



**NUMPY**

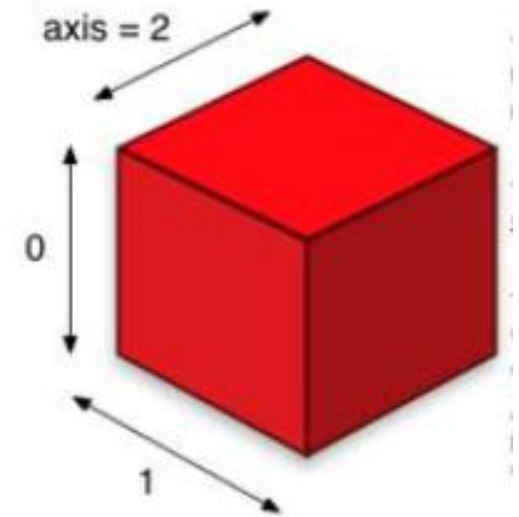
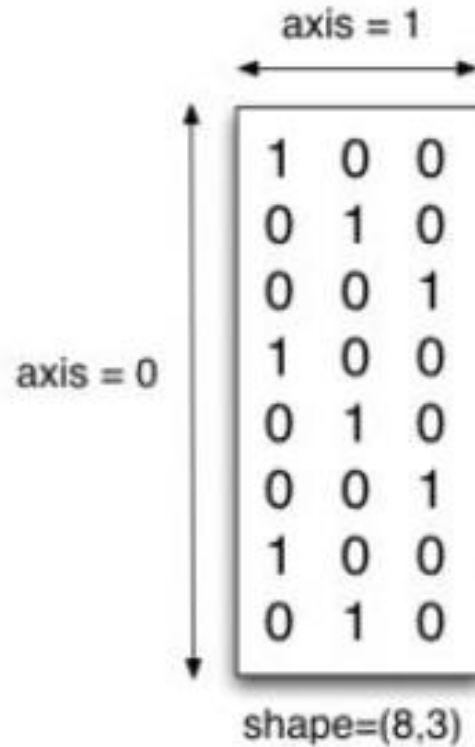
# ВВЕДЕНИЕ

Библиотека для работы с  
**многомерными массивами**

Все элементы массива  
принадлежат одному типу  
данных и последовательно  
располагаются в ОП

Выигрыш в скорости, по  
сравнению со списками

Код становится компактным  
и структурированным



# ТИП ДАННЫХ - ARRAY

import numpy as np

```
arr = np.array([0, 1, 2, 3, 4, 5,  
6, 7, 8])  
print arr  
arr[3:8] = -99  
print arr
```

OUTPUT

- [0 1 2 3 4 5 6 7 8]
- [ 0 1 2 -99 -99 -99 -99 -99 8]

# БЫСТРАЯ ПОЭЛЕМЕНТНАЯ ОБРАБОТКА

- `arr = np.arange(5)`
- `print arr`
- `print np.sqrt(arr)`
- `print np.exp(arr)`

**[0 1 2 3 4]**

**[ 0. 1. 1.4142 1.7321 2. ]**

**[ 1. 2.7183 7.3891 20.0855 54.5982]**

# ПОИСК МАКСИМУМА

- `x = randn(4)`
- `y = randn(4)`
- `print x`
- `print y`
- `print np.maximum(x, y)`
- `[-0.9691 -1.4411 1.2614 -0.9615]`
- `[-0.0398 -0.0692 -1.6854 -0.3902]`
- `[-0.0398 -0.0692 1.2614 -0.3902]`

# СЛОЖЕНИЕ

- `x = randn(4)`
- `y = randn(4)`
- `print x`
- `print y`
- `print np.add(x, y)`
- `[ 0.0987 -1.2579 -1.4827 -1.4299]`
- `[-0.2855 -0.7548 -1.0134 0.7546]`
- `[-0.1868 -2.0127 -2.4961 -0.6753]`

$x+y$

# ФИЛЬТРАЦИЯ ДАННЫХ

- `a = [1, 2, 3]`
- `b = [10, 20, 30]`
- `c = [True, False, True]`
- `result = [(x if z else y)`
- `for x, y, z in`  
`zip(a, b, c)]`
- `print result`
- `OUTPUT`
- `[1, 20, 3]`

- `a = [1, 2, 3]`
- `b = [10, 20, 30]`
- `c = [True, False, True]`
- `np.where(c, a, b)`
- `Output is [ 1 20 3]`

# WHERE

- `arr = np.random.rand(3, 2)`
- `print arr`
- `print np.where(arr > 0.5, 2, -2)`

```
[[ 0.44292516  0.68852318]
 [ 0.52094621  0.98764891]
 [ 0.19472641  0.38844208]]
```

```
[[ -2  2]
 [ 2  2]
 [ -2 -2]]
```



# МАТЕМАТИЧЕСКИЕ И СТАТИСТИЧЕСКИЕ МЕТОДЫ

- `arr = np.random.randn(2, 2)`
- `print arr`
- `print arr.mean()`
- `print np.mean(arr)`
- `print arr.sum()`

# МЕТОДЫ ПО СТРОКАМ И СТОЛБЦАМ

- `arr = np.array([[0, 1, 2], [3, 4, 5], [6, 7, 8]])`
  - `print arr` **[[0 1 2]**
  - `print arr.mean(axis=0)` **[3 4 5]**
  - `print arr.mean(axis=1)` **[6 7 8]**
- [ 3. 4. 5.]**
- [ 1. 4. 7.]**

