# 实验1：矩阵库numpy的使用（一）

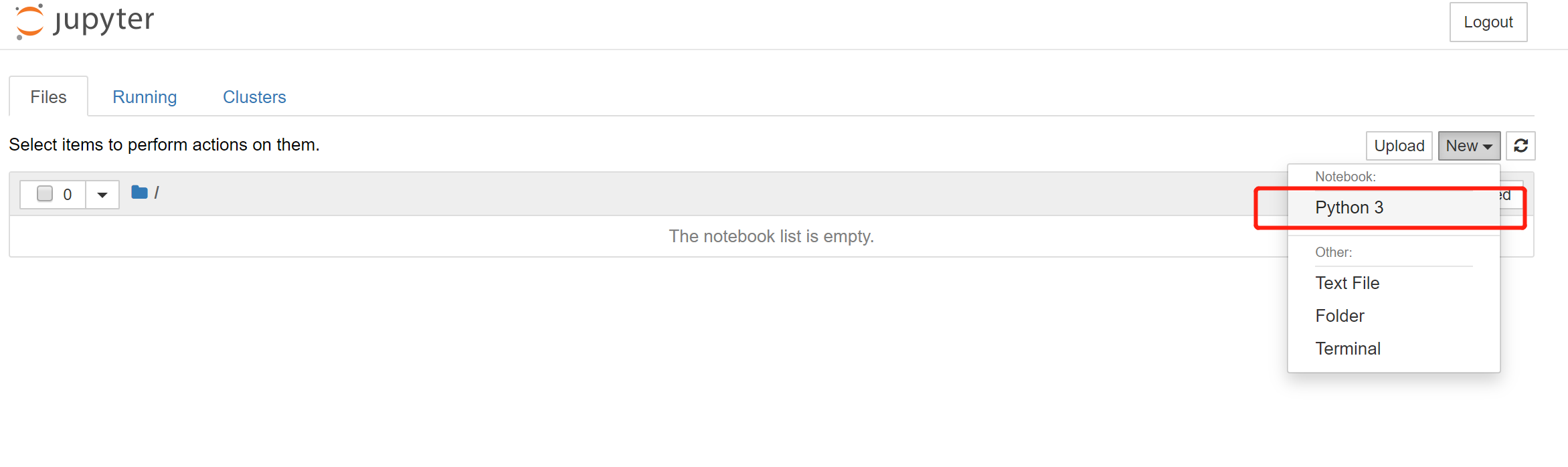
**什么是numpy？**

NumPy是一个功能强大的Python库，主要用于对多维数组执行计算。NumPy这个词来源于两个单词-- Numerical和Python。NumPy提供了大量的库函数和操作，可以帮助程序员轻松地进行数值计算。这类数值计算广泛用于以下任务：

机器学习模型：在编写机器学习算法时，需要对矩阵进行各种数值计算。例如矩阵乘法、换位、加法等。NumPy提供了一个非常好的库，用于简单(在编写代码方面)和快速(在速度方面)计算。NumPy数组用于存储训练数据和机器学习模型的参数。

图像处理和计算机图形学：计算机中的图像表示为多维数字数组。NumPy成为同样情况下最自然的选择。实际上，NumPy提供了一些优秀的库函数来快速处理图像。例如，镜像图像、按特定角度旋转图像等。

**1.1 新建python3的notebook**

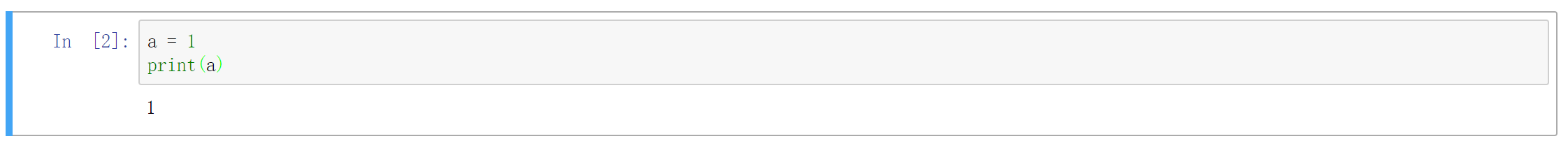


输入：

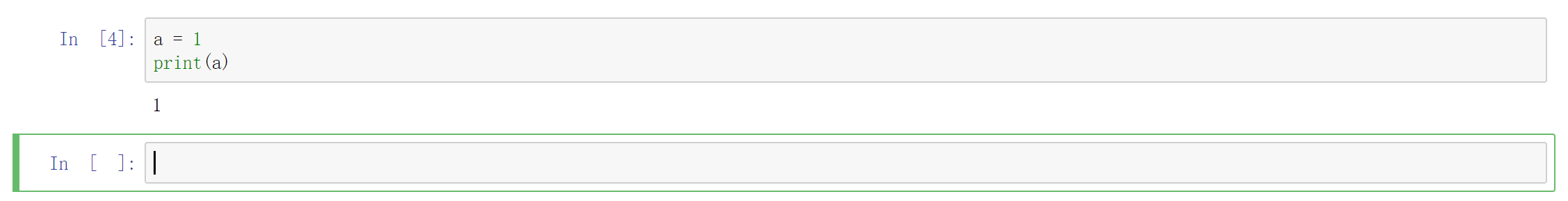
*a = 1*

*print(a)*

然后按键Ctrl+Enter进行测试（本行模式）



或者按键Shift+Enter进行测试（进入到下一个Cell），也可按Run按钮



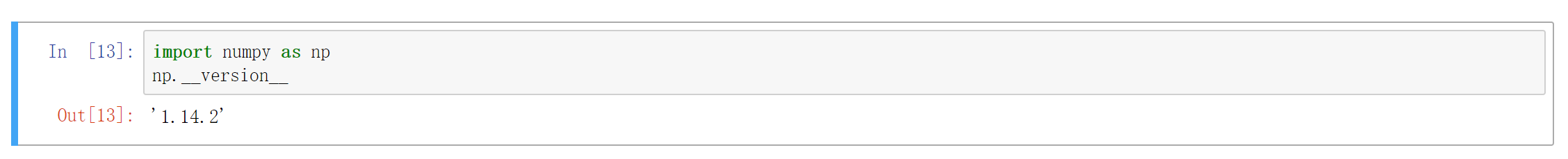
**1.2 引入numpy包**

输入：

*import numpy as np*

打印一个numpy的版本

*np.\_\_version\_\_*



## 2、NumPy Ndarray 对象

NumPy 最重要的一个特点是其 N 维数组对象 ndarray，它是一系列同类型数据的集合，以 0 下标为开始进行集合中元素的索引。

**2.1 创建一个[1,2,3]的列向量**

输入：

*import numpy as np*

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

*print (a)*



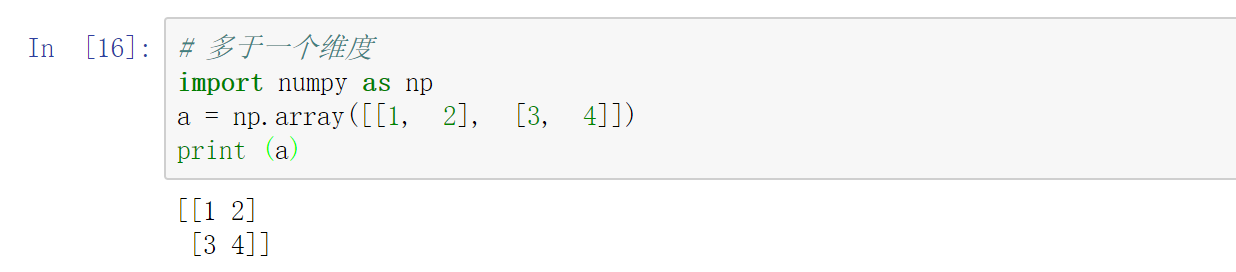
**2.2 创建一个2\*2的矩阵**

# 多于一个维度

*import numpy as np*

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

*print (a)*



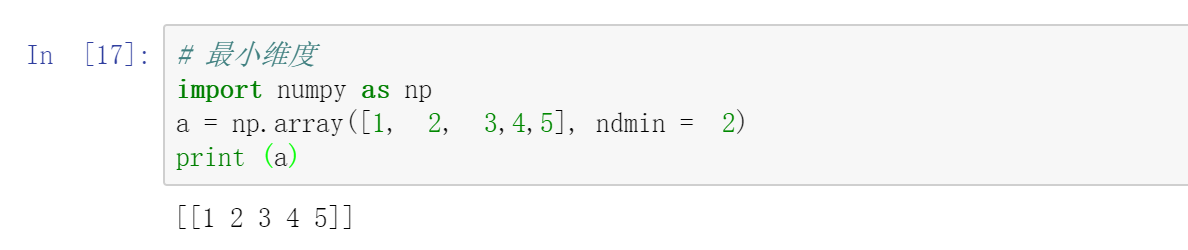
**2.3 创建一个1\*5的矩阵**

*# 最小维度*

*import numpy as np*

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

*print (a)*



**2.4 指定数据类型**

# dtype 参数

*import numpy as np*

*a = np.array([1, 2, 3], dtype = float)*

*print (a)*



## 3、NumPy 数据类型

numpy 支持的数据类型比 Python 内置的类型要多很多，基本上可以和 C 语言的数据类型对应上，其中部分类型对应为 Python 内置的类型。下表列举了常用 NumPy 基本类型。



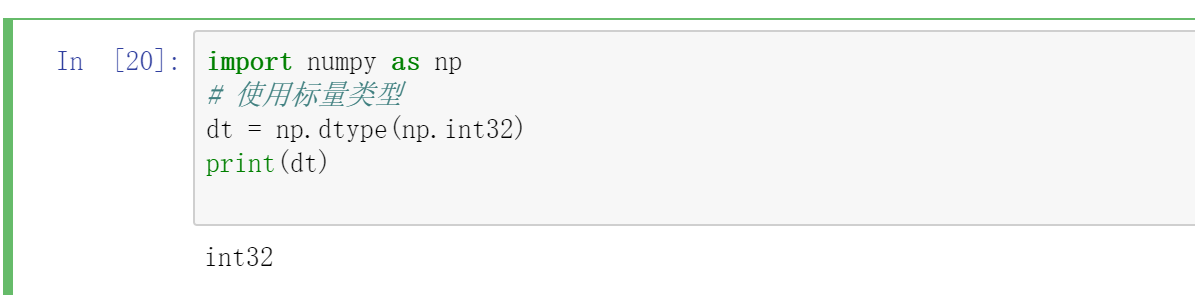
**3.1 打印类型**

*import numpy as np*

*# 使用标量类型*

*dt = np.dtype(np.int32)*

*print(dt)*

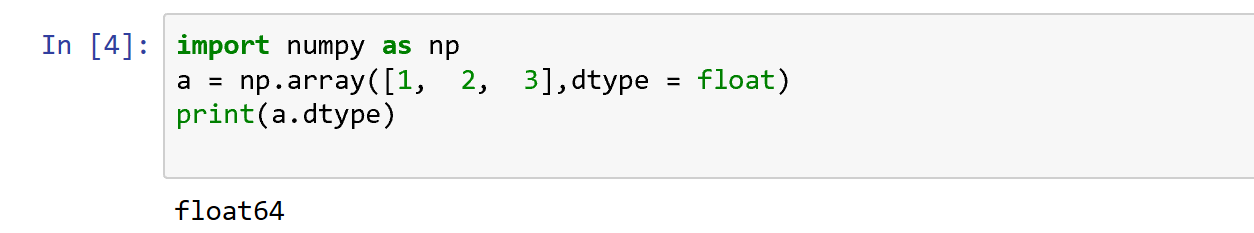


**3.2 打印矩阵数据类型**

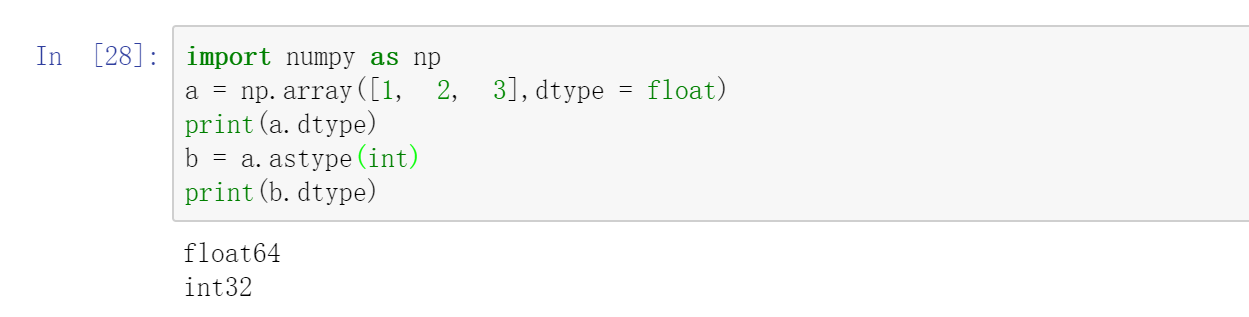
import numpy as np

a = np.array([1, 2, 3],dtype = float)

print(a.dtype)



**3.3 转换矩阵数据类型**



## 4、Numpy数组属性

在 NumPy中，每一个线性的数组称为是一个轴（axis），也就是维度（dimensions）。比如说，二维数组相当于是两个一维数组，其中第一个一维数组中每个元素又是一个一维数组。所以一维数组就是 NumPy 中的轴（axis），第一个轴相当于是底层数组，第二个轴是底层数组里的数组。而轴的数量——秩，就是数组的维数。

很多时候可以声明 axis。axis=0，表示沿着第 0 轴进行操作，即对每一列进行操作；axis=1，表示沿着第1轴进行操作，即对每一行进行操作。

NumPy 的数组中比较重要 ndarray 对象属性有：

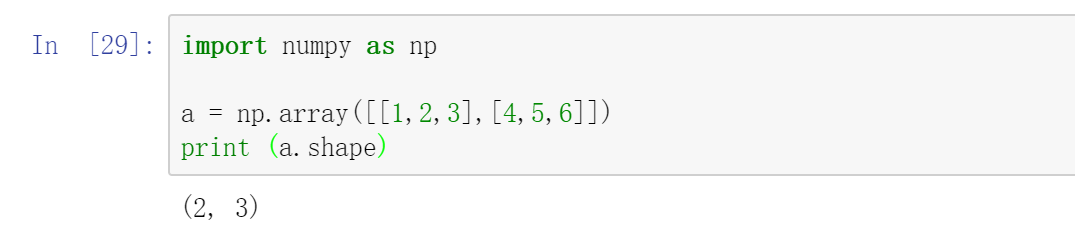


**4.1 矩阵形状**

*import numpy as np*

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

*print (a.shape)*

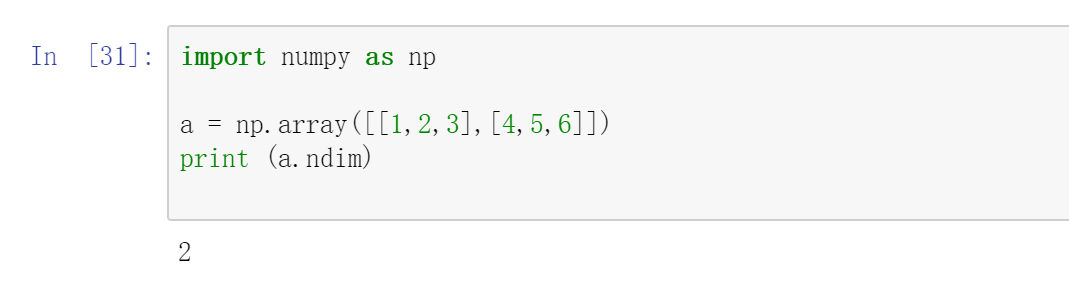


**4.2 矩阵的秩**

*import numpy as np*

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

*print (a.ndim)*



## 5、NumPy 创建数组

ndarray 数组除了可以使用底层 ndarray 构造器来创建外，也可以通过以下几种方式来创建。

**5.1 numpy.zeros**

创建指定大小的数组，数组元素以 0 来填充：

import numpy as np

# 默认为浮点数

*x = np.zeros(5)*

*print(x)*

*# 设置类型为整数*

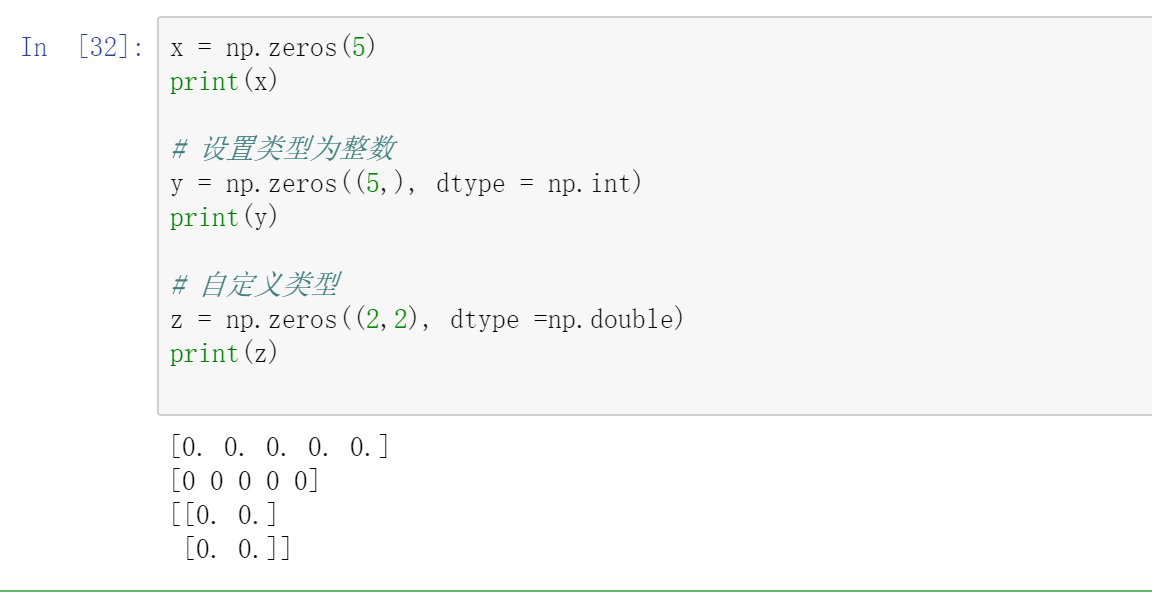
*y = np.zeros((5,), dtype = np.int)*

*print(y)*

*# 自定义类型*

*z = np.zeros((2,2), dtype =np.double)*

*print(z)*



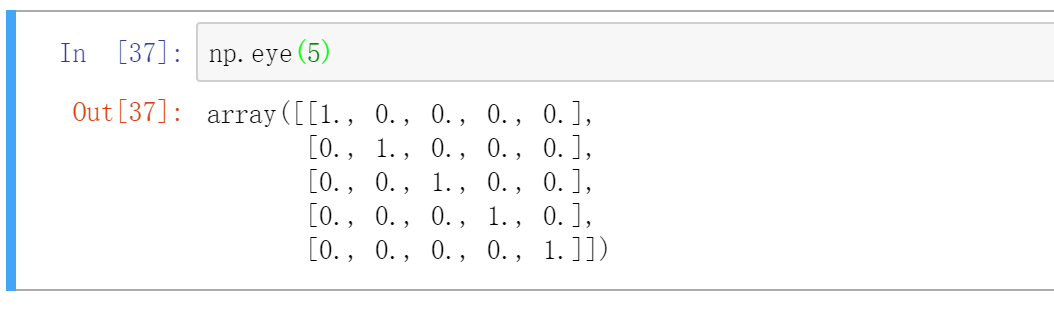
**5.2 numpy.ones**

创建指定形状的数组，数组元素以 1 来填充：

**5.3 numpy.eye**

创建对角矩阵

np.eye(5)

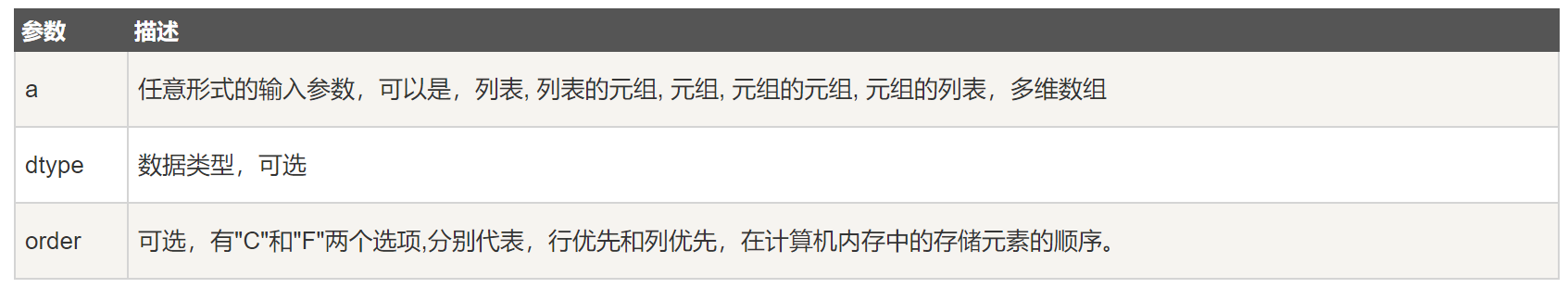


## 6、NumPy 从已有的数组创建数组

**numpy.asarray**

numpy.asarray 类似 numpy.array，但 numpy.asarray 参数只有三个，比 numpy.array 少两个。

numpy.asarray(a, dtype = None, order = None)



