1.创建数组

```
import random
import numpy as np
t1=np.array([1,2,3])
print(t1)
print(type(t1))

t2=np.array(range(10))
print(t2)
print(type(t2))

t3=np.arange(12)
print(t3)
print(t3.dtype)

t4=np.arange(4,10,2)
print(t4)

#numpy中的数据类型
t5=np.array(range(1,4),dtype="float32")
print(t4)
print(t4.dtype)
#布尔类型
t6=np.array([1,1,0,1,0,0],dtype=bool)
print(t6)
print(t6.dtype)

#调整数据类型
t7=t6.astype("int32")
print(t6)
print(t6.dtype)

#numpy中的小数
t8=np.array([random.random() for i in range(10)])
print(t8)
print(t8.dtype)

t9=np.round(t8,2)#保留两位小数
```

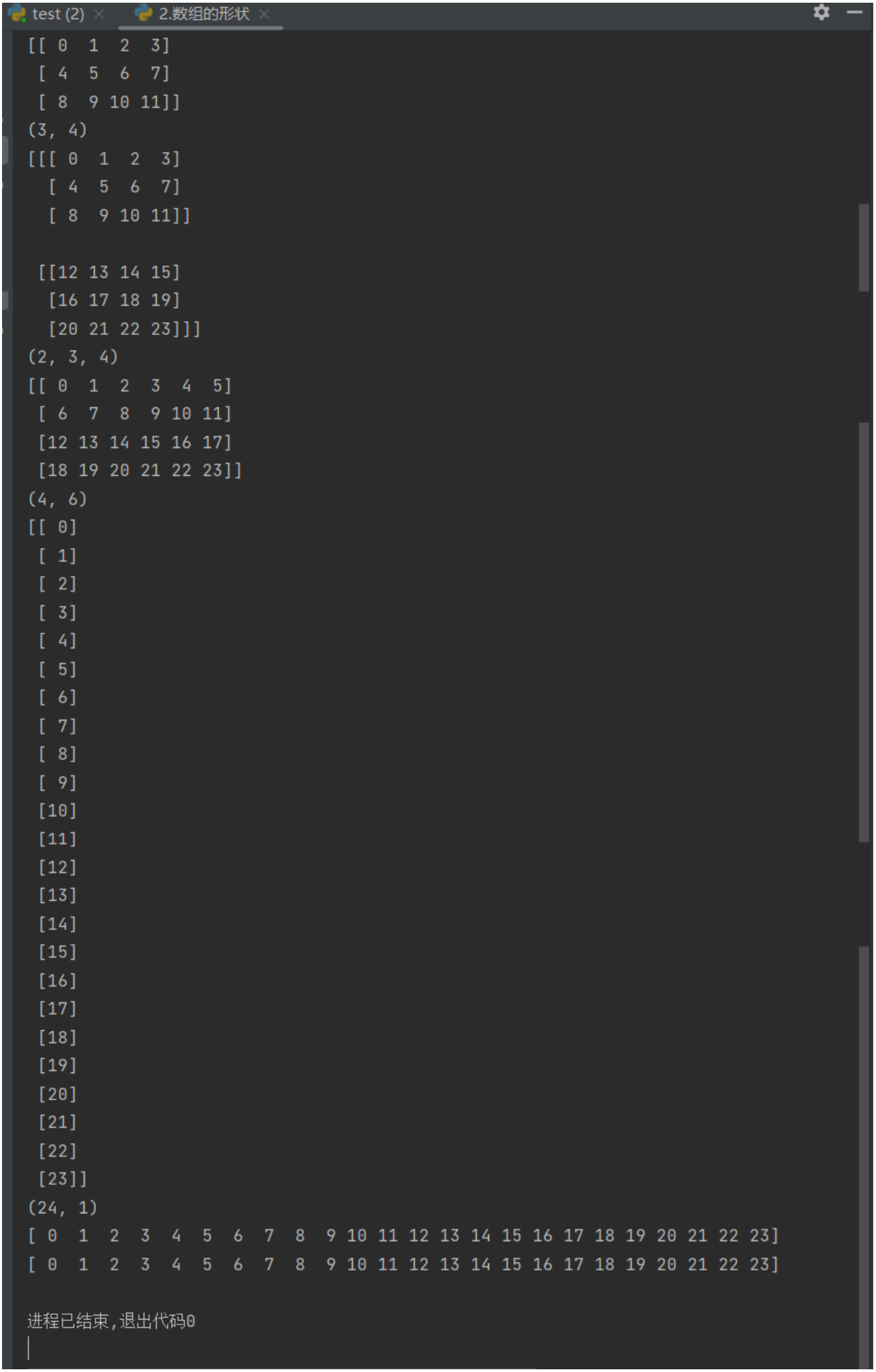
```
print(t9)
print(t9.dtype)
```

```
F:\BaiduSyncdisk\学习\Python\newpython\Scripts\python.exe F:/BaiduSyncdisk/学习/Pyt
[1 2 3]
<class 'numpy.ndarray'>
[0 1 2 3 4 5 6 7 8 9]
<class 'numpy.ndarray'>
[ 0  1  2  3  4  5  6  7  8  9 10 11]
int32
[4 6 8]
[4 6 8]
int32
[ True  True False  True False False]
bool
[ True  True False  True False False]
bool
[0.66228104 0.71366309 0.87746225 0.97084478 0.22733161 0.41907021
 0.42464199 0.78656068 0.04700692 0.94956049]
float64
[0.66 0.71 0.88 0.97 0.23 0.42 0.42 0.79 0.05 0.95]
float64

进程已结束,退出代码0
```

2.数组的形状

```
import numpy as np
t1=np.arange(12)
print(t1)    #[ 0  1  2  3  4  5  6  7  8  9 10 11]
print(t1.shape)  #(12,) -- 一维数组元素个数

t2=np.array([[1,2,3],[4,5,6]])
print(t2)  #[[1 2 3][4 5 6]]
print(t2.shape)  #(2, 3) -- 二维数组行列

t3=np.array([[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]])
print(t3)
'''[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
```

```
  [10 11 12]]]'''
print(t3.shape)  #(2, 2, 3) -- 三维矩阵 -- 块数行列


#修改数组形状
t4=np.arange(12).reshape(3,4) #转二维数组
print(t4)
print(t4.shape) #(3, 4)

t5=np.arange(24).reshape(2,3,4) #转三维数组
print(t5)
print(t5.shape) #(2, 3, 4)

t5=np.arange(24).reshape(4,6) #转二维数组
