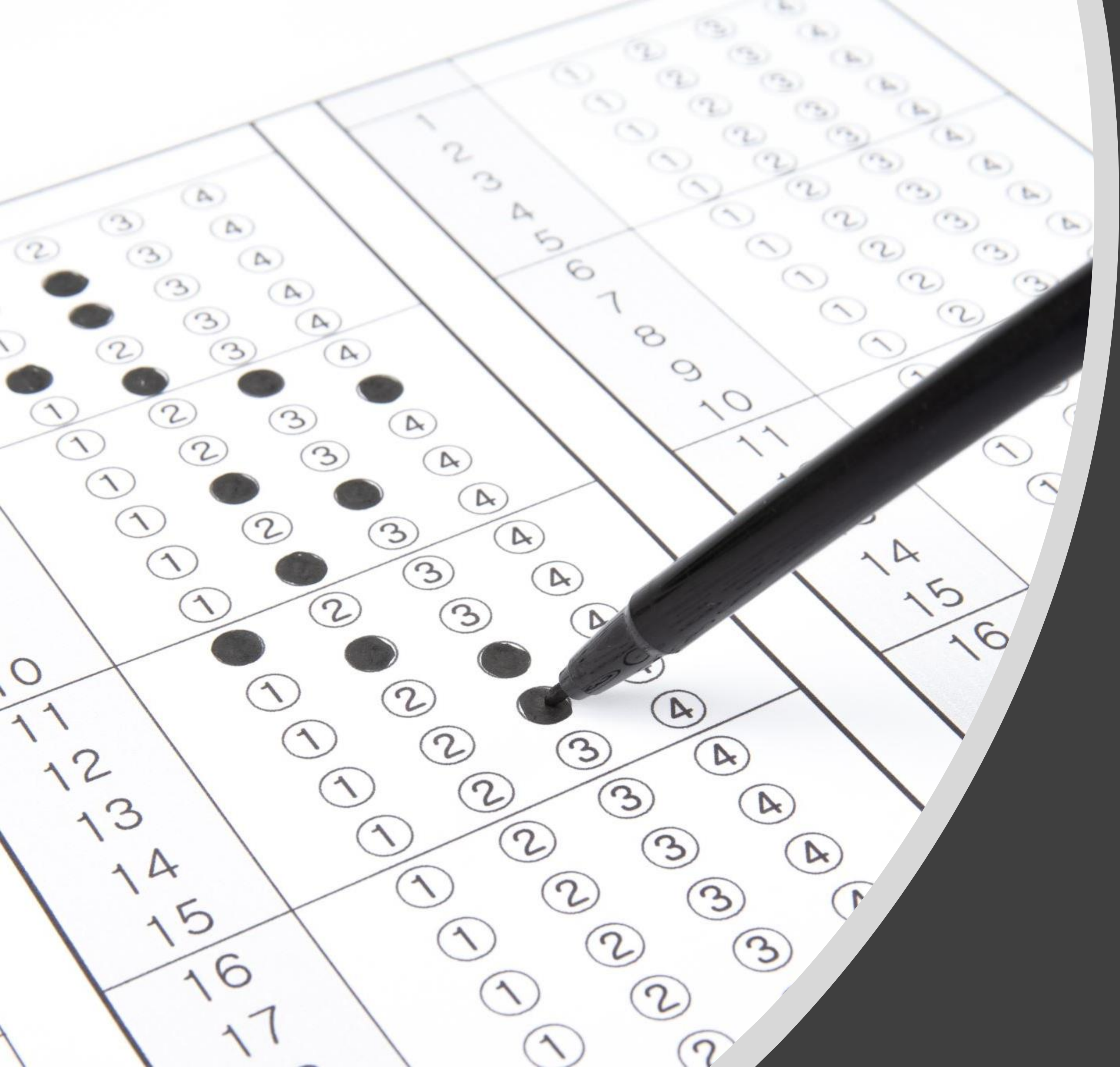


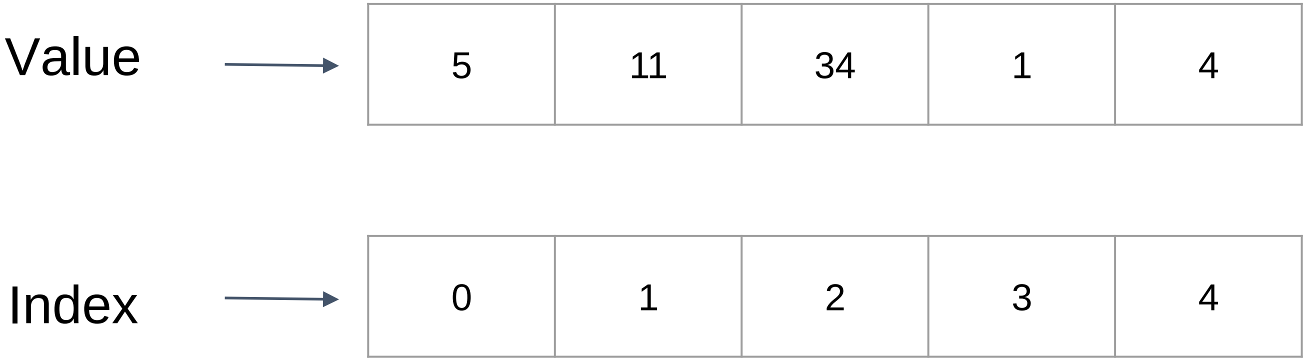
Numpy: Indexing and Selection

Outline

- Indexing in each array type (1D, 2D, 3D)
- Index and mutability in 1D Array
- Index and mutability in 2D Array
- Fancy Indexing
- Conditional Selection