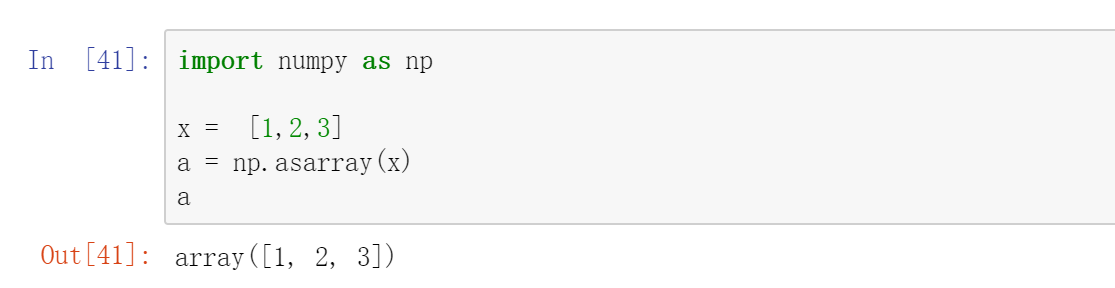
**6.1 list转为向量**

*import numpy as np*

*x = [1,2,3]*

*a = np.asarray(x)*

*print (a)*



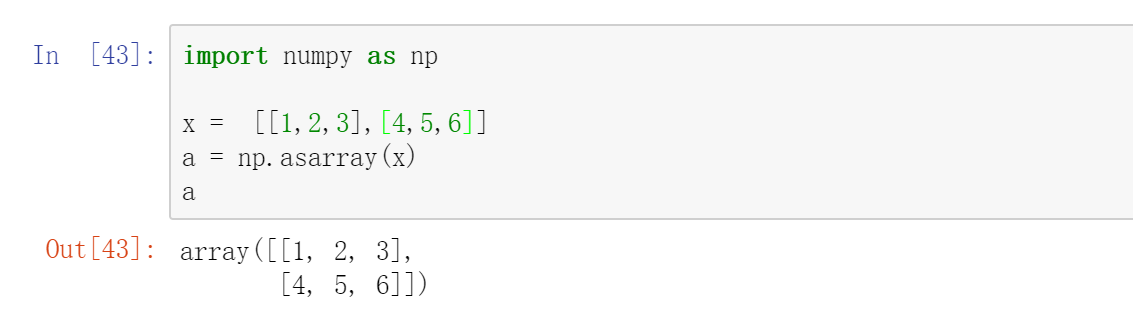
**6.2 list转为矩阵**

*import numpy as np*

*x = [[1,2,3],[4,5,6]]*

*a = np.asarray(x)*

*a*



## 7、NumPy 从数值范围创建数组

**7.1 numpy.arange**

numpy 包中的使用 arange 函数创建数值范围并返回 ndarray 对象，函数格式如下：

numpy.arange(start, stop, step, dtype)

根据 start 与 stop 指定的范围以及 step 设定的步长，生成一个 ndarray。



**创建0-4的向量**

import numpy as np

x = np.arange(5)

print (x)

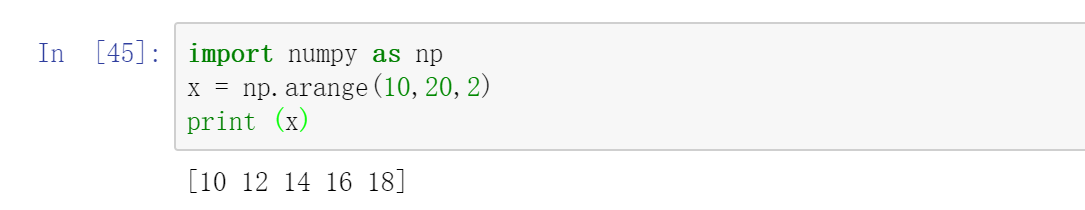


**设置了起始值、终止值及步长**

import numpy as np

x = np.arange(10,20,2)

print (x)



**7.2 numpy.linspace**

numpy.linspace 函数用于创建一个一维数组，数组是一个等差数列构成的，格式如下：

np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)



**设置起始点为 1 ，终止点为 10，数列个数为 10。**

import numpy as np

a = np.linspace(1,10,10)

print(a)



