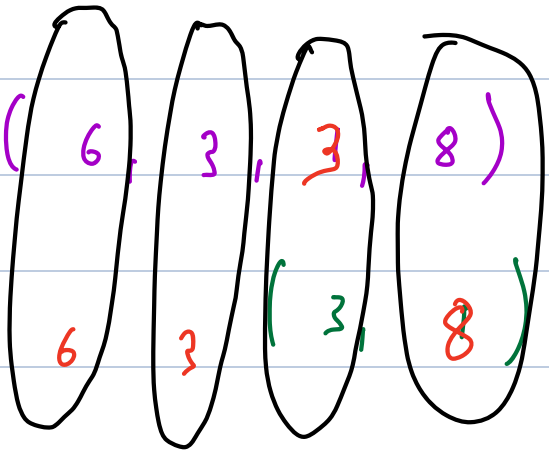
② For each pair, put a ✓ mark if one of these conditions is satisfied:

(a) Both values are equal

(b) If one of the values is BLANK / 1 then BROADCAST it to match the other number

shape a →

shape b →



Result → (6, 3, 3, 8)