Ex. No: 4

Aim: *Indexing and Slicing of NumPy Array.*

Theory:

- Python NumPy array slicing is used to extract some portion of data from the actual array.
- Slicing in python means extracting data from one given index to another given index, however, NumPy slicing is slightly different.
- Slicing can be done with the help of [:].
- A NumPy array slicing object is constructed by giving start, stop, and step parameters to the built-in slicing function.
- This slicing object is passed to the array to extract some portion of the array.

Syntax:

[start: stop: step]

Parameters:

- **Start:** This index by default considers as '0'
- **Stop:** This index considers as a length of the array.
- **Step:** By default, it is considered as '1'.

4 (A):

Slicing 1-D NumPy arrays:

Example:

```
import numpy as np
a=np.array([3, 5, 7, 9, 11, 15, 18, 22])
print("Use slicing a 1D arrays:")
b=a[1:6]
print(b)

print("Slice Starting from 3rd value to end:")
c=a[3:]
print(c)

print("Slice 0 to 4 index:")
d=a[:5]
print(d)

print("Use step value:")
e=a[1:6:2]
print€

print("Use step value:")
f=a[::2]
print(f)
```

Output:

```
C:\Users\MURALI\Desktop\DS\Lab\Exp 4>python 4a.py
Use slicing a 1D arrays:
[ 5  7  9 11 15]
Slice Starting from 3rd value to end:
[ 9 11 15 18 22]
Slice 0 to 4 index:
[ 3  5  7  9 11]
Use step value:
[ 5  9 15]
Use step value:
[ 3  7 11 18]

C:\Users\MURALI\Desktop\DS\Lab\Exp 4>
```

4 (B):

Slicing 2-D NumPy arrays:

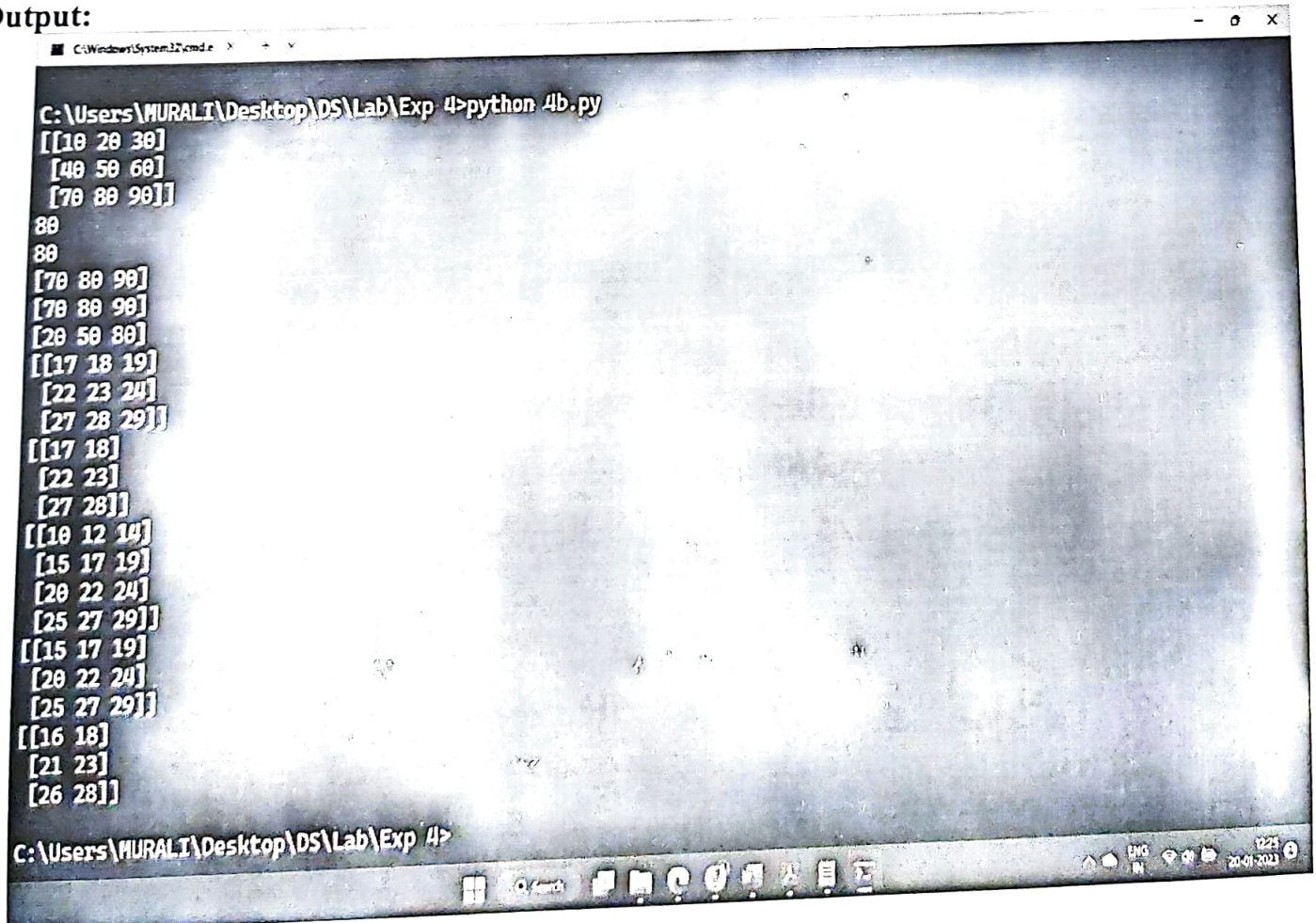
Example:

```
import numpy as np
a=np.array([[10, 20, 30],
            [40, 50, 60],
            [70, 80, 90]])
print(a)
print(a[2,1])
print(a[2][1])
print(a[2])
print(a[2,:])
print(a[:,1])

b = np.array([[10, 11, 12, 13, 14],
              [15, 16, 17, 18, 19],
              [20, 21, 22, 23, 24],
              [25, 26, 27, 28, 29]])

print(b[1:,2:])
print(b[1:,2:4])
print(b[:,::2])
print(b[1:,::2])
print(b[1:,1::2])
```