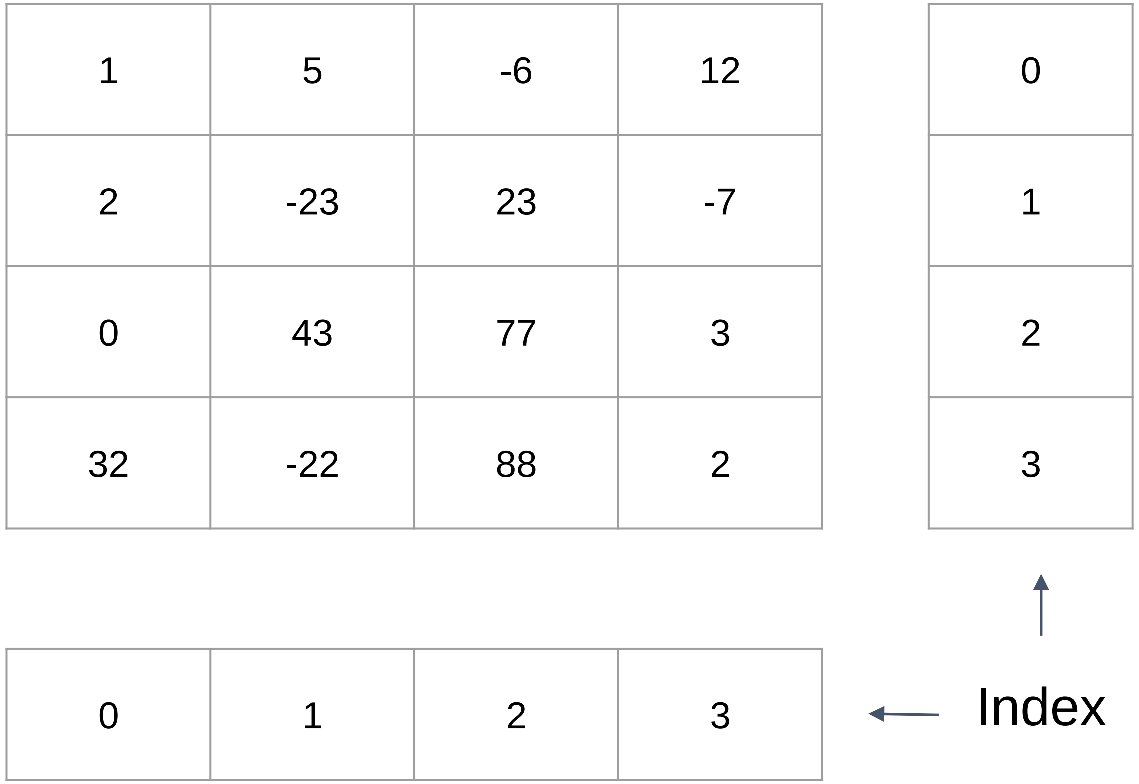
Indexing for each Array Type

1D Array



Value	5	11	34	1	4
Index	0	1	2	3	4

2D Array

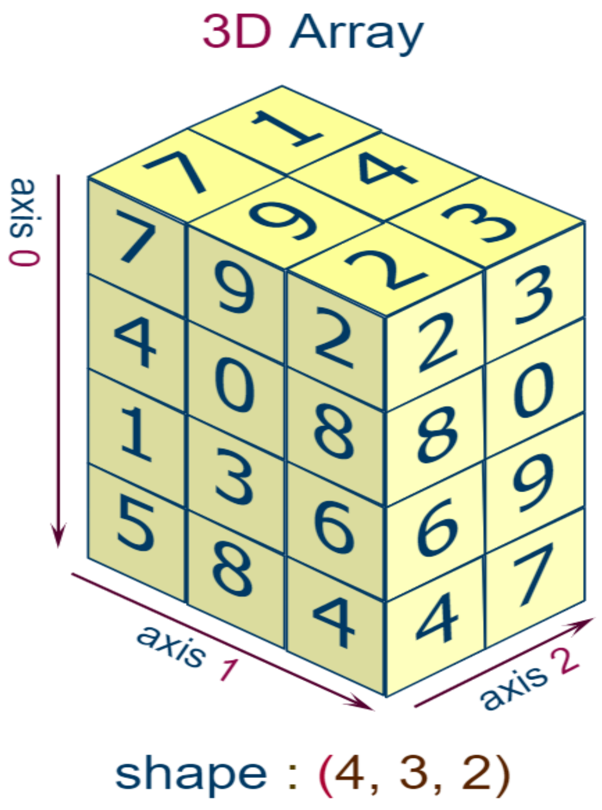


1	5	-6	12
2	-23	23	-7
0	43	77	3
32	-22	88	2

0
1
2
3

0	1	2	3

3D Array



0
1
2
3

