1. 创建宿舍成员基本信息表(DataFrame)。

源程序：

import pandas as pd

# 创建数据

data = {

'编号': [1, 2, 3, 4, 5, 6],

'姓名': ['郭康欣', '边水柔', '崔以墨', '耿雨菲', '郭硕', '占位'],

'学号': ['2305014208', '2305014202', '2305014204', '2305014207', '2305014209', '2300000000'],

'性别': ['女', '女', '女', '女', '女', '女'],

'年龄': [19, 19, 19, 19, 19, 19],

'出生年月日': ['2004-01-15', '2004-03-22', '2004-05-10', '2004-07-18', '2004-09-25', '2004-01-01'],

'身高': [165, 160, 158, 162, 167, 160],

'体重': [50, 48, 47, 52, 55, 60],

'爱好': ['阅读', '跑步', '音乐', '绘画', '篮球', '占位'],

'宿舍号': [101, 101, 101, 101, 101, 101]

}

# 创建DataFrame

df = pd.DataFrame(data)

display(df)

过程性结果：



过程性结论：

成功创建了包含6名成员信息的DataFrame。表格涵盖了编号、姓名、学号、性别、年龄等关键信息，为后续分析奠定了基础。

2. 求该宿舍成员的身高、体重的均值、最大值、最小值、标准差等统计信息，并按出生年月日对该宿舍成员进行排序。

源程序：

# 计算统计信息

height\_stats = df['身高'].astype(float).describe()

weight\_stats = df['体重'].astype(float).describe()

print("身高统计信息：")

display(height\_stats)

print("体重统计信息：")

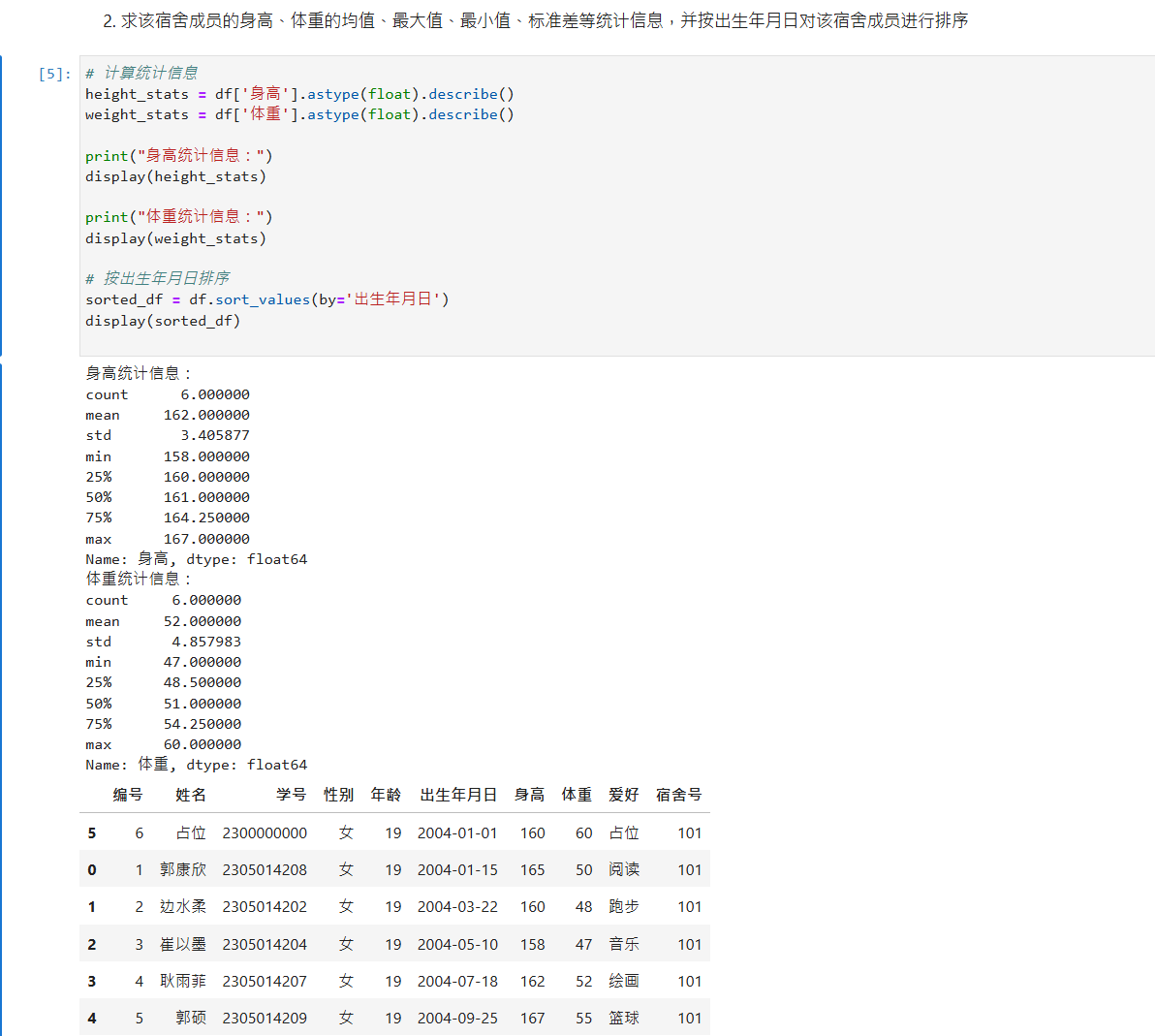
display(weight\_stats)

# 按出生年月日排序

sorted\_df = df.sort\_values(by='出生年月日')

display(sorted\_df)

过程性结果：



过程性结论：

计算了身高和体重的统计数据，可以看出成员间的身体特征差异。例如，身高的平均值、最大值、最小值等。按出生日期排序后，可以清晰地看到成员的年龄顺序。

3. 创建2023-2024学年第一学期宿舍成员科目成绩信息表

源程序：

# 创建成绩数据

grades = {

'编号': [1, 2, 3, 4, 5, 6],

'姓名': ['郭康欣', '边水柔', '崔以墨', '耿雨菲', '郭硕', '占位'],

'学号': ['2305014208', '2305014202', '2305014204', '2305014207', '2305014209', '2300000000'],

'数学分析(一)': [85, 90, 78, 88, 92, 90],

'解析几何': [88, 85, 80, 82, 89, 90],

'大学计算机': [90, 92, 85, 87, 91, 90],

'大学英语(1)': [87, 89, 83, 86, 88, 90],

'思想道德修养与法律基础': [86, 88, 81, 84, 87, 90]

}

# 创建DataFrame

grades\_df = pd.DataFrame(grades)

display(grades\_df)

过程性结果：



过程性结论：

成功建立了包含5个科目成绩的表格。这为后续的成绩分析提供了必要的数据基础。

4. 使用isnull()对宿舍成员成绩信息表进行缺失值检测

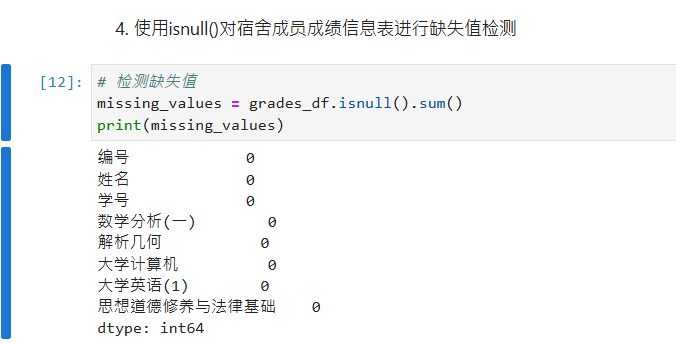
源程序：

# 检测缺失值

missing\_values = grades\_df.isnull().sum()

print(missing\_values)

过程性结果：



过程性结论：

使用isnull()函数检测了成绩表中的缺失值。这一步骤确保了数据的完整性，为后续分析提供了可靠保障。

5. 宿舍成员科目成绩求和，并追加到表最后一列，并按成绩进行排序

源程序：

# 计算总成绩

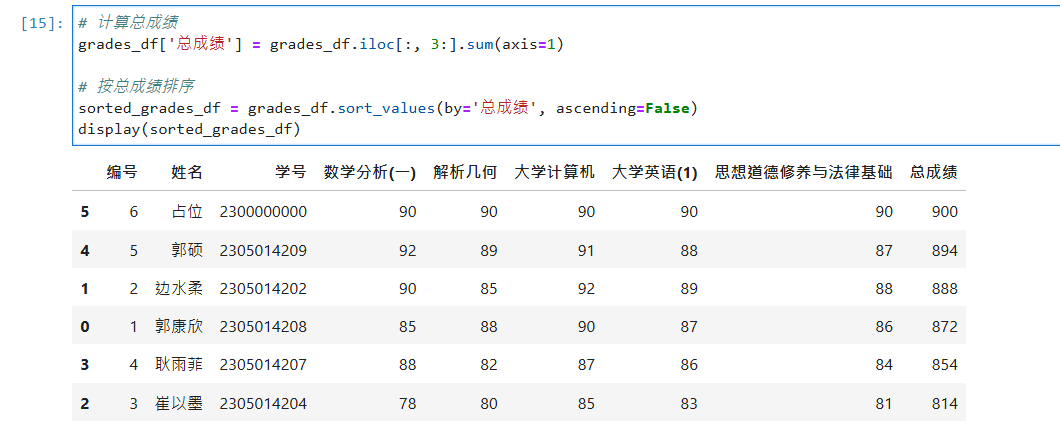
grades\_df['总成绩'] = grades\_df.iloc[:, 3:].sum(axis=1)

# 按总成绩排序

sorted\_grades\_df = grades\_df.sort\_values(by='总成绩', ascending=False)

display(sorted\_grades\_df)

过程性结果：



过程性结论：

计算了总成绩并进行了排序，便于比较不同学生间的成绩差异。

6. 用describe()方法对各科目成绩进行描述性统计

源程序：

# 描述性统计

stats = grades\_df.describe()

display(stats)

过程性结果：



过程性结论：

对各科目成绩进行了描述性统计，可以快速了解每门课程的整体表现，如平均分、最高分、最低分等。

7. 使用Matplotlib库分别绘制数学分析（一）成绩的柱状图

源程序：

import matplotlib.pyplot as plt

plt.rcParams['font.sans-serif']=['SimHei'] # 用来正常显示中文标签（中文乱码问题）

# 绘制柱状图

plt.bar(grades\_df['姓名'], grades\_df['数学分析(一)'])