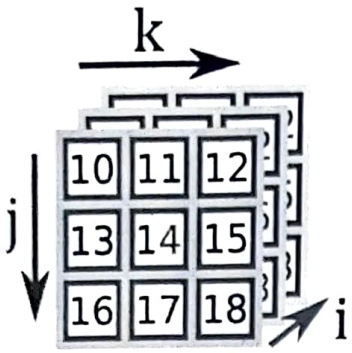
Output:



4 (C):

Slicing 3-D NumPy arrays:

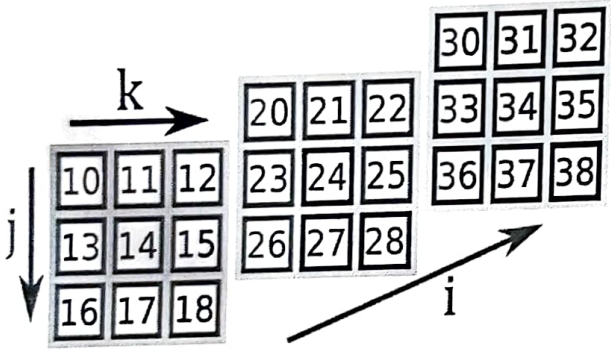
Here is a diagram of the array:



A 3D array is like a stack of matrices:

- The first index, i, selects the matrix
- The second index, j, selects the row
- The third index, k, selects the column

Here is the same diagram, spread out a bit so we can see the values:



Example:

```
import numpy as np
a=np.array([[[10, 11, 12], [13, 14, 15], [16, 17, 18]],
            [[20, 21, 22], [23, 24, 25], [26, 27, 28]],
            [[30, 31, 32], [33, 34, 35], [36, 37, 38]]])
print(a)
print("Indexing:")
print(a[2,0,1])
print(a[1,2])
print(a[0,:,1])
print(a[:,1,2])
print(a[2])
print(a[:,1])
print(a[:,:,0])
print("Slicing:")
```

```
print(a[1:,:2,:])
print(a[1:,:2])
print(a[:2,1:,:2])
print(a[1:,:,:])
#print(a[1:,:])
#print(a[1:])
```

Output:

```
C:\Windows\System32\cmd.e

C:\Users\MURALI\Desktop\DS\Lab\Exp 4>python 4c.py
[[[10 11 12]
  [13 14 15]
  [16 17 18]]

 [[20 21 22]
  [23 24 25]
  [26 27 28]]

 [[30 31 32]
  [33 34 35]
  [36 37 38]]]
Indexing:
31
[26 27 28]
[11 14 17]
[15 25 35]
[[30 31 32]
 [33 34 35]
 [36 37 38]]
[[13 14 15]
 [23 24 25]
 [33 34 35]]
[[10 13 16]
 [20 23 26]
 [30 33 36]]
Slicing:
[[[20 21 22]
  [23 24 25]]

 [[30 31 32]
  [33 34 35]]]
[[[20 21 22]
  [23 24 25]]

 [[30 31 32]
  [33 34 35]]]
[[[13 14]
  [16 17]]

 [[23 24]
  [26 27]]]
[[[20 21 22]
  [23 24 25]
  [26 27 28]]

 [[30 31 32]
  [33 34 35]
  [36 37 38]]]
```

4 (D):

Negative slicing of NumPy arrays:

- Negative slicing prints elements from the end rather than the beginning.
- Minus operator is used to refer to an index from the end.

Example:

```
import numpy as np
a = np.array([[1,2,3,4,5],
              [6,7,8,9,10]])
print(a)
print(a[:,-1])
print(a[:,-1:-3:-1])
print(a[:,-1:-4:-1])
print('Reversed array:')
print(a[::-1,::-1])
print('Reversed array vertically:')
print(np.flip(a,axis=1))
print('Reversed array horizontally:')
print(np.flip(a,axis=0))
```

Output: