西安交通大学实验报告

课程 **Python数据处理** 实验名称 **NUMPY应用**  共 **15** 页

系 别 **电信学部** 实 验 日 期 **2023**年**3**月**22**日

专业班级 实 验 报 告 日 期 **2023**年**3**月**22**日

姓 名 学号 报 告 退 发( 订正 、 重做 )

同 组 人\_\_\_无\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 教 师 审 批 签 字

*本次实验合计约使用****2****个小时*

# **实验目的**

1. **熟悉numpy库的基本操作**
2. **掌握numpy数组的创建、索引和切片方法**
3. **学习numpy的常用函数和运算**
4. **实践numpy在数据分析和科学计算中的应用**

# **实验内容**

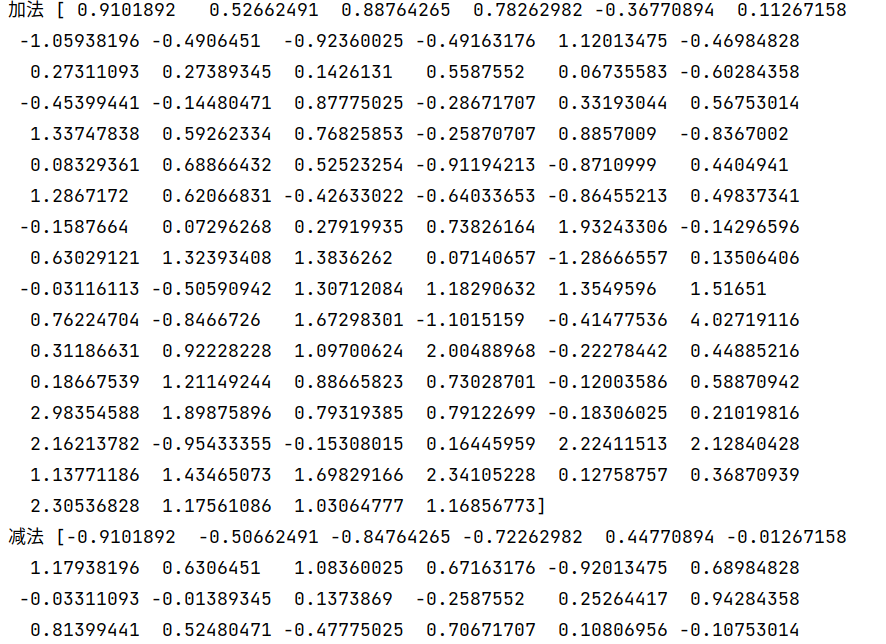
## 实验题目一：

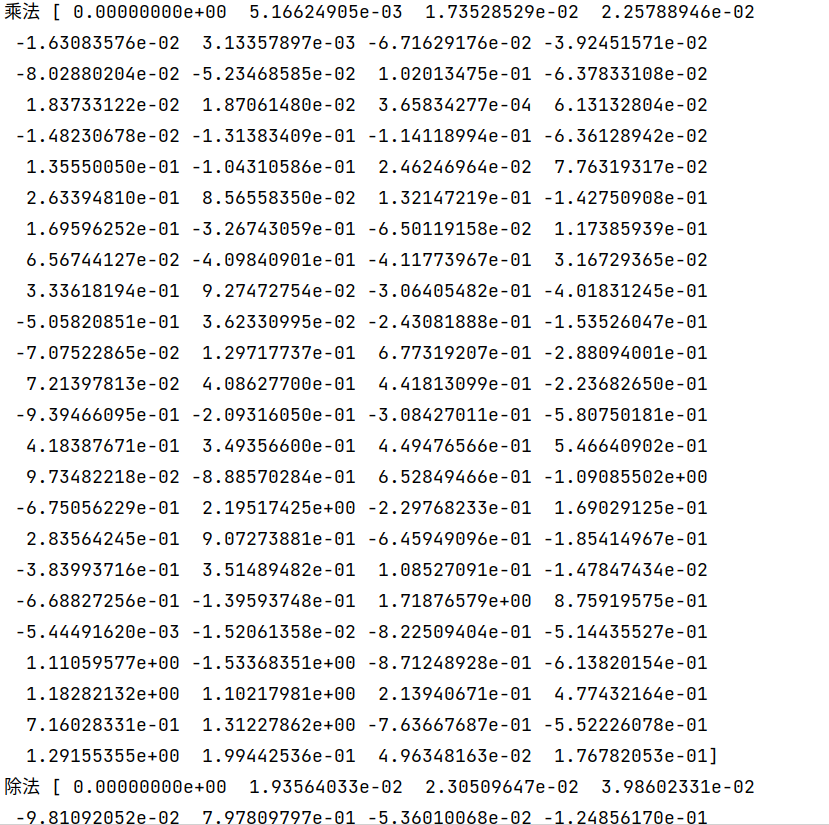
**创建一个数值范围为0~1，间隔为0.01的数组，再创建100个服从正态分布的随机数数组，对上面创建的两个数组进行相加、相减、相乘和相除的四则运算。**

#### 程序源代码：

**import numpy as np  
  
arr1 = np.arange(0, 1.0, 0.01)  
  
arr2 = np.random.normal(size=100)  
  
add = arr1 + arr2  
subtract = arr1 - arr2  
multiply = arr1 \* arr2  
divide = arr1 / arr2  
  
print(f"加法 {add }")  
print(f"减法 {subtract}")  
print(f"乘法 {multiply}")  
print(f"除法 {divide }")**

#### 运行结果：





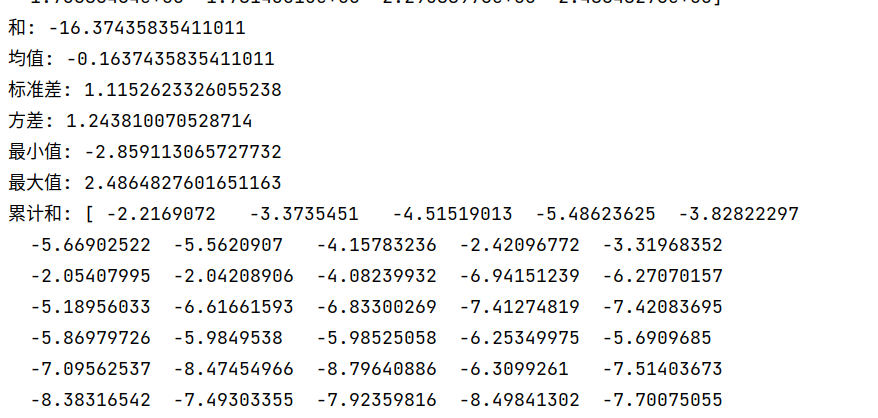
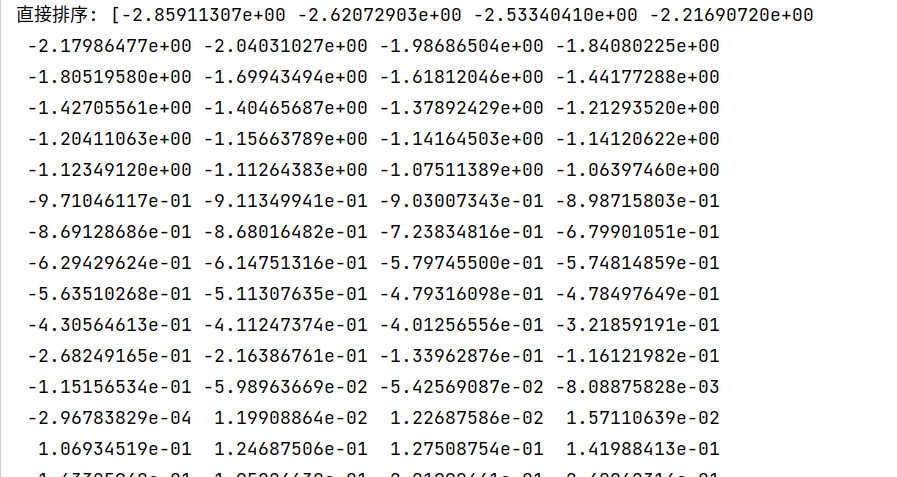
## 实验题目二：