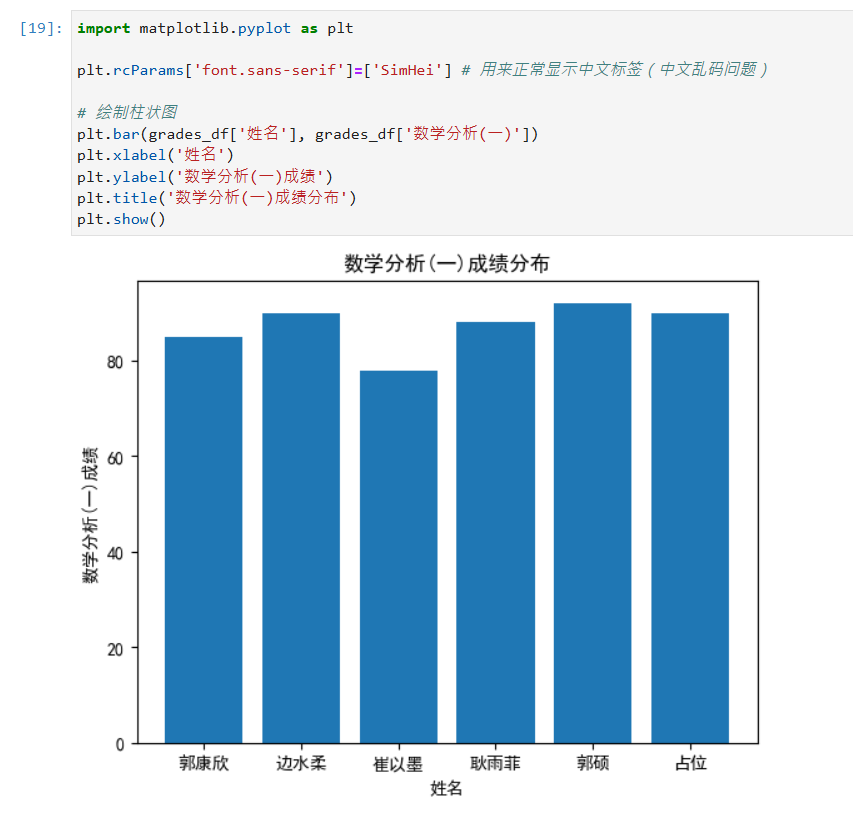
plt.xlabel('姓名')

plt.ylabel('数学分析(一)成绩')

plt.title('数学分析(一)成绩分布')

plt.show()

过程性结果：



过程性结论：

数学分析（一）成绩柱状图：

通过柱状图直观展示了每个学生在数学分析（一）科目上的成绩，便于比较个体差异。

8. 使用Matplotlib库分别绘制数学分析（一）成绩的饼图

源程序：

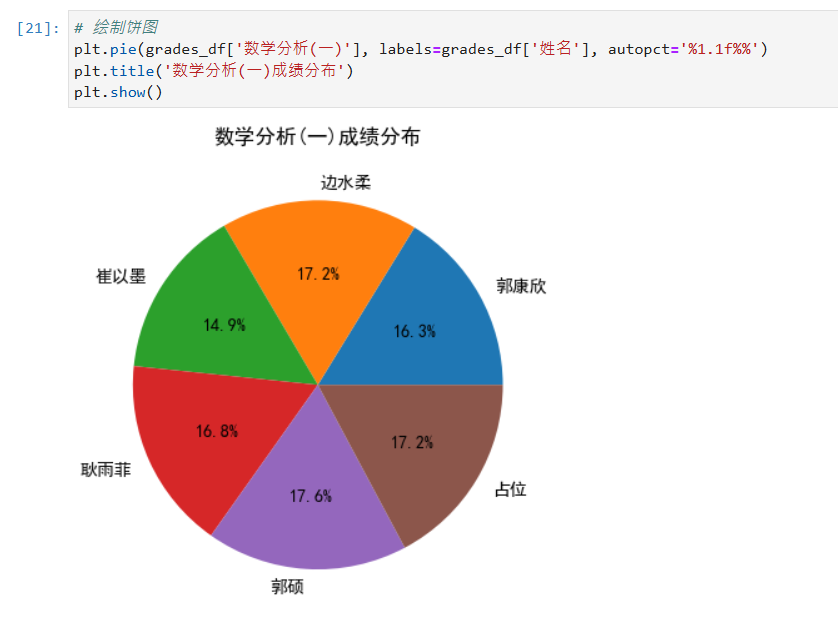
# 绘制饼图

plt.pie(grades\_df['数学分析(一)'], labels=grades\_df['姓名'], autopct='%1.1f%%')

plt.title('数学分析(一)成绩分布')

plt.show()

过程性结果：



过程性结论：

数学分析（一）成绩饼图：

饼图展示了数学分析（一）成绩的分布比例，有助于理解成绩的整体构成。

9. 用箱线图分别对数学分析（一）成绩进行异常值检测

源程序：

# 绘制箱线图

plt.boxplot(grades\_df['数学分析(一)'])

