NP - 4

Prereqs:

① You have attended and fully understood the ideas and concepts discussed in ALL 3 prev sessions

② You have:

(a) Colab open & ready

(b) Pen + Paper (or equivalent)

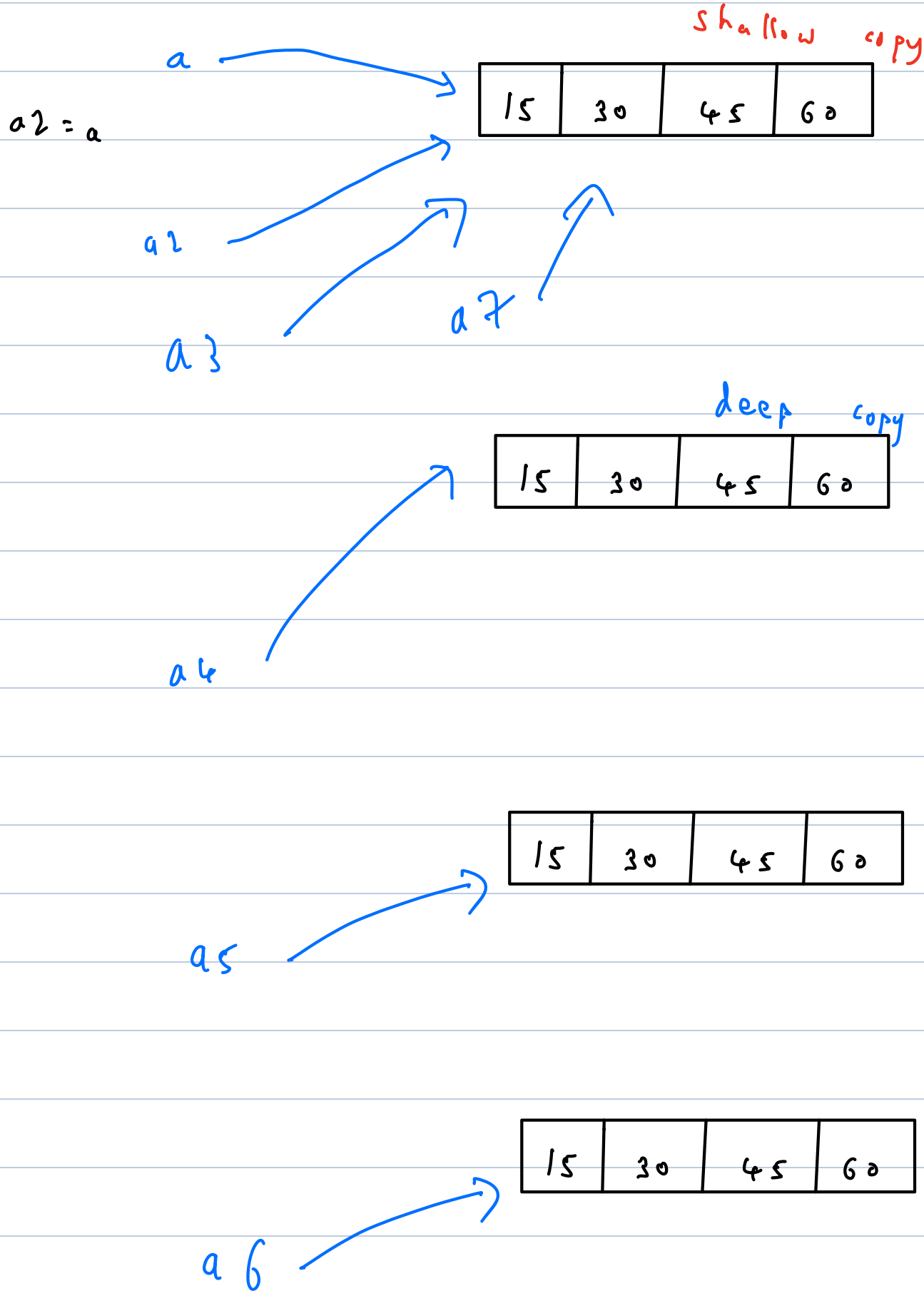
Agenda:

① Copying Arrays: shallow / deep

① Splitting and Stacking Arrays

Bonus Content:

How 3D + arrays are displayed in Python output


shallow copy
a
a2 = a
15
30
45
60
a2
a3
a7
deep copy
15
30
45
60
a4
15
30
45
60
a5
15
30
45
60
a6

