1、x为1×2的矩阵，y为7×2的矩阵

np.sqrt((x1-x2)\*\*2+(y1-y2)\*\*2) 求欧式距离，并对结果进行排序

import numpy as np

# 计算欧几里得距离

def euclidean\_distance(x, y):

    # 计算点差

    diff = x - y

    # 计算平方和

    square\_sum = np.sum(diff \*\* 2, axis=1)

    # 计算欧几里得距离

    distance = np.sqrt(square\_sum)

    return distance

# 对距离进行排序

def sort\_distance(distance):

    # 使用 argsort 函数获取排序后的索引

    sorted\_idx = np.argsort(distance)

    # 使用 sorted\_idx 对距离进行排序

    sorted\_distance = distance[sorted\_idx]

    return sorted\_distance

# 测试代码

x = np.array([[1, 3]])

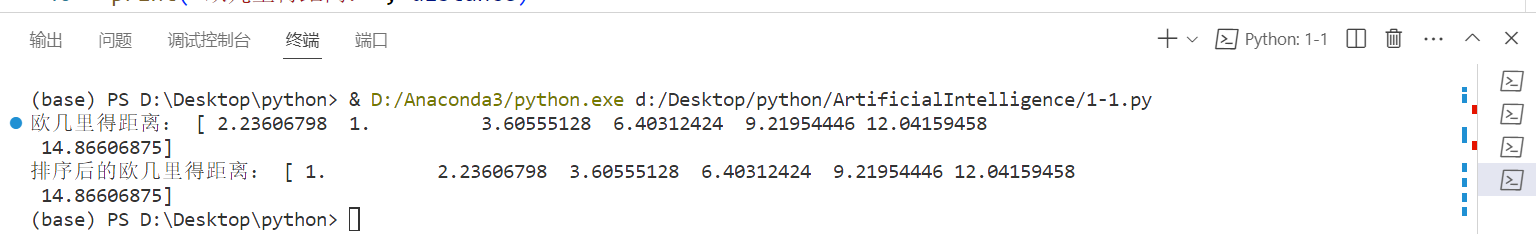
y = np.array([[0, 1], [2, 3], [4, 5], [6, 7], [8, 9], [10, 11], [12, 13]])

distance = euclidean\_distance(x, y)

sorted\_distance = sort\_distance(distance)

print("欧几里得距离：", distance)

print("排序后的欧几里得距离：", sorted\_distance)



2、读取stock\_data.csv数据，

（1）提取数值部分的数据，输出数据的维度，

（2）对每一列求均值、最大值、最小值并输出，

（3）选择其中一列作为排序依据对整个数据进行排序

（4）删除最后一列，对删除后的数据进行归一化处理，归一化公式为(x-μ)/(max-min),输出归一化后的数据

import pandas as pd

# 读取数据

data = pd.read\_csv(

    "D:\\Desktop\\python\\ArtificialIntelligence\\stock\_data.csv",

    encoding="gbk")

# (1) 提取数值部分的数据，输出数据的维度

numerical\_data = data.select\_dtypes(include=["number"])

print("数据维度：", numerical\_data.shape)

# (2) 对每一列求均值、最大值、最小值并输出

for column in numerical\_data.columns:

    mean = numerical\_data[column].mean()

    max\_value = numerical\_data[column].max()

    min\_value = numerical\_data[column].min()

    print(f"列：{column}")

    print(f"均值：{mean}")

    print(f"最大值：{max\_value}")

    print(f"最小值：{min\_value}\n")

# (3) 选择其中一列作为排序依据对整个数据进行排序

sort\_column = numerical\_data.columns[0]

sorted\_data = numerical\_data.sort\_values(by=sort\_column)

print("排序后的数据：", sorted\_data)

# (4) 删除最后一列，对删除后的数据进行归一化处理

normalized\_data = sorted\_data.drop(columns=sorted\_data.columns[-1])

for column in normalized\_data.columns:

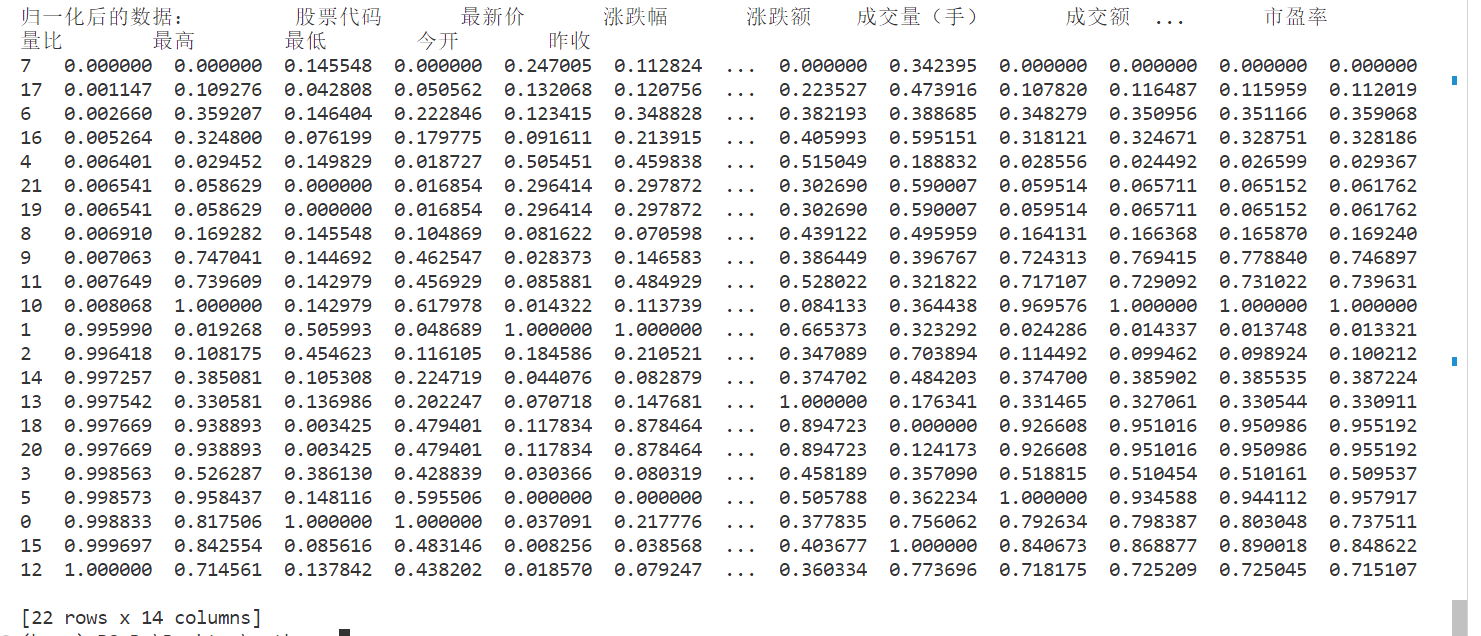
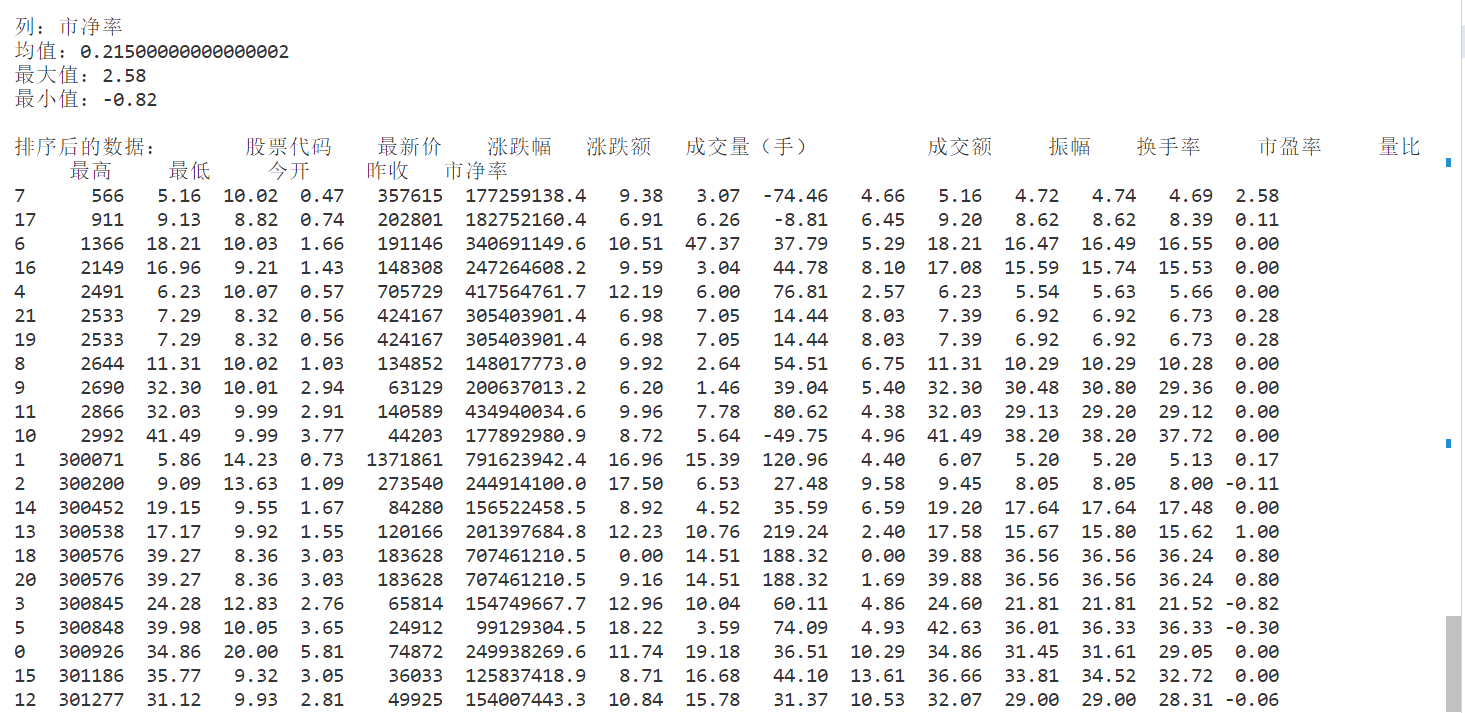
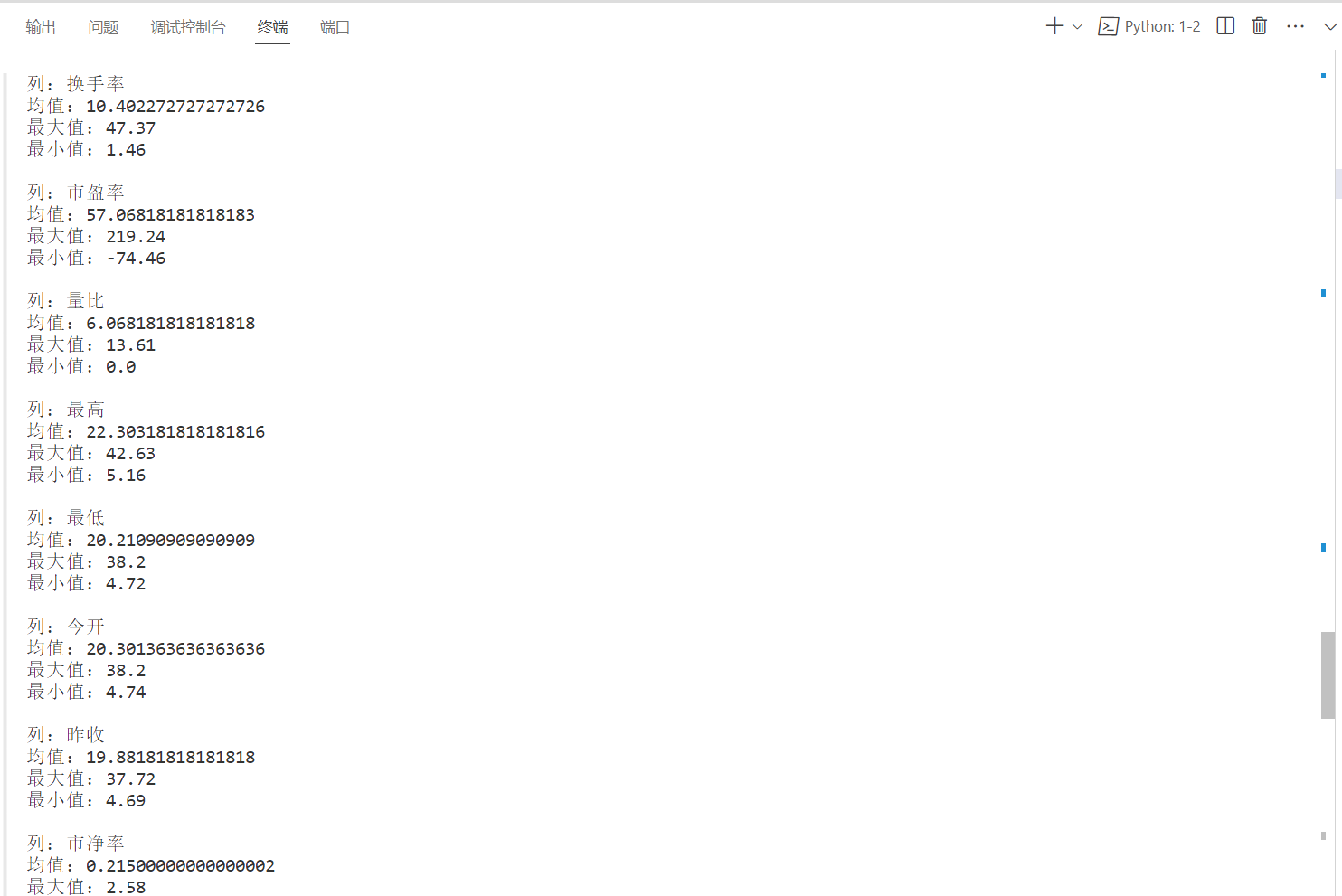
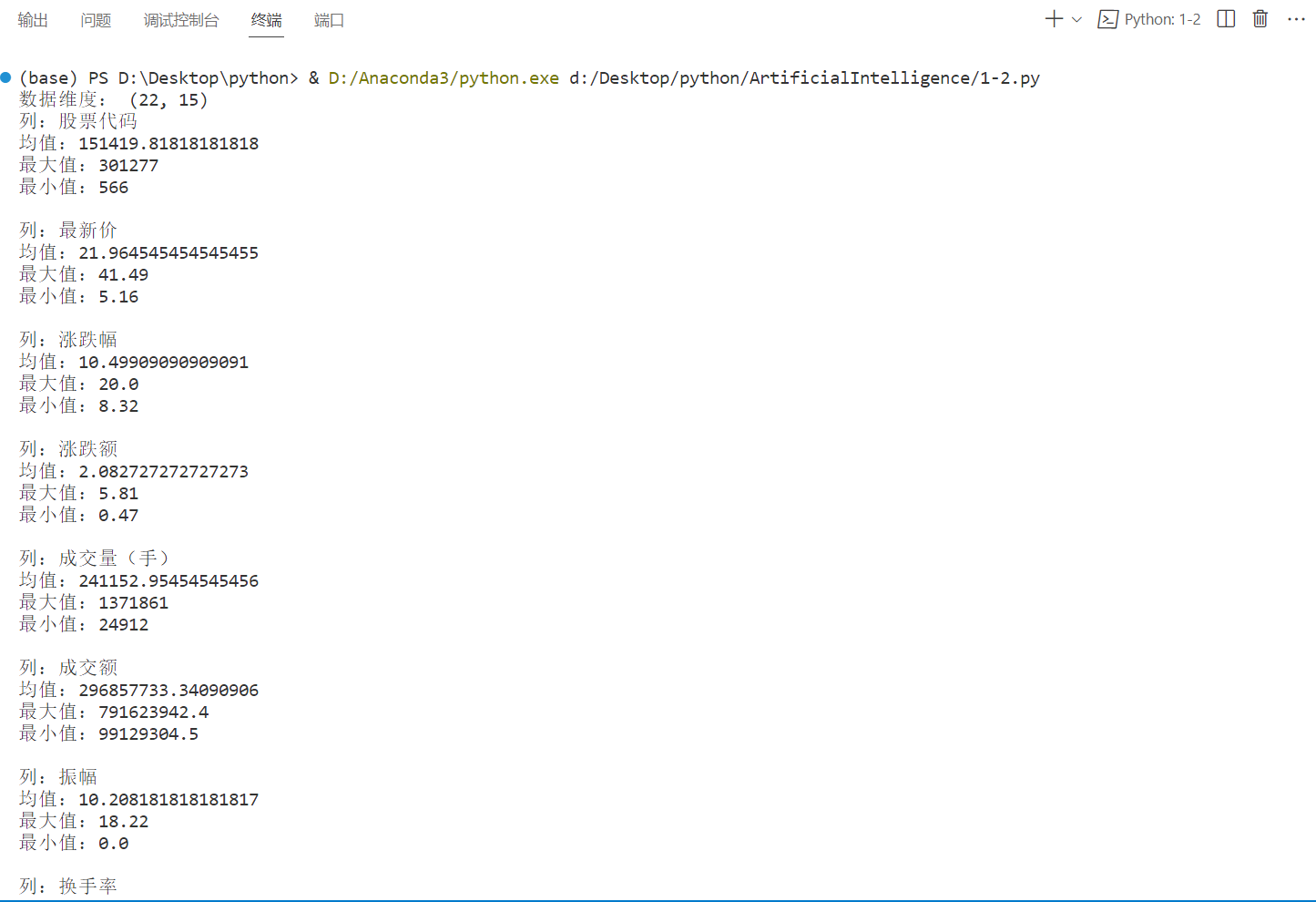
    max\_value = normalized\_data[column].max()

