```
numpy.array(object, dtype = None, copy = True, order = None, subok = False, ndmin = 0)
'''
    object: 数组或嵌套的数列
    dtype: 数据类型
    copy: 对象是否需要复制(new的内存空间)
    order: 创建数组的样式，A为任意方向，C为行方向，F为列方向
    subok: 返回一个与基类型一致的数组
    ndmin: 指定生成数组的最小维度
'''
```

1.生成最小维度

```
# 最小维度
import numpy as np
a = np.array([1, 2, 3, 4, 5], ndmin =  2)
print (a)

# [[1 2 3 4 5]]
```

Numpy的数组维度数量称为秩(rank)，每一个线性的数组是一个轴(axis)，也就是维度

```
data=[
    [A,B,C],
    [D,E,F],
    [G,H,I] ]
data.max(axis=0)
data.max(axis=1)
```

来看看对`axis=0`的操作，此时我们做`max(axis=0)`，理应从列出发，得到如下结果：

$$[max(A,D,G),max(B,E,H),max(C,F,I)]$$

那`axis=1`的操作，应该就是：

$$[max(A,B,C),max(D,E,F),max(G,H,I)]$$

2.常用属性

```
import numpy as np

a=np.arange(24)
a.ndim # 1

a=a.reshape(2,3,4)
a.ndim # 3

a.shape # (2,3,4)


a.dtype # dtype('int32')
a.dtype=np.float64
a.dtype # dtype('float64')

numpy.empty(shape, dtype = float, order = 'C')

'''
    shape: 数组形状
    dtype: 数据类型
    order: 行优先还是列优先，表示在计算机内中存储元素的顺序
'''



x=np.empty([3,2],dtype=int)

numpy.zeros(shape, dtype = float, order = 'C')
# 默认浮点
x=np.zeros(5)
print(x) # [0. 0. 0. 0. 0.]

# 设置类型为整数
y=np.zeros((5,),dtype=int)
print(y) # [0 0 0 0 0]
z=np.zeros((2,2),dtype=int)
print(z)
'''[[0 0]
 [0 0]]'''


numpy.ones(shape,dtype,order)
numpy.arange(start=0,stop,step,dtype)#返回一个0~n-1的ndarray数组
numpy.eye(n)#生成单位矩阵
numpy.diag(list)#生成对角线矩阵


np.linspace(start, stop, num=50, endpoint=True, retstep=False,
dtype=None)
'''
    endpoint: 是否包含终止点
```

```
    retstep: 是否输出间距
'a=np.linspace(1,10,10)
print(a) # [ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
''
```

3.切片

```
A[star:step:end]#注意左闭右开

x=np.arange(1,7).reshape(3,2)
print(x)
y=x[[0,1,2,1],[0,1,0,0]]
print(y)

"""[[1 2]
 [3 4]
 [5 6]]

[1 4 5 3]
"""


x=np.array([np.nan,1,2,np.nan,3,4,5])
print(x[x>5]) # 找出大于5的元素
print(x[~np.isnan(x)])
#[1. 2. 3. 4. 5.]
```

4.广播

```
#在两个数组形状不同时，numpy将自动触发广播机制。
a=np.array([
    [0,0,0],
    [10,10,10],
    [20,20,20],
    [30,30,30]
])
b=np.array([0,1,2])
print(a+b)

'''[[ 0  1  2]
 [10 11 12]
 [20 21 22]
 [30 31 32]]

'''
```

5.逻辑运算

```
a=np.random.randint(0,20,10)
print(a[a>5])#[ 7 11  8 13  7 16  7 19]

a=np.random.randint(0,20,10)
print(a[(a>5)|(a*2<10)])#[10 18  8  1  4 16 13 18  1]


np.any()#只要有一个元素满足条件就返回True
np.any(stock_day_rise[0:2,0:5] > 0)
```

6.其他

```
#numpy.transpose 函数用于对换数组的维度，格式如下：
numpy.transpose(arr, axes)


#numpy.split 函数沿特定的轴将数组分割为子数组，格式如下：
numpy.split(ary, indices_or_sections, axis)
#ary：被分割的数组
#indices_or_sections：如果是一个整数，就用该数平均切分，如果是一个数组，
#为沿轴切分的位置（左开右闭）
#axis：设置沿着哪个方向进行切分，默认为 0，横向切分，即水平方向。
#为 1 时，纵向切分，即竖直方向。
import numpy as np

a = np.arange(9)

print ('第一个数组：')
print (a)
print ('\n')

print ('将数组分为三个大小相等的子数组：')
b = np.split(a,3)
print (b)
print ('\n')

print ('将数组在一维数组中表明的位置分割：')
b = np.split(a,[4,7])
print (b)

'''
第一个数组：
[0 1 2 3 4 5 6 7 8]


将数组分为三个大小相等的子数组：
[array([0, 1, 2]), array([3, 4, 5]), array([6, 7, 8])]


将数组在一维数组中表明的位置分割：
[array([0, 1, 2, 3]), array([4, 5, 6]), array([7, 8])]
'''
```

```
#append 函数返回的始终是一个一维数组。
numpy.append(arr, values, axis=None)
```

- arr：输入数组
- values：要向arr添加的值，需要和arr形状相同（除了要添加的轴）

- axis：默认为 None。当axis无定义时，是横向加成，返回总是为一维数组！当axis有定义的时候，分别为0和1的时候。当axis有定义的时候，分别为0和1的时候（列数要相同）。当axis为1时，数组是加在右边（行数要相同）。

```
import numpy as np

a = np.array([[1,2,3],[4,5,6]])

print ('第一个数组：')
print (a)
print ('\n')

print ('向数组添加元素：')
print (np.append(a, [7,8,9]))
print ('\n')

print ('沿轴 0 添加元素：')
print (np.append(a, [[7,8,9]],axis = 0))
print ('\n')

print ('沿轴 1 添加元素：')
print (np.append(a, [[5,5,5],[7,8,9]],axis = 1))


'''
第一个数组：
[[1 2 3]
 [4 5 6]]


向数组添加元素：
[1 2 3 4 5 6 7 8 9]


沿轴 0 添加元素：
[[1 2 3]
 [4 5 6]
 [7 8 9]]


沿轴 1 添加元素：
[[1 2 3 5 5 5]
 [4 5 6 7 8 9]]
'''
```

```
numpy.around() # 函数返回指定数字的四舍五入值。
numpy.floor() # 返回小于或者等于指定表达式的最大整数，即向下取整。
numpy.ceil() # 返回大于或者等于指定表达式的最小整数，即向上取整。
```

NumPy 算术函数包含简单的加减乘除: add()，subtract()，multiply() 和 divide()。

需要注意的是数组必须具有相同的形状或符合数组广播规则。

```python
import numpy as np

a = np.arange(9, dtype = np.float_).reshape(3,3)
print ('第一个数组：')
print (a)
print ('\n')
print ('第二个数组：')
b = np.array([10,10,10])
print (b)
print ('\n')
print ('两个数组相加：')
print (np.add(a,b))
print ('\n')
print ('两个数组相减：')
print (np.subtract(a,b))
print ('\n')
print ('两个数组相乘：')
print (np.multiply(a,b))
print ('\n')
print ('两个数组相除：')
print (np.divide(a,b))

'''
第一个数组：
[[0. 1. 2.]
 [3. 4. 5.]
 [6. 7. 8.]]


第二个数组：
[10 10 10]


两个数组相加：
[[10. 11. 12.]
 [13. 14. 15.]
 [16. 17. 18.]]


两个数组相减：
[[-10.  -9.  -8.]
 [ -7.  -6.  -5.]
 [ -4.  -3.  -2.]]


两个数组相乘：
[[ 0. 10. 20.]
 [30. 40. 50.]
 [60. 70. 80.]]
```

```
两个数组相除：
[[0.  0.1 0.2]
 [0.3 0.4 0.5]
 [0.6 0.7 0.8]]
'''
```

numpy.mean() 函数返回数组中元素的算术平均值。 如果提供了轴，则沿其计算。

算术平均值是沿轴的元素的总和除以元素的数量。

```
import numpy as np

a = np.array([[1,2,3],[3,4,5],[4,5,6]])
print ('我们的数组是：')
print (a)
print ('\n')
print ('调用 mean() 函数：')
print (np.mean(a))
print ('\n')
print ('沿轴 0 调用 mean() 函数：')
print (np.mean(a, axis =  0))
print ('\n')
print ('沿轴 1 调用 mean() 函数：')
print (np.mean(a, axis =  1))

'''
我们的数组是：
[[1 2 3]
 [3 4 5]
 [4 5 6]]


调用 mean() 函数：
3.6666666666666665


沿轴 0 调用 mean() 函数：
[2.66666667 3.66666667 4.66666667]


沿轴 1 调用 mean() 函数：
[2. 4. 5.]
'''
```

```
np.random.rand(dim)
#返回[0,1)区间内均匀分布的数
np.random.uniform(low=0.0,high=1.0,size=None)
#在均匀分布区间[low,high)中随机采样
np.random,randint(low,high,size,dtype=“I”)
#返回随机整数
```

*DataFrame*常用属性

```
import pandas as pd

data = {'姓名': ['张三', '李四', '王二'], '年龄': [23, 27, 26]
, '性别': ['男', '女', '女']}
df = pd.DataFrame(data)

# 返回对象数据形状 即：三行三列
print(df.shape, '\n')
# 返回序列的值
print(df.values, '\n')
# 返回行索引
print(df.index, '\n')
# 返回列索引
print(df.columns, '\n')
# 返回列标签(只针对dataframe数据结构)
print(df.columns.tolist())
# 返回元素数据类型
print(df.dtypes, '\n')
# 返回对象的维度
print(df.ndim, '\n')
# 返回对象的个数
print(df.size, '\n')
```

```
#df.iloc 和 df.loc 的区别

#df.iloc[row_index位置, column_index位置]
#df.loc[row_index, column_index]
```

```
df.loc[df['gender']=='m','name'] # 选取gender列是m，name列的数据
df.loc[df['gender']=='M',['name','age']] #选取gender列是m，name和age列的数据
```

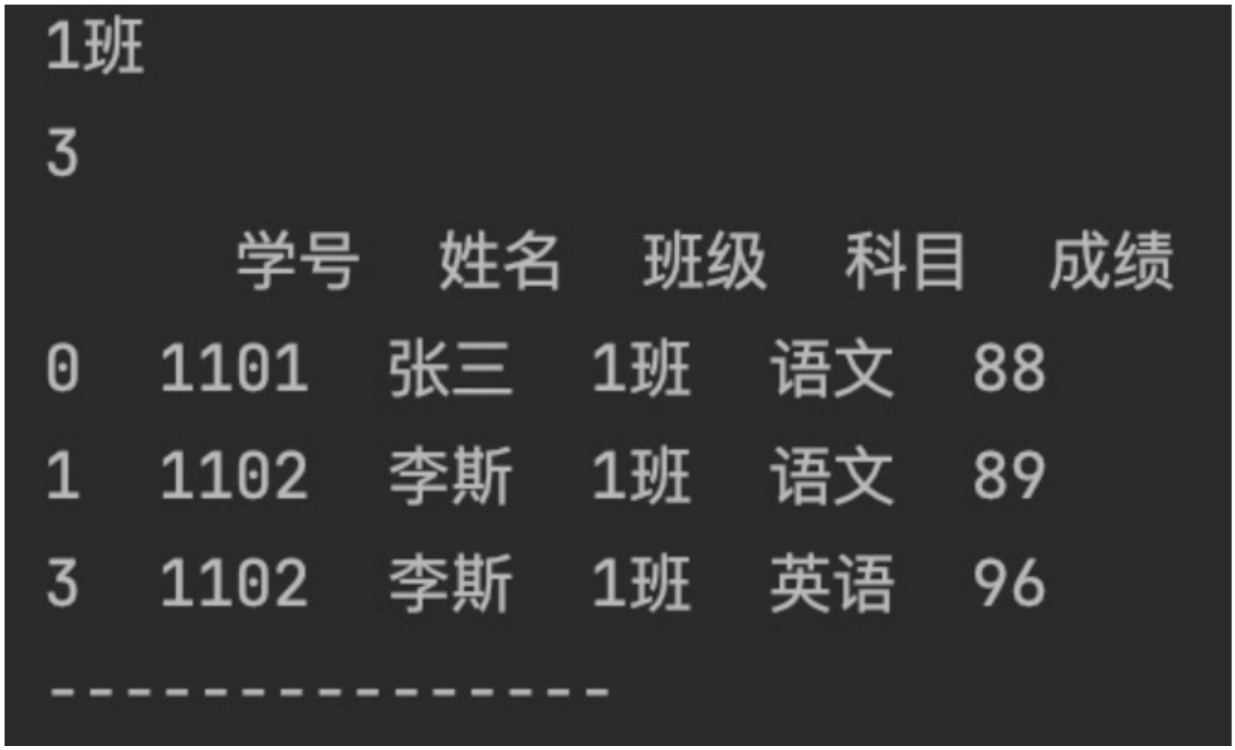
```
import pandas as pd
import numpy as np

arr = np.array([['赵一', 23, '男'], ['钱二', 27, '女'], ['孙三', 26, '女'], ['李四', 12,
'男']])
df1 = pd.DataFrame(arr, columns=['姓名', '年龄', '性别'], index=['a', 'b', 'c', 'd'])

# 读取‘姓名’列
for i in df1.index:
    print(df1['姓名'][i])

# 读取第1行
for j in df1.columns:
    print(df1[j][0])

# 读取每个元素
for i in df1.index:
    for j in df1.columns:
        print(df1[j][i])
```

groupby函数功能：对DataFrame进行分组（可单类分组，可多类分组） 需求：按“字段”列对数据data进行分组 groupby函数基本格式：data.groupby(['分组字段'])

data：要分组的原始数据 分组字段：分组参考的数据列名

```
import pandas as pd
data = pd.read_excel('/Users/ABC/Documents/工作簿1.xlsx')
for name, group in data.groupby(['班级']):
    num_g = group['班级'].count() # 获取组内记录数目
    print(name) # name为班级名称
    print(num_g)
    print(group) # group为每个分组中的记录情况
    print('---------------')
```

班级分组结果：



三、groupby分组运算

聚合操作是groupby后非常常见的操作，聚合操作可以用来求和、均值、最大值、最小值等，下面的表格列出了Pandas中常见的聚合操作。

函数	作用
min	最小值
max	最大值
sum	求和
mean	均值
median	中位数
std	标准差
var	方差
count	计数

```
DataFrame.apply(self, func, axis=0, raw=False, result_type=None,
args=(), **kwds)

#func 代表的是传入的函数或 lambda 表达式;
#axis 参数可提供的有两个，该参数默认为0/列
#0 或者 index ，表示函数处理的是每一列;
#1 或 columns ，表示处理的是每一行;
```