HSBC

transaction

Amit	→	78
Amit	→	350
Ashok	→	700
Daryll	→	400
Daryll	→	300
Amit	→	129

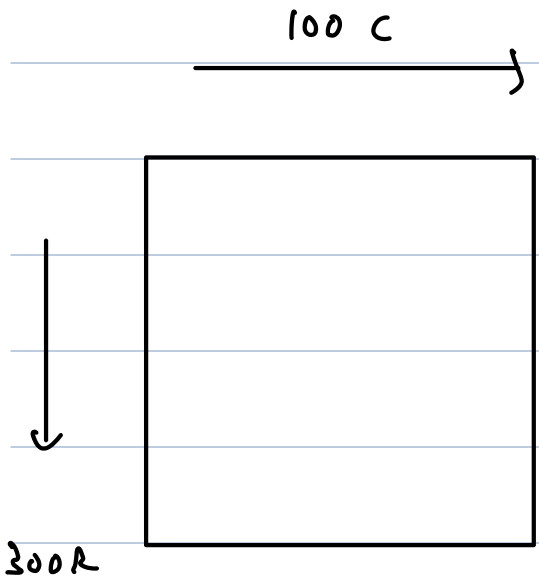
In Python List

Slicing creates a deep copy ✓

In NP array,

Slicing creates a deep copy ✗

Vertical Split



n-pieces: 2, 3, 4 . . . .

I can split into EQUAL pieces

or UNEQUAL pieces

pieces = np.vsplit(arr, n-pieces) → return a

LIST of n "pieces"

Vertical stacking

Reversing

List_of_pieces = [v_pieces[1], v_pieces[0]]

#order matters

np.vstack(List_of_pieces)

Summary of functions

"n" Equal pieces

V split → np.vsplit(arr, n)

V stack → np.vstack(list-of-n-pieces)

"n" equal pieces

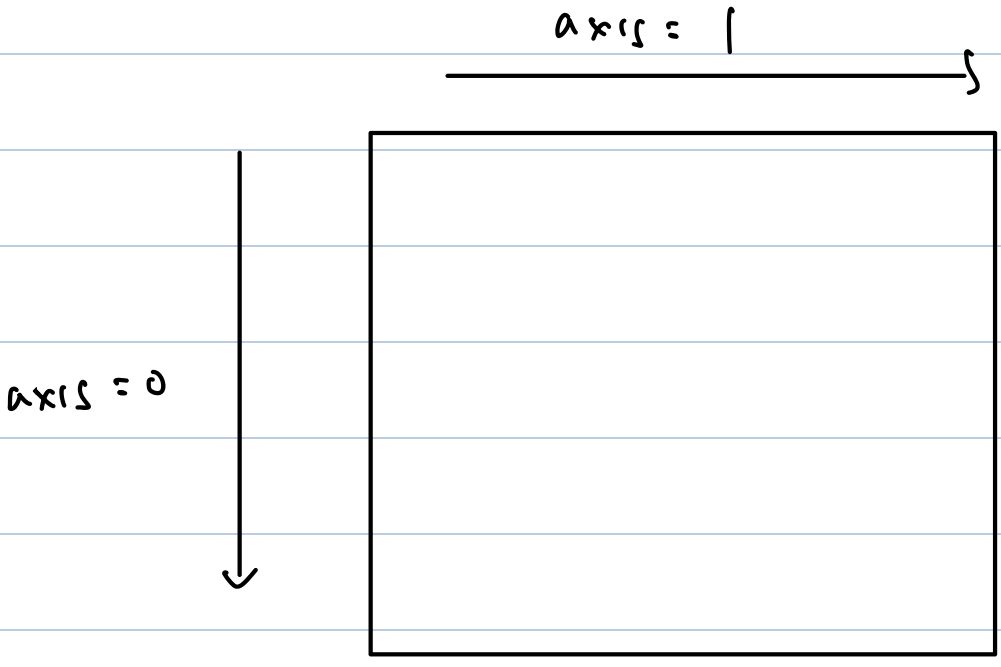
H split → np.hsplit(arr, n)

H stack → np.hstack(list-of-n-pieces)

Generic functions for splitting & stacking

np.split (arr, n, axis: 0/1)

np.stack ([list_of_pieces], axis = 0/1)
np.concatenate([list_of_pieces], axis = 0/1)



batter : ["Virat", "Rohit", "Jaiswal", "Rahul"]
0 1 2 3

runs : [3, 0, 4, 33]
0 1 2 3

np.argsort(runs) → [1, 0, 2, 3]

batter [[1, 0, 2, 3]]

→ ["Rohit", "V", "J", "Rahul"]

np.argmax(runs) → 3

np.argmin(runs) → 1

Q & A segment

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

np.ALL ([T, F, T, T, F])

↳ "T" if every val is True

↳ "F" if ANY one element is False

marks : [58 , 98 , 72 , .. 65]

"33"

np.all (marks >= 33)

np.any (marks >= 33)
np.any (marks < 33)

np.ANY → "T" if ANY ONE is True

"F" if ANY ONE is False