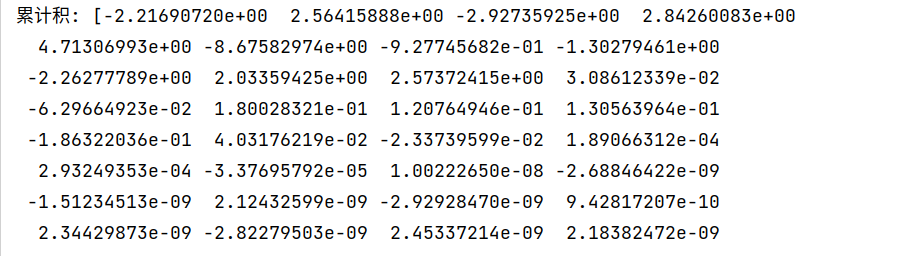
**创建一个数值范围为0~1，间隔为0.01的数组，再创建100个服从正态分布的随机数数组，对上面已经创建的随机数组进行包括直接排序、间接排序、去重、求和、求均值、求标准差、求方差、求最小值、求最大值、求累计和、求累计积等统计分析。**

#### 程序源代码

**import numpy as np  
  
arr1 = np.arange(0, 1.0, 0.01)  
  
arr2 = np.random.normal(size=100)  
  
# 直接排序  
sorted\_arr = np.sort(arr2)  
print(f"直接排序: { sorted\_arr }")  
  
# 间接排序  
argsort\_arr = np.argsort(arr2)  
print(f"间接排序: { argsort\_arr }")  
  
# 去重  
unique\_arr = np.unique(arr2)  
print(f"去重: { unique\_arr }")  
  
# 求和  
sum\_arr = np.sum(arr2)  
print(f"和: { sum\_arr }")  
  
# 求均值  
mean\_arr = np.mean(arr2)  
print(f"均值: { mean\_arr }")  
  
# 求标准差  
std\_arr = np.std(arr2)  
print(f"标准差: { std\_arr }")  
  
# 求方差  
var\_arr = np.var(arr2)  
print(f"方差: { var\_arr }")  
  
# 求最小值  
min\_arr = np.min(arr2)  
print(f"最小值: { min\_arr }")  
  
# 求最大值  
max\_arr = np.max(arr2)  
print(f"最大值: { max\_arr }")  
  
# 求累计和  
cumsum\_arr = np.cumsum(arr2)  
print(f"累计和: { cumsum\_arr }")  
  
# 求累计积  
cumprod\_arr = np.cumprod(arr2)  
print(f"累计积: { cumprod\_arr }")**

#### 运行结果：





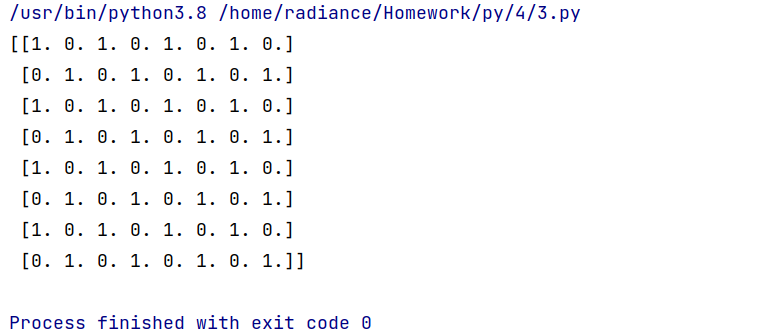
## 实验题目三：

**创建一个8\*8矩阵，矩阵中的每个元素代表国际象棋中对应位置的颜色，颜色一深一浅交错排列在64个位置中，其中1表示黑色，0表示白色，第0行第0列位置的颜色为黑色，请按照上述要求设置矩阵元素的值。**

#### 程序源代码：

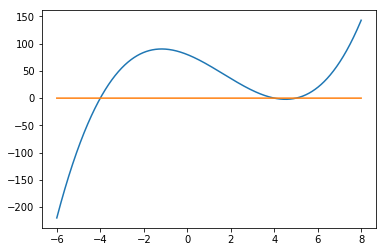
**import numpy as np  
  
chess\_board = np.ones((8, 8))  
  
chess\_board[::2, 1::2] = 0  
chess\_board[1::2, ::2] = 0  
  
print(chess\_board)**

#### 运行结果：



## 实验题目四：

**仿照第2章PPT课件中绘制sin(x)和cos(x)图形的程序，绘制f(x)=x3-5x2-16x+80=0在(-6,8)之间的图形（提示：先创建(-6,8)区间至少1400个浮点数的数组）。同时绘制y=0的水平线，具体图形如下所示：**

****

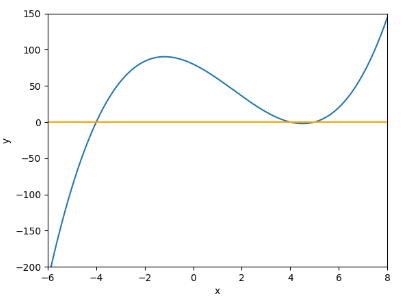
**参考附加章中二分法求方程的根的程序，求方程f(x)在(-6,-2)之间的根。**

#### 程序源代码：

**import numpy as np  
import matplotlib.pyplot as plt  
  
# 创建 (-6,8) 区间至少1400个浮点数的数组  
x = np.linspace(-6, 8, 1400)  
  
y = x \*\* 3 - 5 \* x \*\* 2 - 16 \* x + 80  
  
plt.plot(x, y)  
  
# 绘制水平线  
plt.axhline(y=0, color='orange')  
  
plt.xlim(-6, 8)  
plt.ylim(-100, 100)  
  
plt.xlabel('x')  
plt.ylabel('y')  
  
plt.show()**

**"""  
求方程f(x)在(-6,-2)之间的根  
"""  
def f(x):  
 return x\*\*3 - 5\*x\*\*2 - 16\*x + 80  
  
def solve(a, b, eps):  
 if f(a) \* f(b) >= 0:  
 return None  
 while (b - a) / 2 > eps:  
 c = (a + b) / 2  
 if f(c) == 0:  
 return c  
 elif f(c) \* f(a) < 0:  
 b = c  
 else:  
 a = c  
 return (a + b) / 2  
  
root = solve(-6, -2, 0.0001)  
print("f(x) 在 (-6,-2) 之间的根为:", root)**

#### 运行结果：





## 实验题目五：

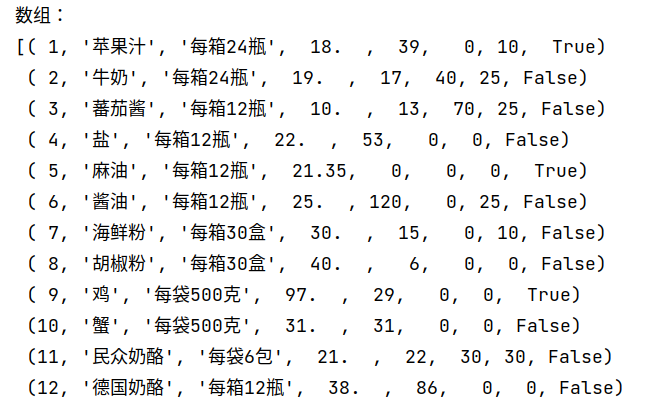
**仿照第2章PPT课件P79页程序例子，将附件中“产品.xlsx”表格数据文件下载到本机，打开产品.xlsx，另存为.csv，逗号分开。采用笔记本打开“产品.csv”，另存为utf-8。采用numpy.genfromtxt()函数导入产品.csv为数组。输出数组。将数组按”单价”排序后，再输出。将数组按单价求和、求均值、求标准差、求方差、求最小值、求最大值、求累计和、求累计积等统计分析。提示：产品表格每项数据类型如下定义**

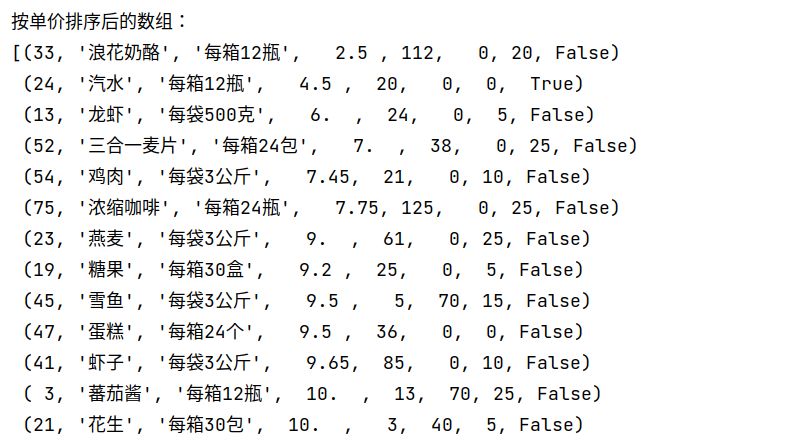
**dt = [('产品ID', 'int32'), ('产品名称', '<U20'), ('单位数量', '<U20'),('单价','float32'),('库存量', 'int32'),('订购量', 'int32'),('再订购量', 'int32'),('中止',bool)]**