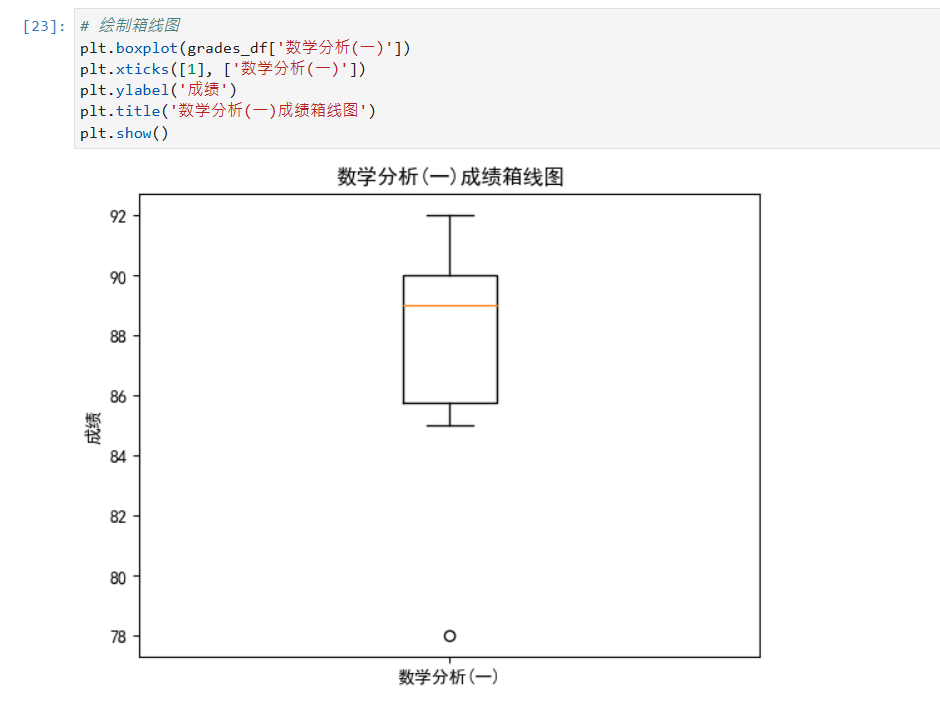
plt.xticks([1], ['数学分析(一)'])

plt.ylabel('成绩')

plt.title('数学分析(一)成绩箱线图')

plt.show()

过程性结果：



过程性结论：

数学分析（一）成绩箱线图：

箱线图有助于识别该科目中的潜在异常值，可以发现是否有特别突出或落后的情况。

10. 按姓名和学号将宿舍成员基本信息表和成绩信息表进行合并

源程序：

# 合并数据

merged\_df = pd.merge(df, grades\_df, on=['姓名', '学号'])

display(merged\_df)

过程性结果：



过程性结论：

将基本信息和成绩信息合并，创建了一个全面的学生信息表，便于进行更加全面和深入的分析。

总体结果分析：

通过这一系列的数据处理和分析，我们可以得出以下结论：

宿舍成员基本情况：本宿舍有6名成员，全部为女性，年龄均为19岁，但出生日期有所不同。身高和体重数据显示成员间存在一定差异。

学业表现方面，从成绩表可以看出，学生们在不同科目上的表现各有千秋。通过总成绩排序，可以大致了解学生的整体学习情况。

数学分析（一）科目这里，通过可视化分析，我们可以直观地比较学生在这门课程上的表现差异。箱线图的异常值检测有助于发现是否有特别突出或落后的情况。

通过缺失值检测，我们确保了数据的完整性，为分析结果的可靠性提供了保障。

综合信息中，最后的信息表合并为我们提供了一个全面的视角，可以同时考虑学生的基本信息和学习成绩，有利于进行更深入的个性化分析。