NUMPY

SLICING

AND

INDEXING

Multi-Dimensional Arrays

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> import numpy as np
> X = np.arange(1,37).reshape(6,6)
array([[ 1,  2,  3,  4,  5,  6],
       [ 7,  8,  9, 10, 11, 12],
       [13, 14, 15, 16, 17, 18],
       [19, 20, 21, 22, 23, 24],
       [25, 26, 27, 28, 29, 30],
       [31, 32, 33, 34, 35, 36]])
```

Transposing Arrays

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> import numpy as np
> X = np.arange(1,37).reshape(6,6)
> Y = X.transpose()
array([[ 1,  7, 13, 19, 25, 31],
       [ 2,  8, 14, 20, 26, 32],
       [ 3,  9, 15, 21, 27, 33],
       [ 4, 10, 16, 22, 28, 34],
       [ 5, 11, 17, 23, 29, 35],
       [ 6, 12, 18, 24, 30, 36]])
```

Reshaping Arrays

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> import numpy as np
> X = np.arange(1,37).reshape(6,6)
> Y = X.reshape(4, 9)
array([[ 1,  2,  3,  4,  5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14, 15, 16, 17, 18],
       [19, 20, 21, 22, 23, 24, 25, 26, 27],
       [28, 29, 30, 31, 32, 33, 34, 35, 36]])
```

Individual Element

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1,2]
9
> Y.shape
()
```

- another way: $Y = x[1][2]$

A Single Row

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[2, :]
array([13, 14, 15, 16, 17, 18])
> Y.shape
(6,)
```

- other possibilities:

```
> Y = X[2, ]
> Y = X[2]
```

A List of Rows

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[[1,3], :]
array([[ 7,  8,  9, 10, 11, 12],
       [19, 20, 21, 22, 23, 24]])
> Y.shape
(2, 6)
```

- other possibilities:

```
> Y = X[[1,3], ]
> Y = X[[1,3]]
```

A Range of Rows

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1: , : ]
array([[ 7,  8,  9, 10, 11, 12],
       [13, 14, 15, 16, 17, 18],
       [19, 20, 21, 22, 23, 24],
       [25, 26, 27, 28, 29, 30],
       [31, 32, 33, 34, 35, 36]])
> Y.shape
(5, 6)
```

- other possibilities:

```
> Y = X[1:, ] ;     Y = X[1:]
```

A (sub)Range of Rows

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1::2 , : ]
array([[ 7,  8,  9, 10, 11, 12],
       [19, 20, 21, 22, 23, 24],
       [31, 32, 33, 34, 35, 36]])
> Y.shape
(3, 6)
```

- other possibilities:

```
> Y = X[1::2, ]
> Y = X[1::2]
```

A Single Column

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[:, 4]
array([ 5, 11, 17, 23, 29, 35])
> Y.shape
(6,)
```

A Single Column (cont'd)

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[:, [4]]
array([[ 5],
       [11],
       [17],
       [23],
       [29],
       [35]])
> Y.shape
(6, 1)
```

A List of Columns

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[:, [3,4]]
array([[ 4,  5],
       [10, 11],
       [16, 17],
       [22, 23],
       [28, 29],
       [34, 35]])
> Y.shape
(6, 2)
```

A Range of Columns

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[:, 1:6:2]
array([[ 2,  4,  6],
       [ 8, 10, 12],
       [14, 16, 18],
       [20, 22, 24],
       [26, 28, 30],
       [32, 34, 36]])
> Y.shape
(6, 3)
```

Row/Column Slicing

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[2: , 1:  ]
array([[14, 15, 16, 17, 18],
       [20, 21, 22, 23, 24],
       [26, 27, 28, 29, 30],
       [32, 33, 34, 35, 36]])
> Y.shape
(4, 5)
```

Row/Column Slicing (cont'd)

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1:3, 1:3]
array([[ 8,  9],
       [14, 15]])
> Y.shape
(2, 2)
```

Row/Column Slicing (cont'd)

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1::2, 1:6:2]
array([[ 8, 10, 12],
       [20, 22, 24],
       [32, 34, 36]])
> Y.shape
(3, 3)
```

Reverse Slicing

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[1::2, 5::-2]
array([[12, 10,  8],
       [24, 22, 20],
       [36, 34, 32]])
> Y.shape
(3, 3)
```

Reversing Rows/Cols

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

```
> Y = X[ ::-1, ::-1]
array([[36, 35, 34, 33, 32, 31],
       [30, 29, 28, 27, 26, 25],
       [24, 23, 22, 21, 20, 19],
       [18, 17, 16, 15, 14, 13],
       [12, 11, 10,  9,  8,  7],
       [ 6,  5,  4,  3,  2,  1]])
> Y.shape
(6, 6)
```

Concepts Check:

(a) reshaping

(b) transposing arrays

(c) element selection

(d) row(s), column(s) selection

(e) row(s), column(s) slicing

(f) row(s), column(s) reversing