print(t5)
print(t5.shape) #(4, 6)

t5=np.arange(24).reshape(24,1) #转二维数组
print(t5)
print(t5.shape) #(24, 1)

t6=t5.reshape((t5.shape[0]*t5.shape[1],))  #转一维数组
print(t6)  #转成一维数组

#将多维数组直接转成一维数组
t7=t5.flatten()
print(t7)
```


test (2)
2.数组的形状
[[0 1 2 3]
 [4 5 6 7]
 [8 9 10 11]]
(3, 4)
[[[0 1 2 3]
  [4 5 6 7]
  [8 9 10 11]]
 [[12 13 14 15]
  [16 17 18 19]
  [20 21 22 23]]]
(2, 3, 4)
[[0 1 2 3 4 5]
 [6 7 8 9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]]
(4, 6)
[[0]
 [1]
 [2]
 [3]
 [4]
 [5]
 [6]
 [7]
 [8]
 [9]
 [10]
 [11]
 [12]
 [13]
 [14]
 [15]
 [16]
 [17]
 [18]
 [19]
 [20]
 [21]
 [22]
 [23]]
(24, 1)
[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23]
[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23]
进程已结束,退出代码0

3.数组的计算

```python
import numpy as np
t5=np.arange(24).reshape(4,6) #转二维数组
print(t5)
print(t5.shape) #(4, 6)

print(t5+2)
'''
[[ 2  3  4  5  6  7]
 [ 8  9 10 11 12 13]
 [14 15 16 17 18 19]
 [20 21 22 23 24 25]]
'''
print(t5*2)
'''
[[ 0  2  4  6  8 10]
 [12 14 16 18 20 22]
 [24 26 28 30 32 34]
 [36 38 40 42 44 46]]
'''
print(t5/2)
'''
[[ 0.   0.5  1.   1.5  2.   2.5]
 [ 3.   3.5  4.   4.5  5.   5.5]
 [ 6.   6.5  7.   7.5  8.   8.5]
 [ 9.   9.5 10.  10.5 11.  11.5]]
'''
print(t5/0)
'''
[[nan inf inf inf inf inf]
 [inf inf inf inf inf inf]
 [inf inf inf inf inf inf]
 [inf inf inf inf inf inf]]
'''
t6=np.arange(100,124).reshape((4,6))
print(t6)
print(t6+t5)
'''
[[100 102 104 106 108 110]
```

```
 [112 114 116 118 120 122]
 [124 126 128 130 132 134]
 [136 138 140 142 144 146]]
'''
print(t6*t5)
'''
[[   0  101  204  309  416  525]
 [ 636  749  864  981 1100 1221]
 [1344 1469 1596 1725 1856 1989]
 [2124 2261 2400 2541 2684 2829]]
'''
print(t6/t5)
'''
[[         inf 101.          51.          34.33333333  26.
   21.        ]
 [ 17.66666667  15.28571429  13.5         12.11111111  11.
   10.09090909]
 [  9.33333333   8.69230769   8.14285714   7.66666667   7.25
    6.88235294]
 [  6.55555556   6.26315789   6.           5.76190476   5.54545455
    5.34782609]]
'''
t7=np.arange(0,6)
print(t5)
'''
[[ 0  1  2  3  4  5]
 [ 6  7  8  9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]]
'''
print(t7)
'''
[0 1 2 3 4 5]
'''
print(t5-t7)
'''
[[ 0  0  0  0  0  0]
 [ 6  6  6  6  6  6]
 [12 12 12 12 12 12]
 [18 18 18 18 18 18]]
```

```
'''
t8=np.arange(4).reshape((4,1))
print(t5-t8)
'''
[[ 0  1  2  3  4  5]
 [ 5  6  7  8  9 10]
 [10 11 12 13 14 15]
 [15 16 17 18 19 20]]
'''
t9=np.array(10);
print(t9)
print(t5-t9) #报错，行列都不一样，无法计算
```

```
 [ 6  7  8  9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]]
(4, 6)
[[ 2  3  4  5  6  7]
 [ 8  9 10 11 12 13]
 [14 15 16 17 18 19]
 [20 21 22 23 24 25]]
[[ 0  2  4  6  8 10]
 [12 14 16 18 20 22]
 [24 26 28 30 32 34]
 [36 38 40 42 44 46]]
[[ 0.   0.5  1.   1.5  2.   2.5]
 [ 3.   3.5  4.   4.5  5.   5.5]
 [ 6.   6.5  7.   7.5  8.   8.5]
 [ 9.   9.5 10.  10.5 11.  11.5]]
[[nan inf inf inf inf inf]
 [inf inf inf inf inf inf]
 [inf inf inf inf inf inf]
 [inf inf inf inf inf inf]]
[[100 101 102 103 104 105]
 [106 107 108 109 110 111]
 [112 113 114 115 116 117]
 [118 119 120 121 122 123]]
[[100 102 104 106 108 110]
 [112 114 116 118 120 122]
 [124 126 128 130 132 134]
 [136 138 140 142 144 146]]
[[   0  101  204  309  416  525]
 [ 636  749  864  981 1100 1221]
 [1344 1469 1596 1725 1856 1989]
 [2124 2261 2400 2541 2684 2829]]
[[         inf 101.          51.          34.33333333  26.
```

```
   21.        ]
 [ 17.66666667  15.28571429  13.5         12.11111111  11.
   10.09090909]
 [  9.33333333   8.69230769   8.14285714   7.66666667   7.25
    6.88235294]
 [  6.55555556   6.26315789   6.           5.76190476   5.54545455
    5.34782609]]
[[ 0  1  2  3  4  5]
 [ 6  7  8  9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]]
[0 1 2 3 4 5]
F:\BaiduSyncdisk\学习\Python\Python数据分析与可视化\numpy\1.数组\3.数组的计算.py:28: Runt
  print(t5/0)
F:\BaiduSyncdisk\学习\Python\Python数据分析与可视化\numpy\1.数组\3.数组的计算.py:28: Runt
  print(t5/0)
F:\BaiduSyncdisk\学习\Python\Python数据分析与可视化\numpy\1.数组\3.数组的计算.py:51: Runt
  print(t6/t5)
[[ 0  0  0  0  0  0]
 [ 6  6  6  6  6  6]
 [12 12 12 12 12 12]
 [18 18 18 18 18 18]]
[[ 0  1  2  3  4  5]
 [ 5  6  7  8  9 10]
 [10 11 12 13 14 15]
 [15 16 17 18 19 20]]
10
[[-10  -9  -8  -7  -6  -5]
 [ -4  -3  -2  -1   0   1]
 [  2   3   4   5   6   7]
 [  8   9  10  11  12  13]]

进程已结束,退出代码0
```

4.转置

```python
import numpy as np

t=np.arange(24).reshape((4,6))
print(t)
print(t.transpose())
print(t.T)
print(t.swapaxes(1,0)) #1和0轴交换
```

```
F:\BaiduSyncdisk\学习\Python\newpython\Scripts\python.exe F:/BaiduSyncdisk/学习/Pyth
[[ 0  1  2  3  4  5]
 [ 6  7  8  9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]]
[[ 0  6 12 18]
 [ 1  7 13 19]
 [ 2  8 14 20]
 [ 3  9 15 21]
 [ 4 10 16 22]
 [ 5 11 17 23]]
[[ 0  6 12 18]
 [ 1  7 13 19]
 [ 2  8 14 20]
 [ 3  9 15 21]
 [ 4 10 16 22]
 [ 5 11 17 23]]
[[ 0  6 12 18]
 [ 1  7 13 19]
 [ 2  8 14 20]
 [ 3  9 15 21]
 [ 4 10 16 22]
 [ 5 11 17 23]]

进程已结束,退出代码0
```

5.索引和切片

```python
import numpy as np

#****************** 一维数组 **********************
print("*"*40,"一维数组","*"*40)
arr = np.arange(10)
print("*"*20,"print(arr)","*"*20)
print(arr)
print("*"*20,"print(arr[5])","*"*20)
print(arr[5]) #索引为5的元素
print("*"*20,"print(arr[5:8])","*"*20)
print(arr[5:8]) #索引为5 - 8的元素

#改变索引为5 - 8的元素的值
arr[5:8]=12
print("*"*20,"print(arr)","*"*20)
print(arr)
```

```
17
18 #视图上的任何修改都会直接反映到源数组上
19 arr_slice=arr[5:8]
20 print("*"*20,"print(arr_slice))","*"*20)
21 print(arr_slice)
22
23 arr_slice[1]=12345
24 print("*"*20,"print(arr)","*"*20)
25 print(arr)
26
27 #切片[ : ]会给数组中的所有值赋值
28 arr_slice[:]=64
29 print("*"*20,"print(arr)","*"*20)
30 print(arr)
31
32 #**************** 多维数组 *****************
33 print("*"*40,"二维数组","*"*40)
34
35 arr2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
36 print("*"*20,"print(arr2d[2]))","*"*20)
37 print(arr2d[2]) #索引2是个数组
38 print("*"*20,"print(arr2d[0][2])","*"*20)
39 print(arr2d[0][2]) #(1,3)元素
40 print("*"*20,"print(arr2d))","*"*20)
41 print(arr2d)
42 print("*"*20,"print(arr2d[:2])","*"*20)
43 print(arr2d[:2]) #索引0，1的数组
44 print("*"*20,"print( arr2d[:2, 1:])","*"*20)
45 print( arr2d[:2, 1:]) #切行和列
46 print("*"*20,"print( arr2d[1, :2])","*"*20)
47 print( arr2d[1, :2])#第二行的前两列
48 print("*"*20,"print(arr2d[:2, 2])","*"*20)
49 print(arr2d[:2, 2])#第三列的前两行
50 print("*"*20,"print(arr2d[:, :1])","*"*20)
51 print(arr2d[:, :1]) #每个数组的第一个元素
52 arr2d[:2, 1:] = 0
53 print("*"*20,"print(arr2d)","*"*20)
54 print(arr2d)
55
56 arr3d = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
```

```
print("*"*20,"print(arr3d)","*"*20)
print(arr3d)
print("*"*20,"print(arr3d[0])","*"*20)
print(arr3d[0]) #索引0是一个二维数组


old_values = arr3d[0].copy()
arr3d[0] = 42
print("*"*20,"print(old_values)","*"*20)
print(old_values) #保存原数组
print("*"*20,"print(arr3d[0])","*"*20)
print(arr3d[0]) #修改索引0的二维数组的值
arr3d[0]=old_values
print("*"*20,"print(arr3d)","*"*20)
print(arr3d) #把源数据还原

#**********  布尔型索引  **************************
print("*"*40,"布尔型索引","*"*40)
names = np.array(['Bob', 'Joe', 'Will', 'Bob', 'Will', 'Joe', 'Joe'])
print("*"*20,"print(names)","*"*20)
print(names)
print("*"*20,"print(names=='Bob')","*"*20)
print(names=='Bob') #比较运算将会产生一个布尔型数组

data = np.random.randn(7, 4)
print("*"*20,"data = np.random.randn(7, 4)","*"*20)
print(data)

print("*"*20,"print(data[names == 'Bob'])","*"*20)
print(data[names == 'Bob'])#布尔型数组可用于数组索引 -- 布尔型数组的长度必须跟被索引的轴长度一致
print("*"*20,"print(data[names == 'Bob', 2:])","*"*20)
print(data[names == 'Bob', 2:])

#要选择除"Bob"以外的其他值，既可以使用不等于符号（!=），也可以通过~对条件进行否定
print("*"*20,"print(names != 'Bob')","*"*20)
print(names != 'Bob')

print("*"*20,"print(data[~(names == 'Bob')])","*"*20)
print(data[~(names == 'Bob')])
```

```
96
97  #~操作符用来反转条件
98  cond = names == 'Bob'
99  print("*"*20,"cond = names == 'Bob'","*"*20)
100 print(data[~cond])
101 #选取这三个名字中的两个需要组合应用多个布尔条件
102 mask = (names == 'Bob') | (names == 'Will')
103 print("*"*20,"print(mask)","*"*20)
104 print(mask)
105 print("*"*20,"print(data[mask])","*"*20)
106 print(data[mask])
107
108
109 #将data中的所有负值都设置为0
110 data[data < 0] = 0
111 print("*"*20,"data[data < 0] = 0","*"*20)
112 print(data)
113 data[names != 'Joe'] = 7