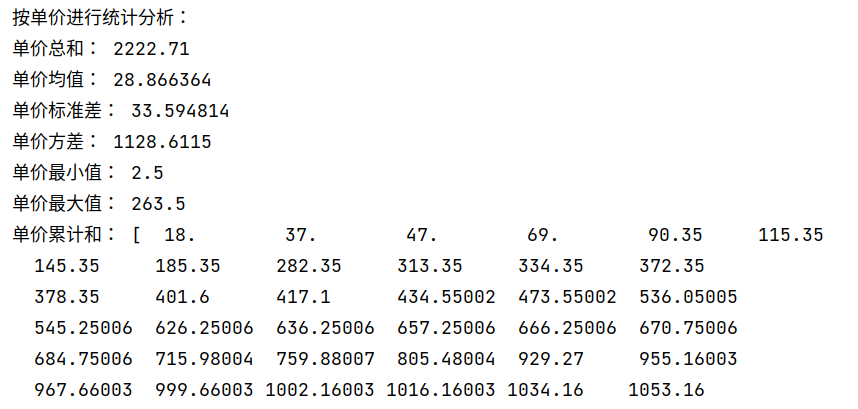
#### 程序源代码：

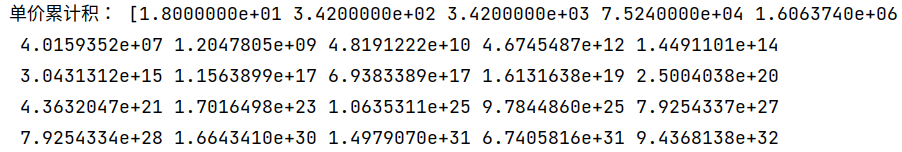
**import numpy as np  
  
arr = np.genfromtxt('产品.csv', delimiter=',', dtype=[(' 产品 ID', 'int32'), (' 产品名称 ','<U20'),  
 (' 单位数量 ', '<U20'),(' 单价 ','float32'),  
 (' 库存量 ', 'int32'),(' 订购量 ', 'int32'),  
 (' 再订购量 ', 'int32'),(' 中止', '?')])  
arr = arr[1:]  
  
print('数组：')  
print(arr)  
  
# 按单价排序  
arr\_sorted = np.sort(arr, order='单价')  
  
# 输出排序后的数组  
print('按单价排序后的数组：')  
print(arr\_sorted)  
  
# 按单价进行统计分析  
print('按单价进行统计分析：')  
print('单价总和：', np.sum(arr['单价']))  
print('单价均值：', np.mean(arr['单价']))  
print('单价标准差：', np.std(arr['单价']))  
print('单价方差：', np.var(arr['单价']))  
print('单价最小值：', np.min(arr['单价']))  
print('单价最大值：', np.max(arr['单价']))  
print('单价累计和：', np.cumsum(arr['单价']))  
print('单价累计积：', np.cumprod(arr['单价']))**

#### 运行结果：









## 实验题目六：

**创建一个存储餐饮企业库存信息的数据类型，用长度为40个字符的字符串表示商品的名称，用一个64位的整数表示商品的库存数量，用一个64位的单精度浮点数表示商品的价格，用一个64位的单精度浮点数表示商品的金额**

**A） 按照前面定义的数据类型创建4个数组，分别是("tomatoes", 42, 4.14，0)，("cabbages", 13, 1.72,0)，("potatoes",30,1.2,0)，("celery",23,2.1,0)**

**B）将4个数组按纵向组合成一个新数组**

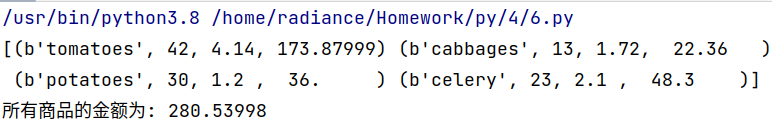
**C）按照金额=数量\*价格，重新计算新数组中每类商品的金额并显示修改后的数组**

**D）显示所有商品的金额**

#### 程序源代码：

**import numpy as np  
  
# 定义数据类型  
dt = np.dtype([('name', 'S40'), ('quantity', np.int64), ('price', np.float32), ('amount', np.float32)])  
  
# 定义4个数组  
arr1 = np.array([("tomatoes", 42, 4.14, 0)], dtype=dt)  
arr2 = np.array([("cabbages", 13, 1.72, 0)], dtype=dt)  
arr3 = np.array([("potatoes", 30, 1.2, 0)], dtype=dt)  
arr4 = np.array([("celery", 23, 2.1, 0)], dtype=dt)  
  