# МЕТОДЫ ПО СТРОКАМ И СТОЛБЦАМ

- `arr = np.array([[0, 1, 2], [3, 4, 5], [6, 7, 8]])`

- `print arr`

`[[0 1 2]`

- `print arr.sum(0)`

`[3 4 5]`

- `print arr.sum(1)`

`[6 7 8]]`

`[ 9 12 15]`

`[ 3 12 21]`

# СОВОКУПНАЯ СУММА

- `arr = np.array(`
- `[[0, 1, 2],`
- `[3, 4, 5], [6, 7, 8]])`
- `print arr`
- `print arr.cumsum(0)`
- `print arr.cumsum(1)`

```
[[0 1 2]  
 [3 4 5]  
 [6 7 8]]
```

```
[[ 0  1  2]  
 [ 3  5  7]  
 [ 9 12 15]]
```

```
[[ 0  1  3]  
 [ 3  7 12]  
 [ 6 13 21]]
```

# MACCIB BOOLEAN

```
import numpy as np
arr=np.random.randn(5)
print(arr)
print ((arr>0).sum())
```

```
[-1.71931388  0.48652174 -0.12731192  0.04951212  0.87630357]
3
```

```
bools = np.array(
[False, False, True, False])
print bools.any()
print bools.all()
```

**output**

**True**

**False**

# СОРТИРОВКА

- `arr = randn(4)`
- `print arr`
- `arr.sort()`
- `print arr`
- **OUTPUT**
- `[-0.301 -0.1785 -0.9659 -0.6087]`
- `[-0.9659 -0.6087 -0.301 -0.1785]`

```
arr = randn(2, 3)
print arr
arr.sort(0)
print arr
arr.sort(1)
print arr
```

```
[[1 3 2]  [[1 3 2]  [[1 2 3]
 [8 4 9]  [3 4 8]  [3 4 8]
 [3 5 8]] [8 5 9]] [5 8 9]]
```

# УНИКАЛЬНЫЕ ЗНАЧЕНИЯ

```
names = np.array(['Bob', 'Joe',  
                  'Will', 'Bob', 'Will', 'Joe',  
                  'Joe'])  
print np.unique(names)  
print sorted(set(names))
```

- **OUTPUT**

- ['Bob' 'Joe' 'Will']
- ['Bob', 'Joe', 'Will']



# ВКЛЮЧАЕТ

```
values = np.array([6,0,0,3,2,5,6])  
print(np.isin(values, [2,3,6]))
```

```
[ True False False  True  True False  True]
```

# РАБОТА С ФАЙЛОМ

- `arr = np.arange(10)`
- `print arr`
- `np.save('some_array', arr)`
- `arr1 = np.load('some_array.npy')`
- `print arr1`

- `arr = np.loadtxt('array_ex.txt', delimiter=',')`

- `print arr`
- `print type(arr)`

*array\_ex.txt*

1,2,3,4

3,4,5,6

## OUTPUT

`[[ 1. 2. 3. 4.]`

`[ 3. 4. 5. 6.]]`

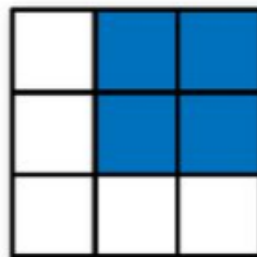
`<type`

`'numpy.ndarray'>`



# СРЕЗЫ

		axis 1		
		0	1	2
axis 0	0	0,0	0,1	0,2
	1	1,0	1,1	1,2
	2	2,0	2,1	2,2



Expression

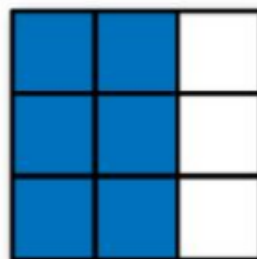
`arr[:2, 1:]`



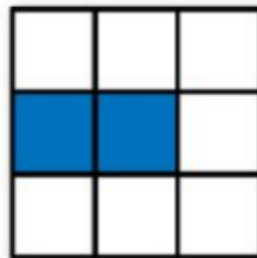
`arr[2]`

`arr[2, :]`

`arr[2:, :]`



`arr[:, :2]`



`arr[1, :2]`

`arr[1:2, :2]`

## 3d 2x2x2

```
a=np.array([  
    [  
        [3, 1], [4, 3]  
    ],  
    [  
        [2, 4], [3, 3]  
    ]  
])
```

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

2,4  
3,1  
3,3  
4,3  
a[0][1]

2,4  
3,1  
3,3  
4,3  
a[1]

2,4  
3,1  
3,3  
4,3  
a[0][0][0]

# СРЕЗЫ ДЛЯ ТРЕХМЕРНОГО МАССИВА

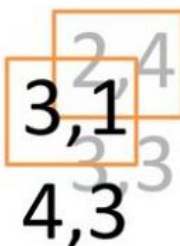
```
a=np.array([
```

```
[ [3, 1], [4, 3]
```

```
],
```

```
[ [2, 4], [3, 3]
```

```
])
```

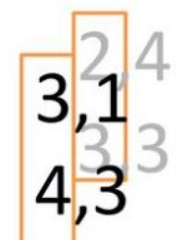
  
`a[:, 0]`

```
a=np.array([
```

```
[ [3, 1], [4, 3]
```

```
[ [2, 4], [3, 3]
```

```
])
```

  
`a[:, :, 0]`