114 print("*"*20,"data[names != 'Joe'] = 7","*"*20)
115 print(data)
116
117
118 #******************** 花式索引 **********************
119 print("*"*40,"花式索引","*"*40)
120 arr = np.empty((8, 4))
121 print("*"*20,"arr = np.empty((8, 4))","*"*20)
122 for i in range(8):
123     arr[i] = i
124 print(arr)
125
126 print("*"*20,"print(arr[[4, 3, 0, 6]])","*"*20)
127 print(arr[[4, 3, 0, 6]])
128
129 print("*"*20,"print(arr[[-3, -5, -7]])","*"*20)
130 print(arr[[-3, -5, -7]])
131
132 arr = np.arange(32).reshape((8, 4))
133 print("*"*20,"arr = np.arange(32).reshape((8, 4))","*"*20)
134 print(arr)
135
```

```
print("*"*20,"print(arr[[1, 5, 7, 2], [0, 3, 1, 2]])","*"*20)
print(arr[[1, 5, 7, 2], [0, 3, 1, 2]])

print("*"*20,"print(arr[[1, 5, 7, 2]][:, [0, 3, 1, 2]])","*"*20)
print(arr[[1, 5, 7, 2]][:, [0, 3, 1, 2]])
```

```
******************** print(arr) ********************
[0 1 2 3 4 5 6 7 8 9]
******************** print(arr[5]) ********************
5
******************** print(arr[5:8]) ********************
[5 6 7]
******************** print(arr) ********************
[ 0  1  2  3  4 12 12 12  8  9]
******************** print(arr_slice)) ********************
[12 12 12]
******************** print(arr) ********************
[    0     1     2     3     4    12 12345    12     8     9]
******************** print(arr) ********************
[ 0  1  2  3  4 64 64 64  8  9]
****************************************** 二维数组 ******************************************
******************** print(arr2d[2])) ********************
[7 8 9]
******************** print(arr2d[0][2]) ********************
3
******************** print(arr2d)) ********************
[[1 2 3]
 [4 5 6]
 [7 8 9]]
******************** print(arr2d[:2]) ********************
[[1 2 3]
 [4 5 6]]
******************** print( arr2d[:2, 1:]) ********************
[[2 3]
 [5 6]]
******************** print( arr2d[1, :2]) ********************
[4 5]
******************** print(arr2d[:2, 2]) ********************
[3 6]
******************** print(arr2d[:, :1]) ********************
[[1]
 [4]
 [7]]
******************** print(arr2d) ********************
[[1 0 0]
 [4 0 0]
 [7 8 9]]
```

```
****************** print(arr3d) ******************
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
****************** print(arr3d[0]) ******************
[[1 2 3]
 [4 5 6]]
****************** print(old_values) ******************
[[1 2 3]
 [4 5 6]]
****************** print(arr3d[0]) ******************
[[42 42 42]
 [42 42 42]]
****************** print(arr3d) ******************
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
************************************** 布尔型索引 **********************************
****************** print(names) ******************
['Bob' 'Joe' 'Will' 'Bob' 'Will' 'Joe' 'Joe']
****************** print(names=='Bob') ******************
[ True False False  True False False False]
****************** data = np.random.randn(7, 4) ******************
[[ 0.56176335 -2.27953263  2.6739339   0.60535076]
 [-1.80742387  0.01469392  0.55585296 -2.3426252 ]
 [ 0.03846213 -0.47067325  0.69516361 -0.94925829]
 [-2.35810361  2.32159685  1.01093924  0.19586357]
 [-1.74091188  2.03249641 -0.9936282   0.12584777]
 [-0.533427    1.13533162  1.17162903 -0.36516024]