    min\_value = normalized\_data[column].min()

    normalized\_data[column] = (

        normalized\_data[column] - min\_value) / (max\_value - min\_value)

print("归一化后的数据：", normalized\_data)



3、读取iris.csv数据

（1）将前四列(不含编号列)读取到data变量中，最后一列读取到labels变量中，

（2）输出labels的类别及每个类别出现的数量

（3）输出labels值为setosa对应的样本

（4）筛选第一列值小于5.0，第三列值大于1.5的所有data

import pandas as pd

import os

import sys

os.chdir(sys.path[0])

# 读取数据

iris\_data = pd.read\_csv("iris.csv")

# (1) 将前四列(不含编号列)读取到data变量中，最后一列读取到labels变量中

data = iris\_data.iloc[:, 1:5]

labels = iris\_data.iloc[:, -1]

# (2) 输出labels的类别及每个类别出现的数量

print("类别及数量：")

print(labels.value\_counts())

# (3) 输出labels值为setosa对应的样本

setosa\_data = data[labels == "setosa"]

print("setosa样本：", setosa\_data)

# (4) 筛选第一列值小于5.0，第三列值大于1.5的所有data

filtered\_data = data[(data.iloc[:, 0] < 5.0) & (data.iloc[:, 2] > 1.5)]

print("筛选后的data：", filtered\_data)