# 将4个数组按纵向组合成一个新数组  
new\_arr = np.concatenate((arr1, arr2, arr3, arr4), axis=0)  
  
# 按照金额=数量\*价格，重新计算新数组中每类商品的金额并显示修改后的数组  
new\_arr['amount'] = new\_arr['quantity'] \* new\_arr['price']  
print(new\_arr)  
  
# 显示所有商品的金额  
print("所有商品的金额为:", np.sum(new\_arr['amount']))**

#### 运行结果：



## 实验题目七：

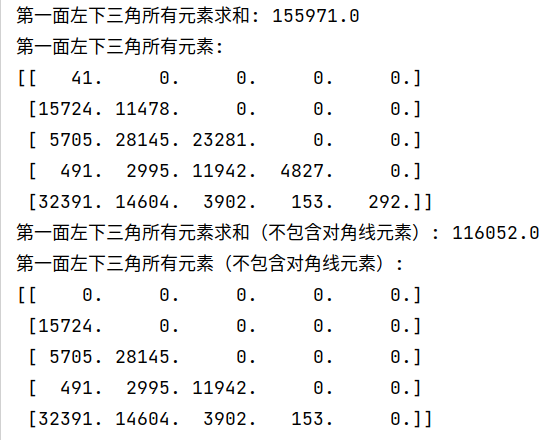
**本题数据源grade0.txt从附件﻿中下载到本地电脑中。编写程序完成如下功能：(1)采用numpy.genfromtxt()函数导入grade0.txt为数组。(2)输出数组的维数、形状、元素总个数。(3)将数组重新设置形状为三维数组4\*5\*5，以每一面为单位输出均值、最小值、最大值。(4)定义函数统计每一面高于均值元素个数。(5)计算第一面左下三角所有元素之和，并输出，同时输出第一面左下三角所有元素。**

#### 程序源代码：

**import numpy as np  
  
# (1) 导入grade0.txt为数组  
arr = np.genfromtxt('grade0.txt', delimiter=',')  
  
# (2) 输出数组的维数、形状、元素总个数  
print("数组的维数:", arr.ndim)  
print("数组的形状:", arr.shape)  
print("数组的元素总个数:", arr.size)  
  
# (3) 将数组重新设置形状为三维数组4\*5\*5，以每一面为单位输出均值、最小值、最大值  
arr\_3d = np.reshape(arr, (4, 5, 5))  
print("三维数组的形状:", arr\_3d.shape)  
for i in range(4):  
 print("第{}面的均值:".format(i+1), np.mean(arr\_3d[i]))  
 print("第{}面的最小值:".format(i+1), np.min(arr\_3d[i]))  
 print("第{}面的最大值:".format(i+1), np.max(arr\_3d[i]))  
  
# (4) 定义函数统计每一面高于均值元素个数  
def count\_above\_mean(arr):  
 mean = np.mean(arr)  
 count = (arr > mean).sum()  
 return count.item()  
  
for i in range(4):  
 print("第{}面高于均值的元素个数:".format(i+1), count\_above\_mean(arr\_3d[i]))  
  
  
elements = np.tril(arr\_3d[0, :, :], k=0)  
print("第一面左下三角所有元素求和:", elements.sum())  
print("第一面左下三角所有元素:")  
print(elements)  
  
  
# 不包含对角线元素  
elements = np.tril(arr\_3d[0, :, :], k=-1)  
print("第一面左下三角所有元素求和（不包含对角线元素）:", elements.sum())  
print("第一面左下三角所有元素（不包含对角线元素）:")  
print(elements)**

#### 运行结果：





# **实验小结**

**本次实验中，我们学习并实践了numpy库的应用，具体涉及了numpy数组的基本操作、创建、索引和切片方法、常用函数和运算。通过对numpy库的学习，我们对其在数据分析和科学计算中的强大作用有了更深入的认识和理解。我们了解了numpy数组的基本操作，包括创建、赋值、重塑、转置、合并和分裂等。接着，我们学习了如何使用索引和切片来访问和修改数组中的元素。同时，我们还学习了numpy库中的一些常用函数和运算，如数学函数、逻辑函数、排序和统计函数等。这些numpy的知识和技能在今后的学习和工作中会对我们产生重要的帮助和作用。**