**设置元素全部是1的等差数列：**

*import numpy as np*

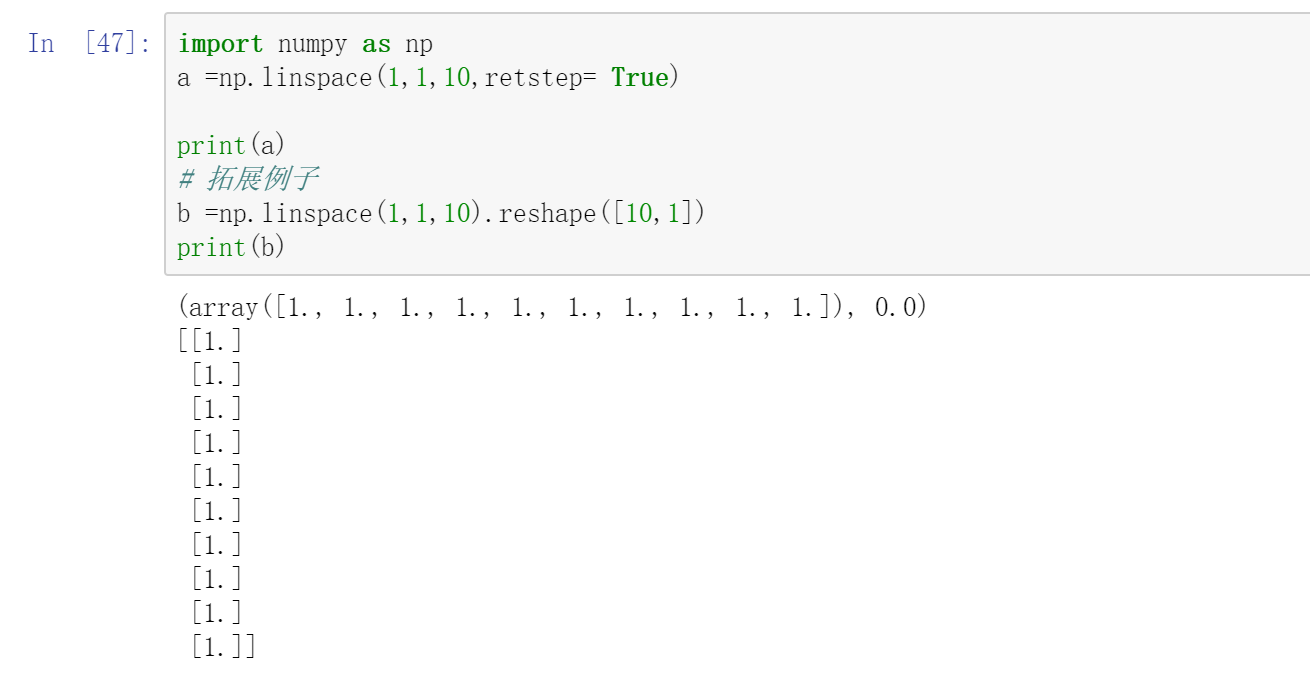
*a =np.linspace(1,10,10,retstep= True)*

*print(a)*

*# 拓展例子*

*b =np.linspace(1,10,10).reshape([10,1])*

*print(b)*



numpy.logspace

numpy.logspace 函数用于创建一个于等比数列。格式如下：

np.logspace(start, stop, num=50, endpoint=True, base=10.0, dtype=None)



## 8、NumPy 索引和切片

ndarray对象的内容可以通过索引或切片来访问和修改，与 Python 中 list 的切片操作一样。ndarray 数组可以基于 0 - n 的下标进行索引，从原数组中切割出一个新数组。

8.1 取出元素

import numpy as np

a = np.arange(10) # [0 1 2 3 4 5 6 7 8 9]

b = a[5]

print(b)

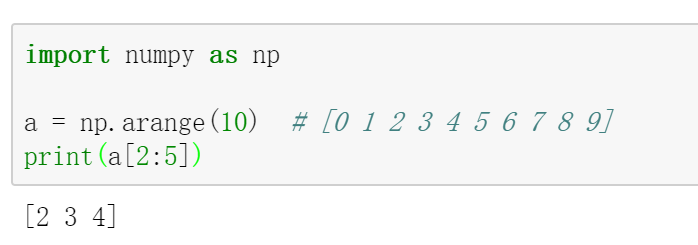


8.2 去除连续的元素

import numpy as np

a = np.arange(10) # [0 1 2 3 4 5 6 7 8 9]

print(a[2:5])



8.3 多维数组提取

import numpy as np

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

print(a)

# 从某个索引处开始切割

print('从数组索引 a[1:] 处开始切割')

print(a[1:])



8.4 获取大于 5 的元素

import numpy as np

x = np.array([[ 0, 1, 2],[ 3, 4, 5],[ 6, 7, 8],[ 9, 10, 11]])

print ('我们的数组是：')

print (x)

print ('\n')

# 现在我们会打印出大于 5 的元素

print ('大于 5 的元素是：')

print (x[x > 5])



## 9、NumPy 广播(Broadcast)

广播(Broadcast)是 numpy 对不同形状(shape)的数组进行数值计算的方式， 对数组的算术运算通常在相应的元素上进行。

如果两个数组 a 和 b 形状相同，即满足 a.shape == b.shape，那么 a\*b 的结果就是 a 与 b 数组对应位相乘。这要求维数相同，且各维度的长度相同。

9.1 对位相乘

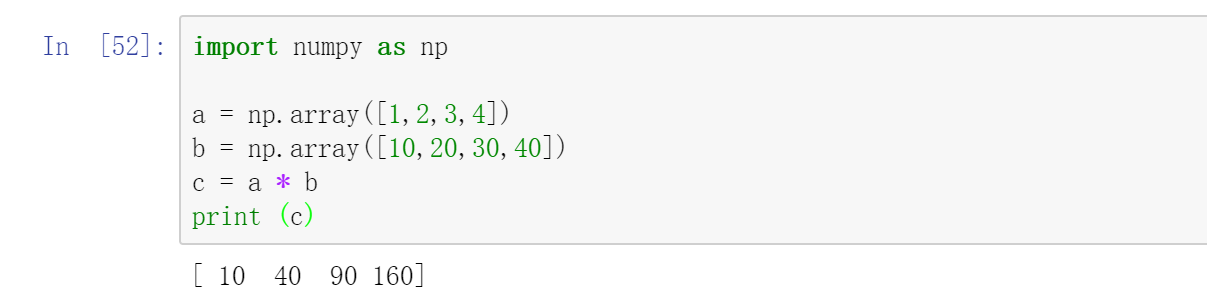
import numpy as np

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

b = np.array([10,20,30,40])

c = a \* b

print (c)



9.2 广播相乘

import numpy as np

a = np.array([[ 0, 0, 0],

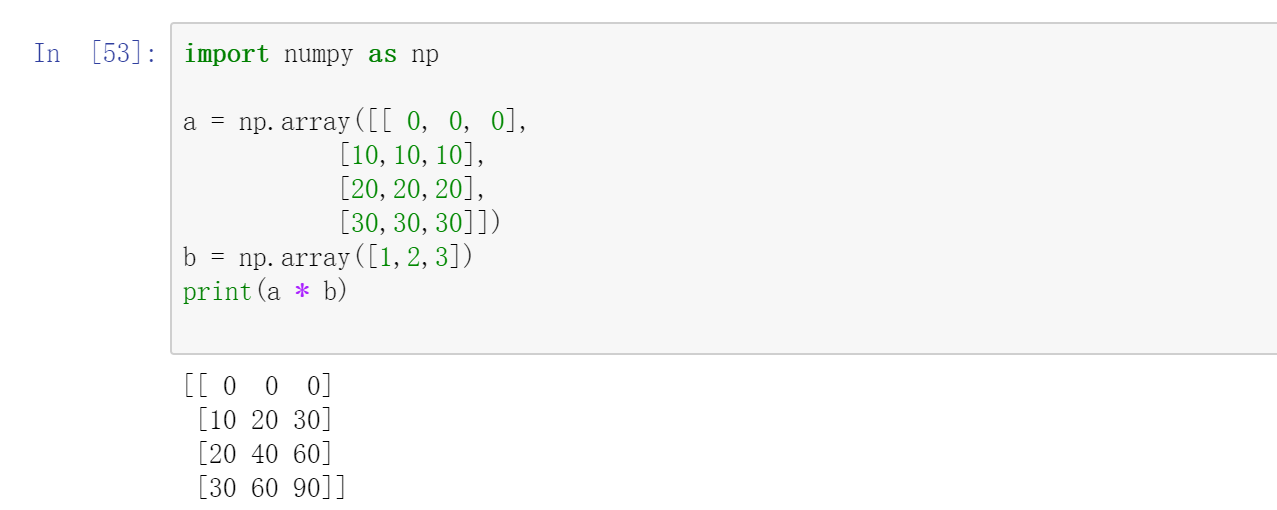
[10,10,10],

[20,20,20],

[30,30,30]])

b = np.array([1,2,3])

print(a \* b)



## 练习：

1、x = np.array([[1,2,3,4],[4,5,6,7]]) 分别取出x的第2-3列；x的第2行；x的第2行第3列的元素。