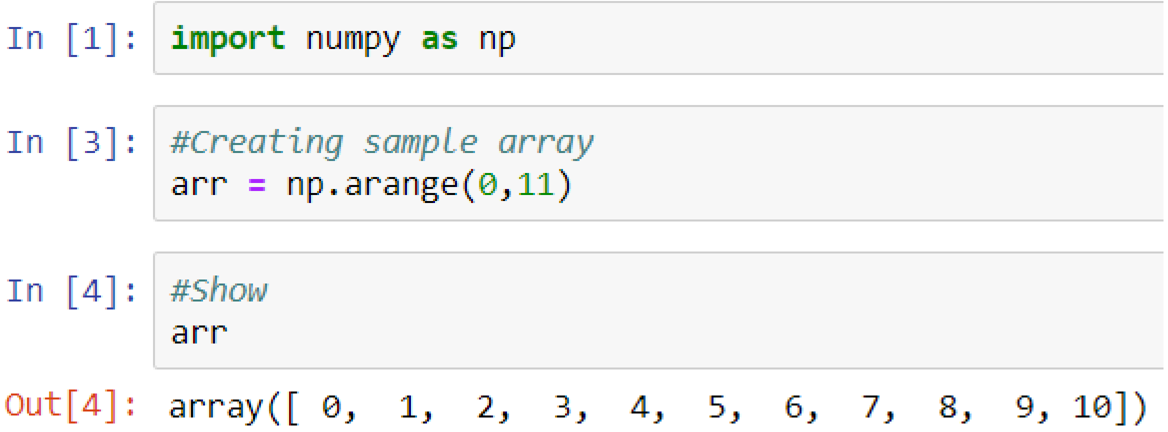
0	1

0	1	2

Index

Index and mutability in 1D Array

New Array Arr



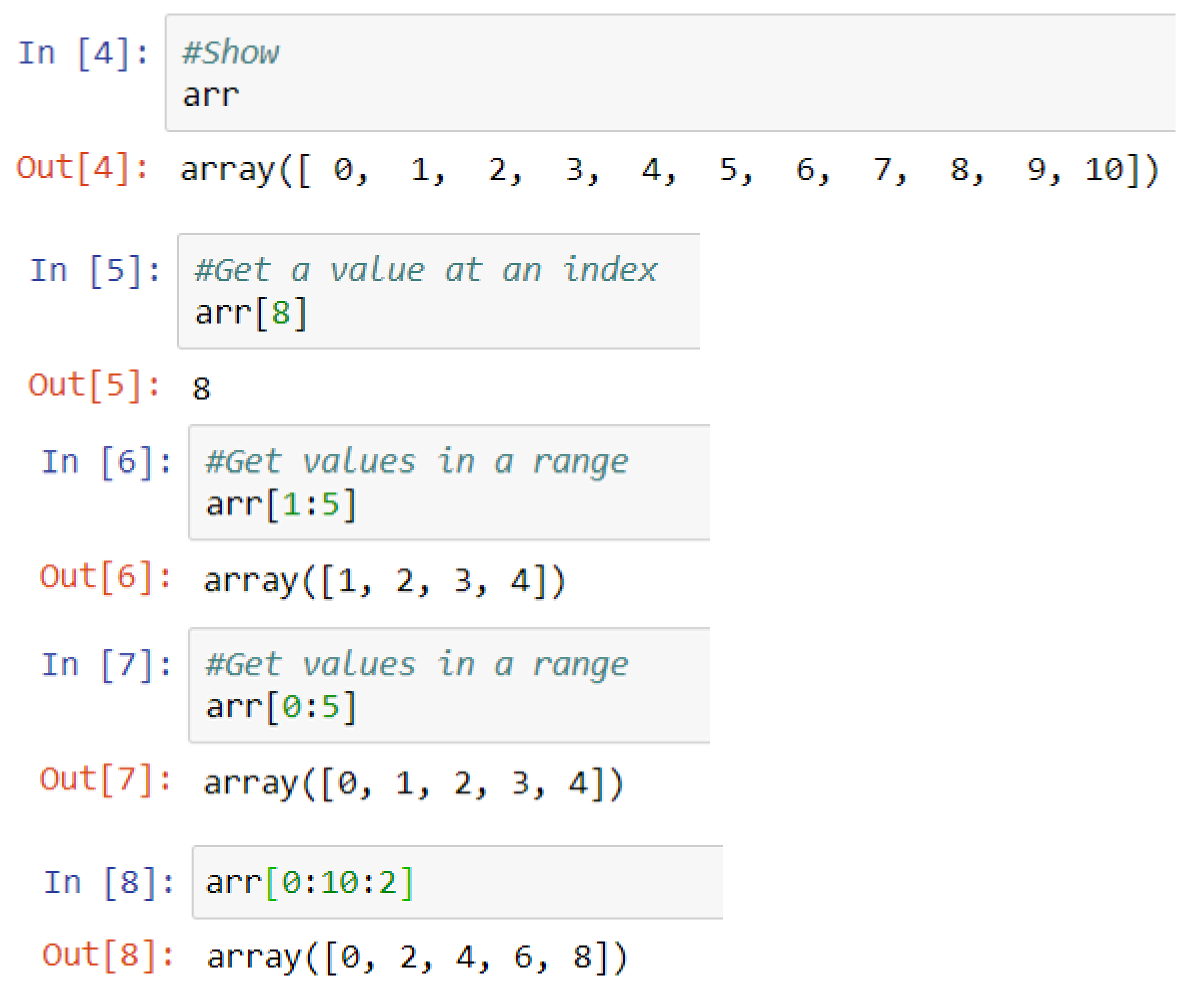
```
In [1]: import numpy as np
```

```
In [3]: #Creating sample array
        arr = np.arange(0,11)
```

```
In [4]: #Show
        arr
```

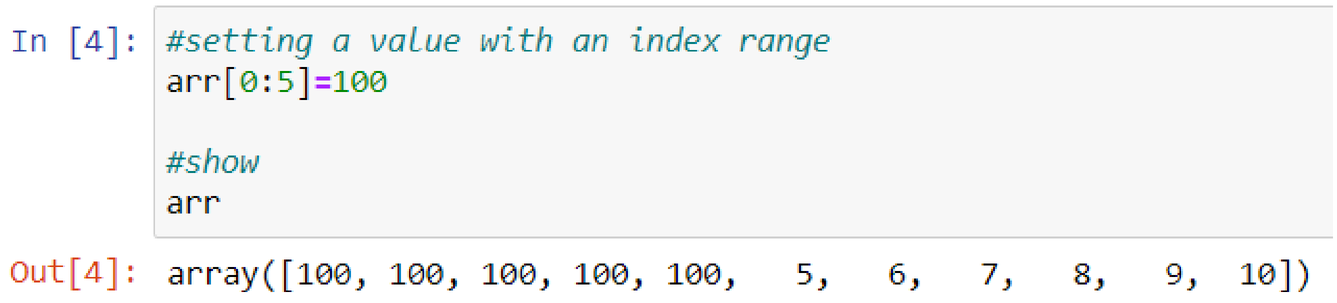
```
Out[4]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

Bracket Indexing and Selection

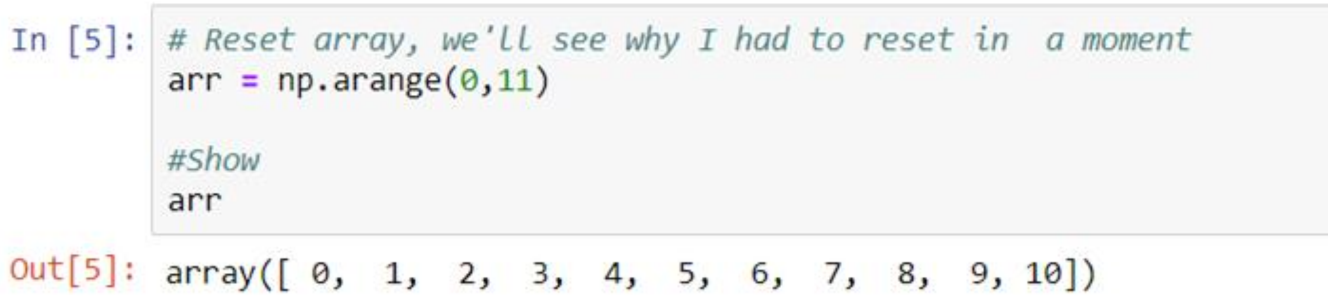


```
In [4]: #Show
        arr

Out[4]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [5]: #Get a value at an index
        arr[8]

Out[5]: 8

In [6]: #Get values in a range
        arr[1:5]

Out[6]: array([1, 2, 3, 4])

In [7]: #Get values in a range
        arr[0:5]

Out[7]: array([0, 1, 2, 3, 4])

In [8]: arr[0:10:2]

Out[8]: array([0, 2, 4, 6, 8])
```

Slicing

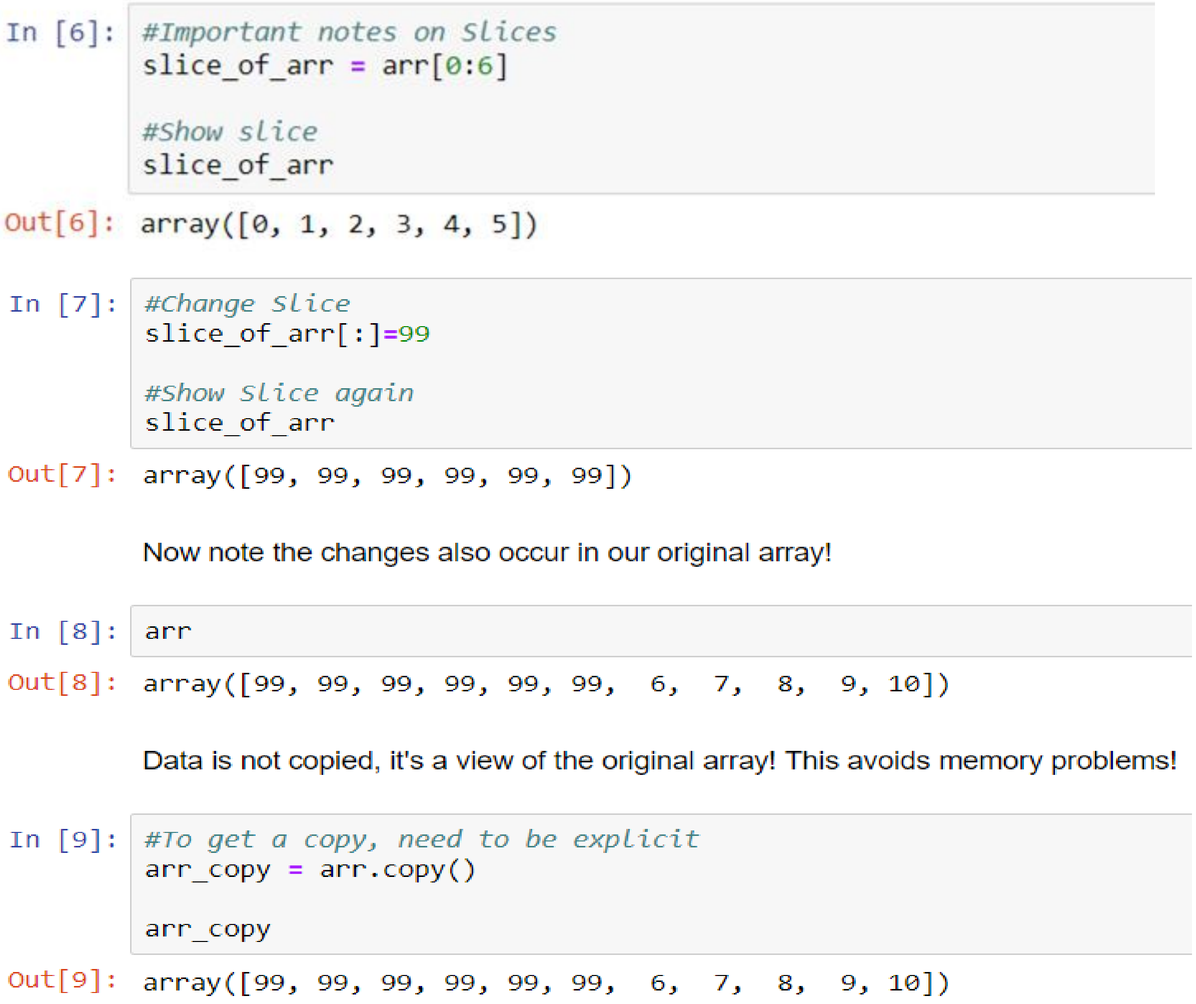


```
In [4]: #setting a value with an index range
        arr[0:5]=100

        #show
        arr

Out[4]: array([100, 100, 100, 100, 100,   5,   6,   7,   8,   9,  10])
```



```
In [5]: # Reset array, we'll see why I had to reset in  a moment
        arr = np.arange(0,11)

        #Show
        arr

Out[5]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

Slicing



```
In [6]:  #Important notes on Slices
         slice_of_arr = arr[0:6]

         #Show slice
         slice_of_arr

Out[6]:  array([0, 1, 2, 3, 4, 5])
```

```
In [7]:  #Change Slice
         slice_of_arr[:]=99

         #Show Slice again
         slice_of_arr

Out[7]:  array([99, 99, 99, 99, 99, 99])
```

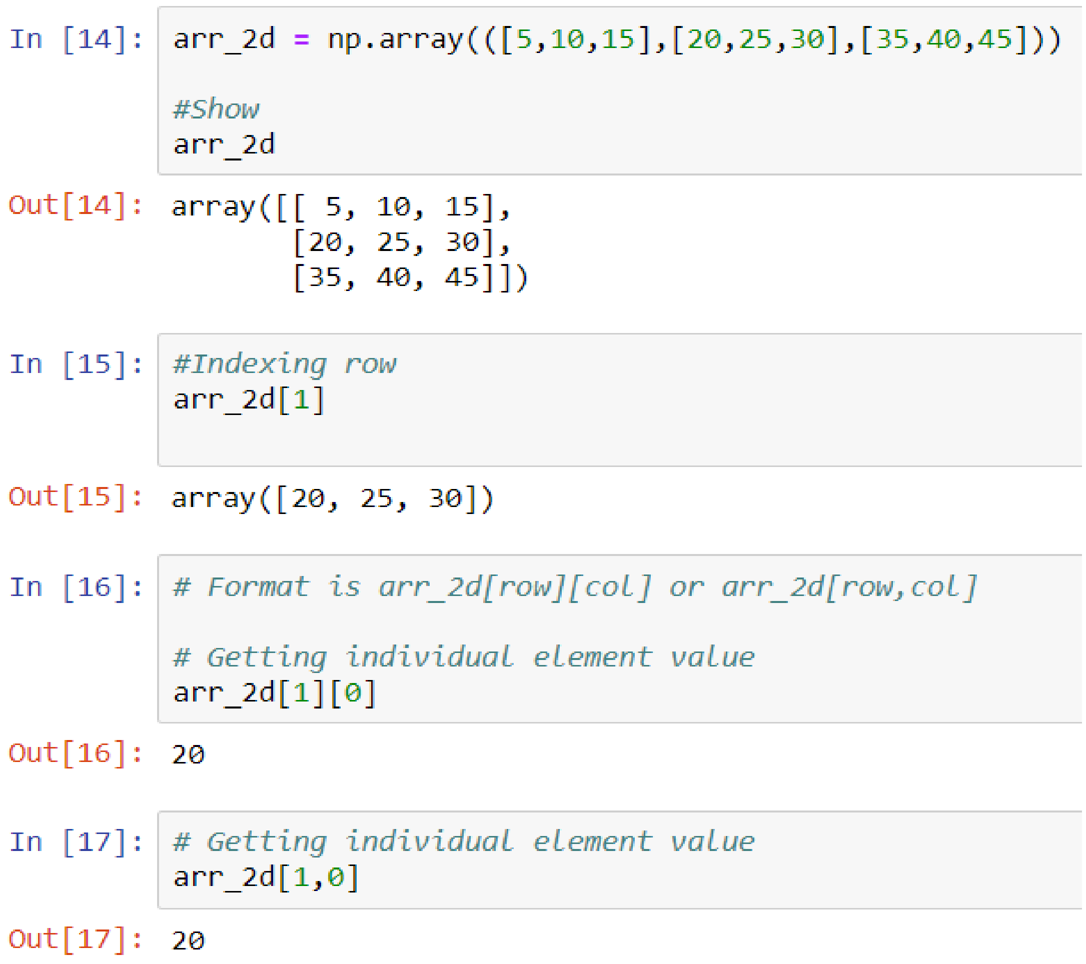
Now note the changes also occur in our original array!

```
In [8]:  arr

Out[8]:  array([99, 99, 99, 99, 99, 99,  6,  7,  8,  9, 10])
```

Data is not copied, it's a view of the original array! This avoids memory problems!

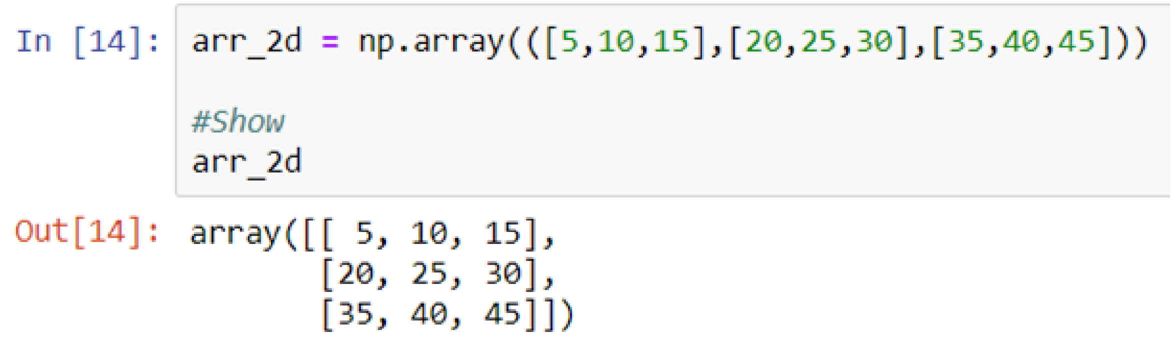
```
In [9]:  #To get a copy, need to be explicit
         arr_copy = arr.copy()

         arr_copy

Out[9]:  array([99, 99, 99, 99, 99, 99,  6,  7,  8,  9, 10])
```

Indexing a 2D array

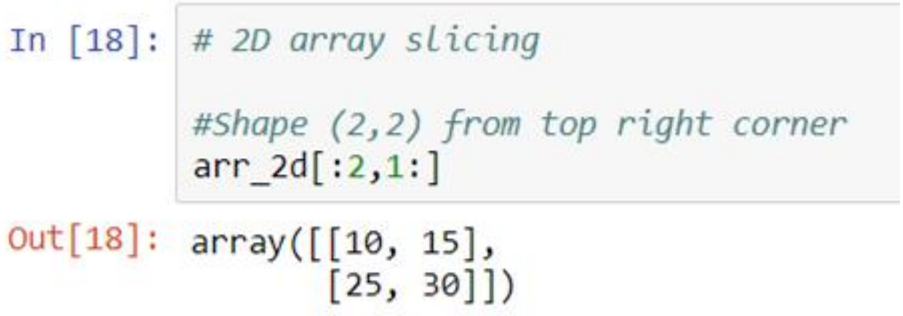


```
In [14]: arr_2d = np.array(([5,10,15],[20,25,30],[35,40,45]))

         #Show
         arr_2d

Out[14]: array([[ 5, 10, 15],
                [20, 25, 30],
                [35, 40, 45]])

In [15]: #Indexing row
         arr_2d[1]

Out[15]: array([20, 25, 30])

In [16]: # Format is arr_2d[row][col] or arr_2d[row,col]

         # Getting individual element value
         arr_2d[1][0]

Out[16]: 20

In [17]: # Getting individual element value
         arr_2d[1,0]

Out[17]: 20
```

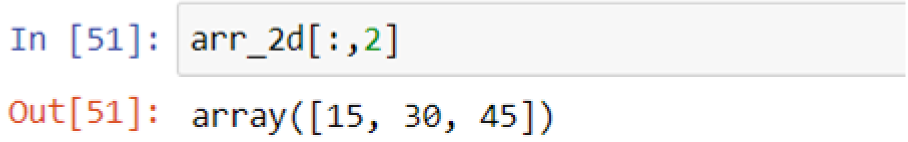
Indexing a 2D array



```
In [14]: arr_2d = np.array(([5,10,15],[20,25,30],[35,40,45]))

         #Show
         arr_2d

Out[14]: array([[ 5, 10, 15],
                [20, 25, 30],
                [35, 40, 45]])
```



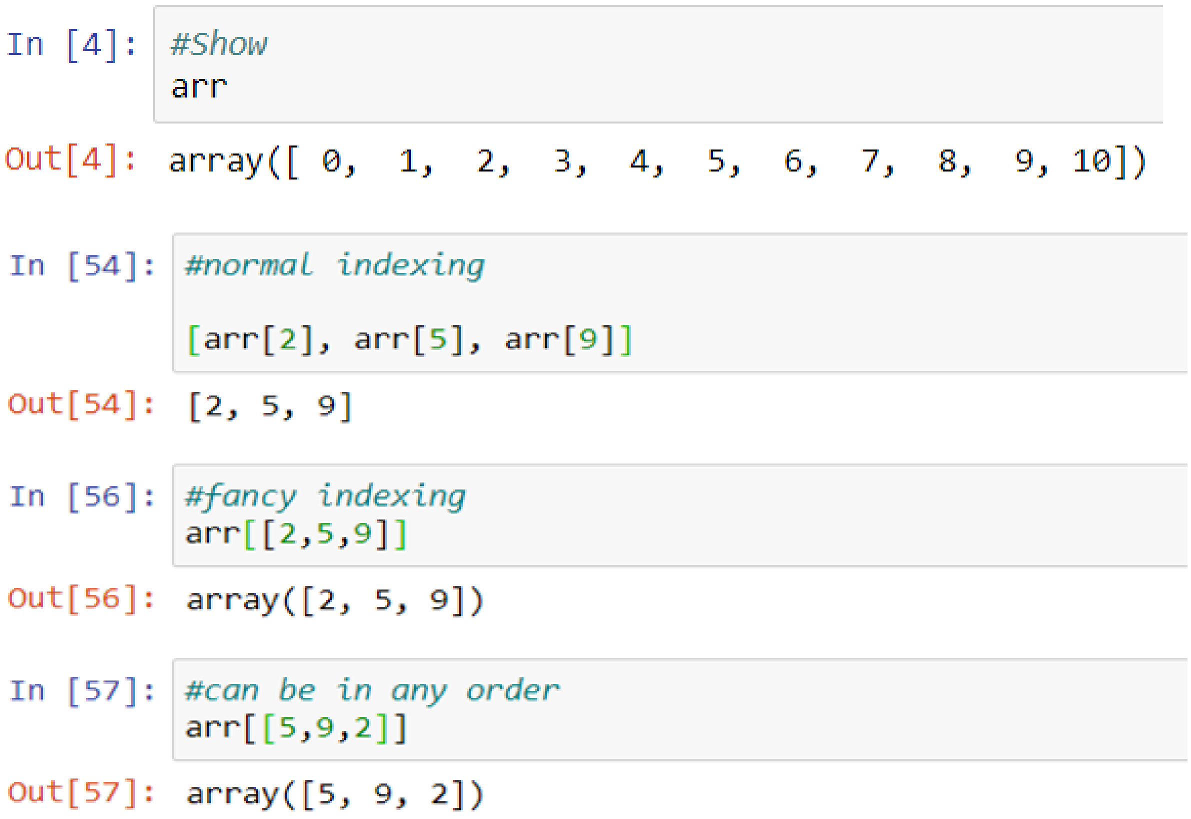
```
In [18]: # 2D array slicing

         #Shape (2,2) from top right corner
         arr_2d[:2,1:]

Out[18]: array([[10, 15],
                [25, 30]])
```



```
In [51]: arr_2d[:,2]

Out[51]: array([15, 30, 45])
```

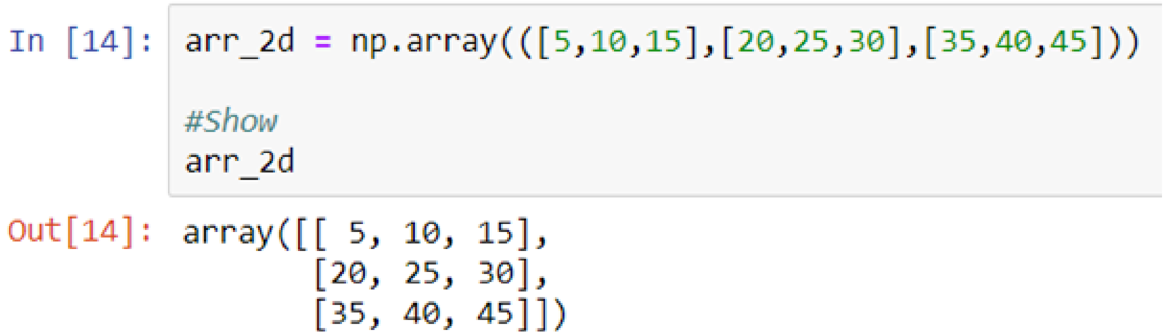
Fancy Indexing

Fancy Indexing

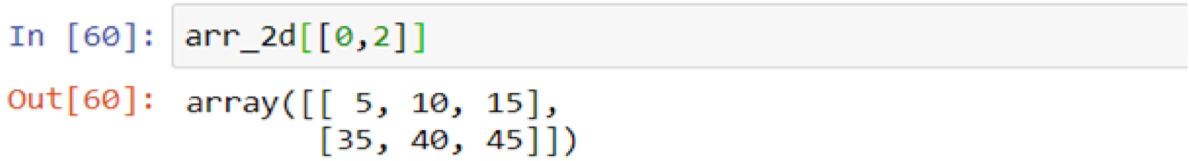


```
In [4]: #Show
        arr

Out[4]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [54]: #normal indexing

         [arr[2], arr[5], arr[9]]

Out[54]: [2, 5, 9]

In [56]: #fancy indexing
         arr[[2,5,9]]

Out[56]: array([2, 5, 9])

In [57]: #can be in any order
         arr[[5,9,2]]

Out[57]: array([5, 9, 2])
```

Fancy Indexing



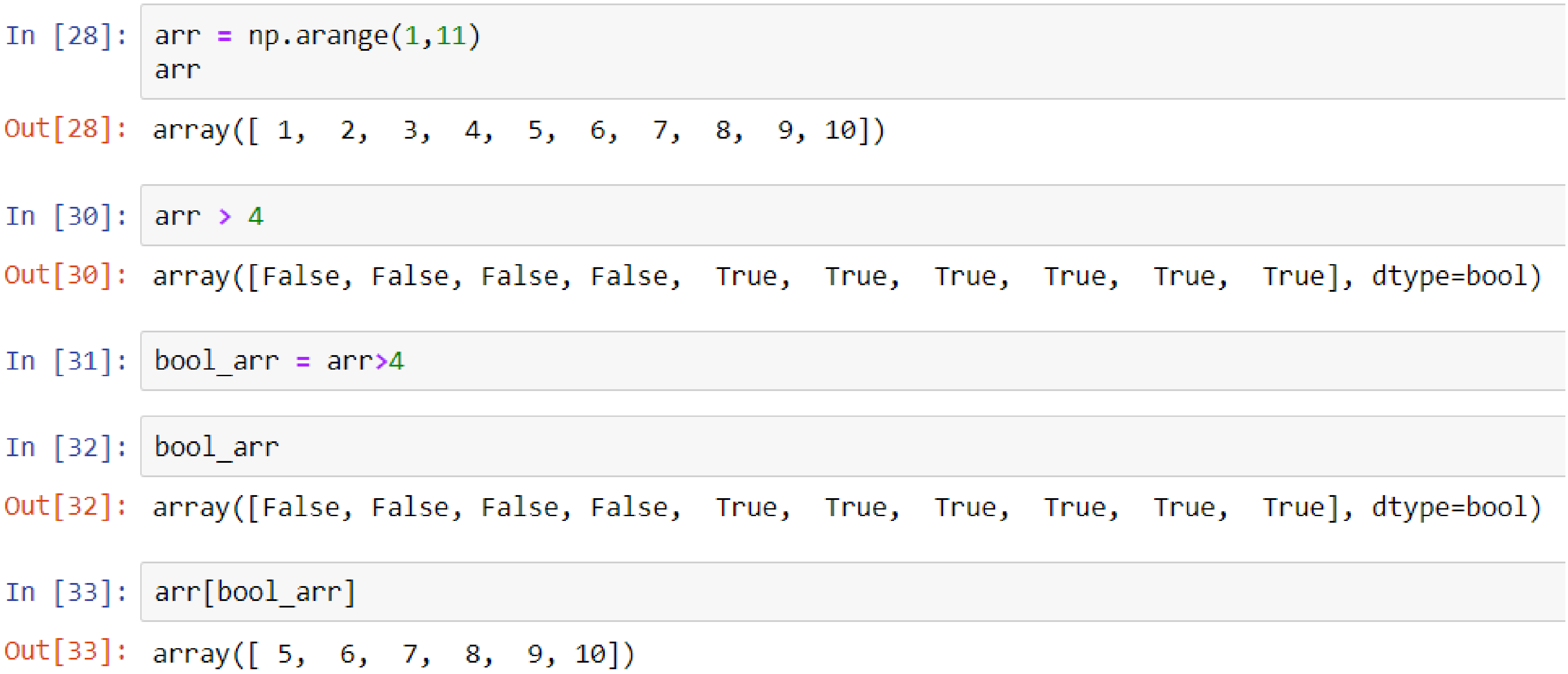
```
In [14]: arr_2d = np.array(([5,10,15],[20,25,30],[35,40,45]))

         #Show
         arr_2d

Out[14]: array([[ 5, 10, 15],
                [20, 25, 30],
                [35, 40, 45]])
```



```
In [60]: arr_2d[[0,2]]

Out[60]: array([[ 5, 10, 15],
                [35, 40, 45]])
```

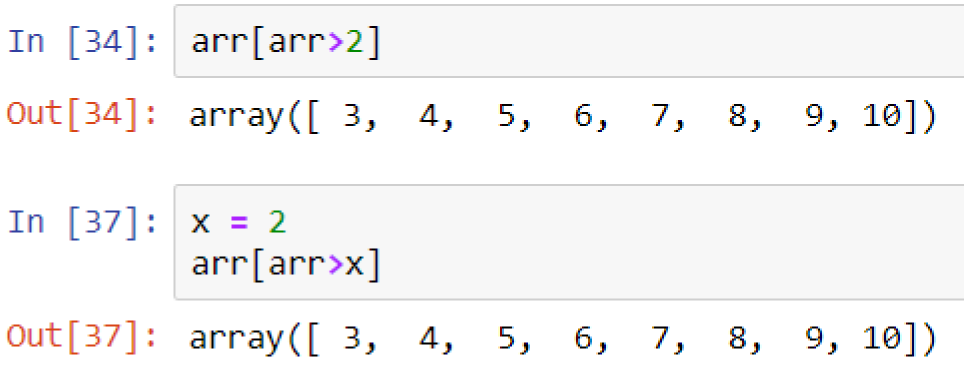
Conditional Selection

Conditional Selection



```
In [28]: arr = np.arange(1,11)
         arr

Out[28]: array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])

In [30]: arr > 4

Out[30]: array([False, False, False, False,  True,  True,  True,  True,  True,  True], dtype=bool)

In [31]: bool_arr = arr>4

In [32]: bool_arr

Out[32]: array([False, False, False, False,  True,  True,  True,  True,  True,  True], dtype=bool)

In [33]: arr[bool_arr]

Out[33]: array([ 5,  6,  7,  8,  9, 10])
```

Conditional Selection



```
In [34]: arr[arr>2]

Out[34]: array([ 3,  4,  5,  6,  7,  8,  9, 10])
```

```
In [37]: x = 2
         arr[arr>x]

Out[37]: array([ 3,  4,  5,  6,  7,  8,  9, 10])
```