PRIME INTUIT
Finishing School

Indexing array

 # Indexing of Array's

I = 1
J = 2
K = 0

Arr3d[I, j, k]

Arr3d[0, :, :]

Arr3d[1, :, :]

Arr3d[:, 1, :]

Arr3d[;, ;, 0:2]

PRIME INTUIT
Finishing School

Indexing array

```
# Fancy indexing

Arr3d % 2 == 0

Arr3d[Arr3d % 2 == 0 ]

Arr3d[Arr3d % 2 == 1 ]

Arr3d[(Arr3d % 2 == 1) & (Arr3d > 3)]

Arr_Slice = Arr3d[:, :, 0:2]

Print(type(Arr_Slice))
```

PRIME INTUIT
Finishing School

Indexing array

Arr_Slice.ndim

Arr_Slice.shape

Arr_Slice[0, 0, 1]

Arr_Slice[0, 0, 1] = 1999
Arr_Slice

Arr3d # original array is also updated as its not a deep copy but only ref to the number is changed.

How to tackle this

Arr_Slice = np.copy(Arr3d[:, :, 0:2])

PRIME INTUIT
Finishing School

Indexing array

How to tackle this

```
Arr_Slice = np.copy(Arr3d[:, :, 0:2])

Arr_Slice[0, 0, 1] = 1

Arr_Slice
Arr3d # Remains the same as we did a deep copy

Arr = np.random.randint(0, 10, (5))
Arr
My_List[1, 3, 4]
Arr[My_List]
```

PRIME INTUIT
Finishing School

Numpy Operations

```
Arr1 = np.zeros((3,4))
Arr2 = np.ones((3,4))

Arr1 + Arr2

Arr3 = np.random.rand(3,4)
Arr4 = np.random.rand(3,4)
Arr3
Arr4
Arr3 + Arr4

Arr3 – Arr4
Arr3 * Arr4
Arr3 / Arr4
```

PRIME INTUIT
Finishing School

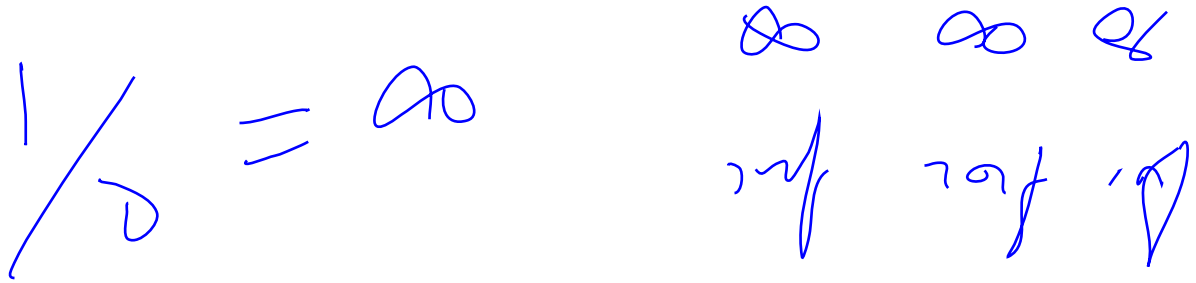
Numpy Operations

```
# Operations on a single Array

Np.exp(Arr3) # exponent the values in array e**x , X is the element in particular location of the array

Np.log(Arr3)  # returns the log of the array element

Np.log(np.exp(Arr3)) # check log and exponent

Np.sin(Arr1)

Np.cos(Arr2)

Np.sqrt(Arr3)
```

PRIME INTUIT
Finishing School

Numpy Operations

```
# Operations on a single Array

Arr_inv = 1 / Arr3

Arr4 = np.zeros((3,4))
Arr_inv1 = 1 / Arr4

Print(Arr_inv1)

# inf referred to infinity

Np.isinf(Arr_inv1[0,0])

Np.isinf(arr_inv)
```



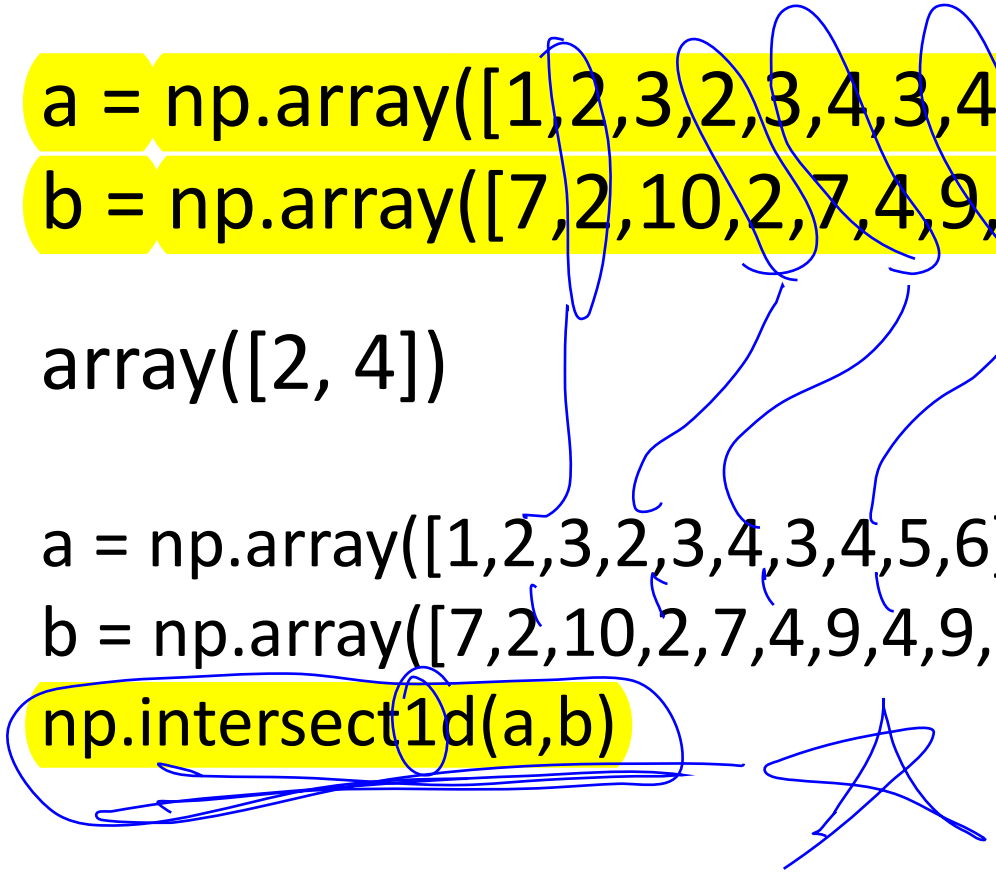
PRIME INTUIT
Finishing School

Numpy Operations

Get the common items between two numpy arrays

```
a = np.array([1,2,3,2,3,4,3,4,5,6])
b = np.array([7,2,10,2,7,4,9,4,9,8])

array([2, 4])
```

```
a = np.array([1,2,3,2,3,4,3,4,5,6])
b = np.array([7,2,10,2,7,4,9,4,9,8])
np.intersect1d(a,b)
```



PRIME INTUIT
Finishing School

Numpy Operations (square and circle exercise)

```
Import numpy as np
Ndim = 2
Npoints = 100000
Points = np.random.rand(npoints,ndim)

dfo = np.zeros((npoints, 1))
Outside_points = 0
For I in range(npoints)
        for j in range(ndim)
                dfo[i] += point[I,j]**2
                dfo[i] =np.sqrt(dfo[i])
        If dfo[i] > 1:
        Outside_points += 1

Print('Fraction of points outside is ', outside_points/npoints)
```

NumPySquare&CIrcleExercise.ipynb - Colaboratory.pdf

PRIME INTUIT
Finishing School

Numpy Operations (square and circle exercise)

```
Import numpy as np
Ndim = 2
Npoints = 100000
Points = np.random.rand(npoints,ndim)

dfo = np.zeros((npoints, 1))
Outside_points = 0
For I in range(npoints)
        for j in range(ndim)
                dfo[i] += point[I,j]**2
                dfo[i] =np.sqrt(dfo[i])
        If dfo[i] > 1:
        Outside_points += 1

Print('Fraction of points outside is ', outside_points/npoints)
```