 [ 0.71145703 -2.04924358  0.56044276  0.24269543]]
****************** print(data[names == 'Bob']) ******************
[[ 0.56176335 -2.27953263  2.6739339   0.60535076]
 [-2.35810361  2.32159685  1.01093924  0.19586357]]
****************** print(data[names == 'Bob', 2:]) ******************
[[2.6739339  0.60535076]
 [1.01093924 0.19586357]]
****************** print(names != 'Bob') ******************
[False  True  True False  True  True  True]
****************** print(data[~(names == 'Bob')]) ******************
[[-1.80742387  0.01469392  0.55585296 -2.3426252 ]
 [ 0.03846213 -0.47067325  0.69516361 -0.94925829]
 [-1.74091188  2.03249641 -0.9936282   0.12584777]
 [-0.533427    1.13533162  1.17162903 -0.36516024]
 [ 0.71145703 -2.04924358  0.56044276  0.24269543]]
****************** cond = names == 'Bob' ******************
```

```
[[-1.80742387  0.01469392  0.55585296 -2.3426252 ]
 [ 0.03846213 -0.47067325  0.69516361 -0.94925829]
 [-1.74091188  2.03249641 -0.9936282   0.12584777]
 [-0.533427    1.13533162  1.17162903 -0.36516024]
 [ 0.71145703 -2.04924358  0.56044276  0.24269543]]
******************** print(mask) ********************
[ True False  True  True  True False False]
******************** print(data[mask]) ********************
[[ 0.56176335 -2.27953263  2.6739339   0.60535076]
 [ 0.03846213 -0.47067325  0.69516361 -0.94925829]
 [-2.35810361  2.32159685  1.01093924  0.19586357]
 [-1.74091188  2.03249641 -0.9936282   0.12584777]]
******************** data[data < 0] = 0 ********************
[[0.56176335 0.         2.6739339  0.60535076]
 [0.         0.01469392 0.55585296 0.        ]
 [0.03846213 0.         0.69516361 0.        ]
 [0.         2.32159685 1.01093924 0.19586357]
 [0.         2.03249641 0.         0.12584777]
 [0.         1.13533162 1.17162903 0.        ]
 [0.71145703 0.         0.56044276 0.24269543]]
******************** data[names != 'Joe'] = 7 ********************
[[7.         7.         7.         7.        ]
 [0.         0.01469392 0.55585296 0.        ]
 [7.         7.         7.         7.        ]
 [7.         7.         7.         7.        ]
 [7.         7.         7.         7.        ]
 [0.         1.13533162 1.17162903 0.        ]
 [0.71145703 0.         0.56044276 0.24269543]]
****************************************** 花式索引 ******************************************
******************** arr = np.empty((8, 4)) ********************
[[0. 0. 0. 0.]
 [1. 1. 1. 1.]
 [2. 2. 2. 2.]
 [3. 3. 3. 3.]
 [4. 4. 4. 4.]
 [5. 5. 5. 5.]
 [6. 6. 6. 6.]
 [7. 7. 7. 7.]]
******************** print(arr[[4, 3, 0, 6]]) ********************
[[4. 4. 4. 4.]
 [3. 3. 3. 3.]
 [0. 0. 0. 0.]
 [6. 6. 6. 6.]]
******************** print(arr[[-3, -5, -7]]) ********************
[[5. 5. 5. 5.]
 [3. 3. 3. 3.]
 [1. 1. 1. 1.]]
******************** arr = np.arange(32).reshape((8, 4)) ********************
[[ 0  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]]
******************** print(arr[[1, 5, 7, 2], [0, 3, 1, 2]]) ********************
[ 4 23 29 10]
******************** print(arr[[1, 5, 7, 2]][:, [0, 3, 1, 2]]) ********************
[[ 4  7  5  6]
 [20 23 21 22]
 [28 31 29 30]
 [ 8 11  9 10]]

进程已结束,退出代码0
```