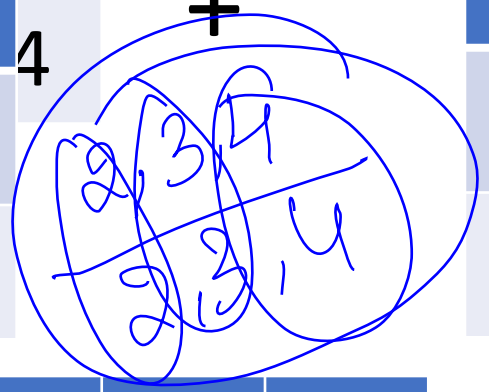
PRIME INTUIT
Finishing School

Numpy Broadcasting

13	14	15	16
17	18	19	20

1	2	3	4
5	6	7	8
9	10	11	12

+



1	2	3	4
5	6	7	8
9	10	11	12

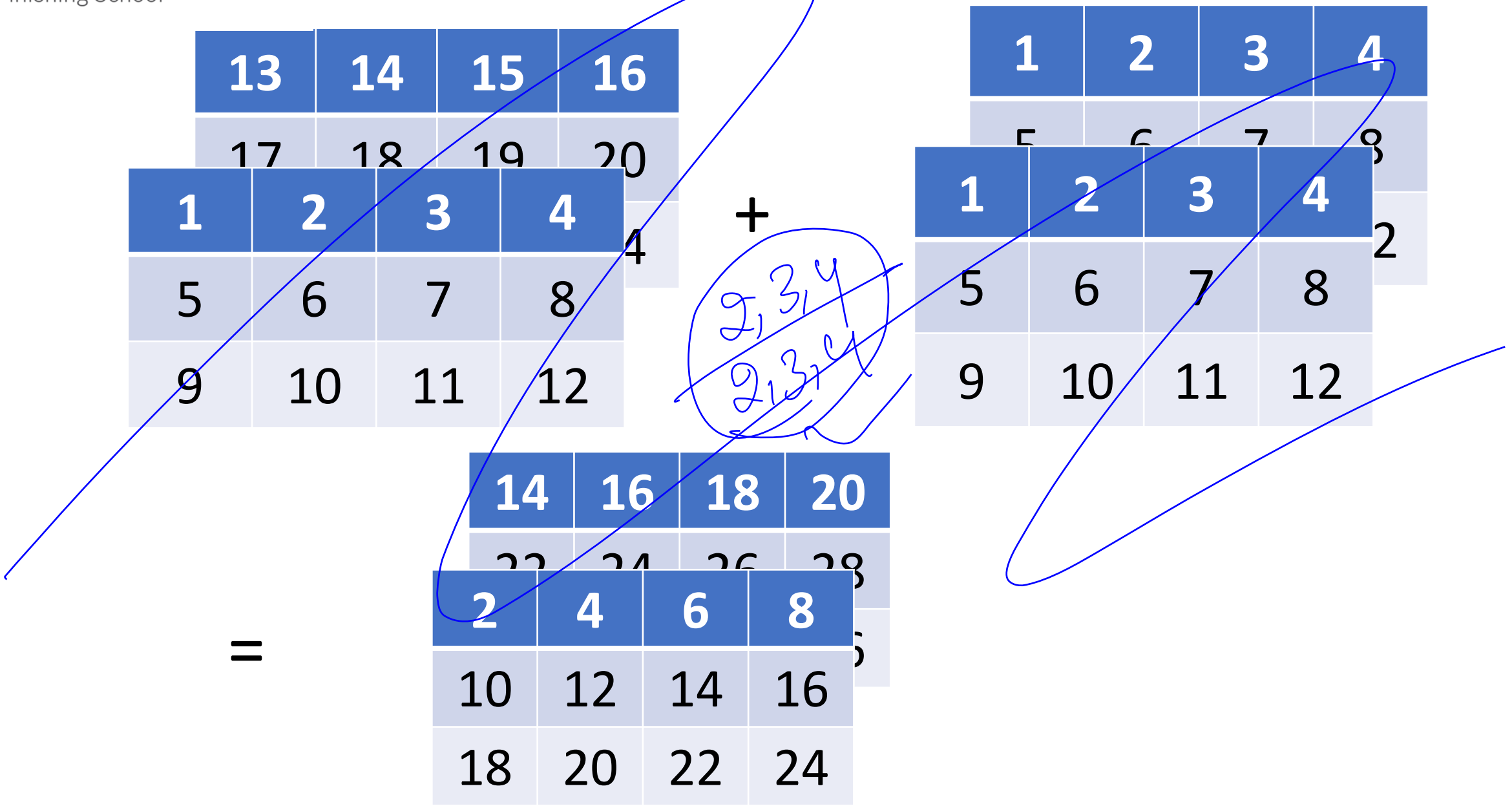
=

14	16	18	20
22	24	26	28

2	4	6	8
10	12	14	16
18	20	22	24

PRIME INTUIT
Finishing School

Numpy Broadcasting



PRIME INTUIT
Finishing School

Numpy Broadcasting

13	14	15	16
17	18	19	20

1	2	3	4
5	6	7	8
9	10	11	12

+

1	1	1	1
5	5	5	5

1	2	3	4
5	6	7	8
9	10	11	12

=

14	15	16	17
22	23	24	25

2	3	4	5
10	11	12	13
18	19	20	21

PRIME INTUIT
Finishing School

Numpy Broadcasting

13	14	15	16
17	18	19	20
5	6	7	8
9	10	11	12

+

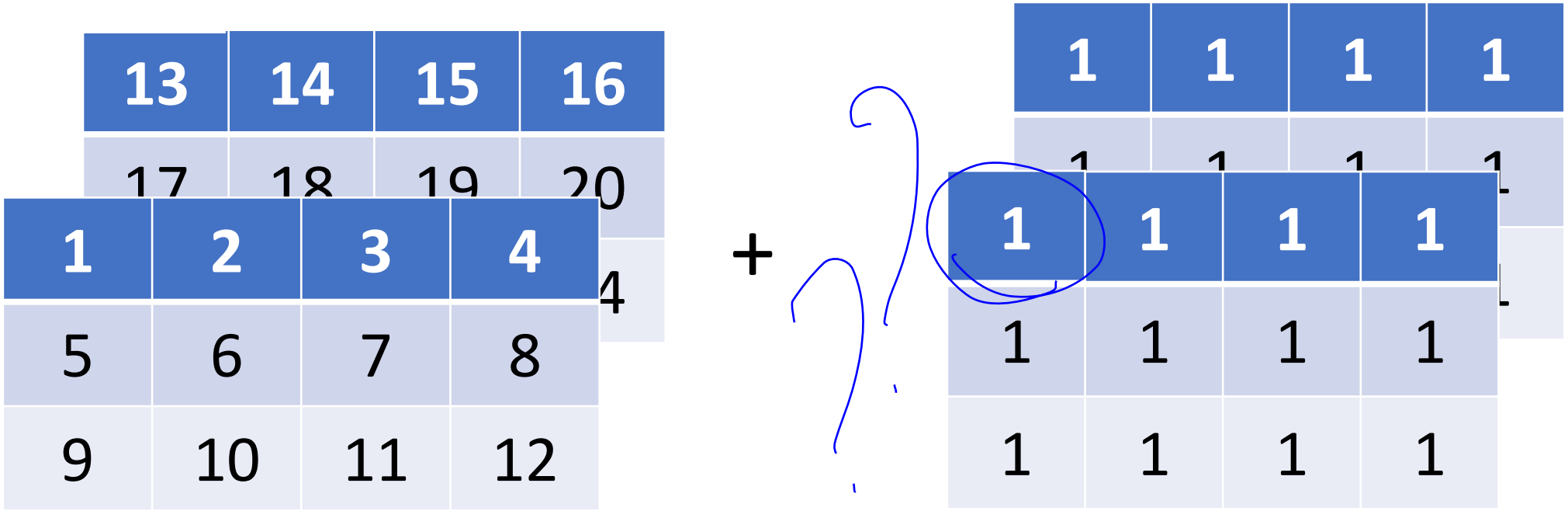
1	2	3	4
1	2	3	4
1	2	3	4

=

14	16	18	20
18	20	22	24
6	8	10	12
10	12	14	16

PRIME INTUIT
Finishing School

Numpy Broadcasting



PRIME INTUIT
Finishing School

Numpy Broadcasting

1	2	3	4

+

1	2	3	4	5

1	1	1	1	1
2	2	2	2	2
3	3	3	3	
4	4	4	4	

+

=

2	3	4	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5