实验 1 Python 开发环境搭建及基础数据结构编程实践实习指导书

一、实验基本信息

项目	内容
实验名称	Python 开发环境搭建及基础数据结构编程实践
授课课程	大数据分析
授课班级	工业工程 2241 班
实验学时	2 学时（90 分钟）
实验类型	验证性 + 操作性实验
实验目标	1. 掌握 Spyder 开发工具基础操作，培养环境配置的严谨性；2. 掌握 Python 库安装与基础数据结构操作，养成代码编写的规范性；3. 掌握选择 / 循环结构代码编写，提升问题排查的细致性。

二、实验原理

1. **Anaconda 与 Spyder 环境原理**

Anaconda 是 Python 的集成发行版，预装了数据分析常用库（如 NumPy、Pandas），并自带 Spyder 集成开发环境（IDE）。Spyder 界面包含**编辑器（编写代码）**、**IPython 控制台（运行代码）**、**变量资源管理器（查看变量）**、**文件浏览器（管理文件）** 四大核心组件，实现 “编写 - 运行 - 调试 - 查看” 一体化，避免单独安装 Python 及库的兼容性问题。

2. **Python 基础数据结构原理**

Python 常用基础数据结构包括：

1. 列表（List）：用[]定义，元素可修改、可混合类型（如[1, "test", 3.5]），支持添加、删除、切片等操作；
2. 字典（Dictionary）：用{}定义，以“键 - 值对”存储（如{"name": "Li", "score": 90}），键唯一且不可变，值可修改；
3. 选择结构（if-else）：通过条件判断执行不同代码块，语法需严格遵循“冒号 + 缩进”；
4. 循环结构（for/while）：for 循环用于遍历序列（如列表、range 对象），while 循环基于条件重复执行，需避免死循环。

1. **Python 库安装原理**

常用库安装工具为 pip（Python 官方包管理器）和 conda（Anaconda 自带管理器），通过命令行指令从远程仓库下载并安装库文件。若网络不佳可配置国内镜像（如清华镜像），确保库版本与 Python 版本兼容。

三、实验环境准备

1. **硬件要求**

计算机内存≥4GB，硬盘剩余空间≥20GB（Anaconda 安装需约 10GB，后续库安装需预留空间）。

2. **软件要求**

1. 操作系统：Windows 10/11（64 位，避免 32 位系统兼容性问题）；
2. Anaconda 版本：建议 Anaconda3-2024.02 及以上（预装 Python 3.9+，兼容后续数据分析库）；
3. 提前检查：确保计算机未安装旧版 Python（避免环境变量冲突），安装路径无中文、空格或特殊字符（如 D:\Anaconda3，禁止 D:\软件\Anaconda）。

四、实验内容与操作步骤（含细节要求）

本实验分 **4 个核心模块**，每个模块需严格遵循操作细节，记录关键现象，培养认真细致的科研习惯。

模块 1：Anaconda 与 Spyder 启动

操作目标

掌握 Anaconda 环境启动流程，验证 Spyder 界面组件完整性，排查启动异常。

操作步骤

1. **启动 Anaconda 与 Spyder**

    1. 点击 Windows 开始菜单→找到 “Anaconda3” 文件夹→双击 “Spyder” 图标（首次启动需加载 5-10 分钟，耐心等待，避免重复点击）；
    2. 若启动失败（如 “找不到组件”）：

① 检查 Anaconda 安装路径是否有中文 / 空格；

② 打开 “Anaconda Prompt”，输入 spyder --reset 重置 Spyder 配置，再重新启动；

③ 记录异常现象及解决过程（实验报告需体现）。

1. **验证 Spyder 界面组件**

启动成功后，确认界面包含以下 4 个核心组件（缺一不可，缺失需排查）：

1. 编辑器（左侧，显示 “untitled0.py” 空白文件）；
2. IPython 控制台（右下方，显示 “Python 3.x.x | Anaconda, Inc.” 版本信息）；
3. 变量资源管理器（右上方，初始为空，显示 “Name/Type/Size/Value” 列）；
4. 文件浏览器（左上方，显示计算机文件目录）；
5. 细节要求：截图保存 Spyder 完整界面（命名为 “Spyder 界面.png”，实验报告需附）。

模块 2：Spyder 基础操作

操作目标

掌握 Spyder 文件管理、代码运行与调试的基础操作，养成规范的编程习惯。

操作步骤

1. **新建并保存 Python 文件**

    1. 点击 Spyder 菜单栏「File」→「New File」，新建空白文件；
    2. 点击菜单栏「File」→「Save As」，选择个人实验文件夹（建议路径：D:\IE2241_BigData\Experiment1），文件名命名为 student_id_name_ex1.py（如 22010101_ZhangSan_ex1.py）；
    3. 细节要求：

① 文件名**禁止含中文、空格或特殊字符**（避免运行时路径解析错误）；

② 必须保存为.py格式（Spyder 仅识别该格式文件为可运行 Python 脚本）。

1. **代码运行与调试基础**

    1. 在编辑器中输入测试代码：

```
# 测试 Spyder 运行功能
print("Hello, Big Data Analysis!")
num_list = [1, 2, 3, 4, 5]
print("列表元素：", num_list)
print("列表长度：", len(num_list))
```

    2. 运行代码：两种方式验证（均需操作，对比差异）：

① 点击工具栏「Run」按钮（绿色三角形），或按快捷键 F5；

② 右键点击编辑器空白处→「Run selection or current line」（仅运行选中代码）；

    3. 查看结果：IPython 控制台显示 “Hello, Big Data Analysis!” 等输出，变量资源管理器显示 num_list 变量（类型为 list，长度 5）；

    4. 调试基础：在代码第 2 行左侧点击，添加 “断点”（红色圆点），按 Ctrl+F5 启动调试，观察控制台 “Debug” 模式提示，按 F10 单步执行，记录变量变化（培养调试时的细致观察习惯）。

模块 3：Python 新库安装

操作目标

掌握 pip/conda 库安装方法，排查安装异常，培养环境配置的严谨性。

操作步骤

1. **打开 Anaconda Prompt**

    1. 点击 Windows 开始菜单→“Anaconda3” 文件夹→双击 “Anaconda Prompt”（建议以 “管理员身份” 运行，避免权限不足）；

    2. 验证环境：在 Prompt 中输入 python --version，显示 Python 3.9 + 版本（与 Anaconda 预装版本一致）；输入 conda --version，显示 conda 版本（如 23.10.0），若报错需检查 Anaconda 环境变量是否配置（路径：Anaconda3\Scripts）。

1. **安装第三方库（以 jieba 库为例）**

2. 检查库是否已安装：输入 pip list | findstr jieba（Windows），若无输出则未安装；
3. 在线安装：输入 pip install jieba，观察安装过程（显示 “Collecting jieba”“Successfully installed jieba-x.x.x”）；
4. 细节处理：

① 若安装速度慢：切换清华镜像，输入 pip install -i https://pypi.tuna.tsinghua.edu.cn/simple jieba；

② 若安装失败（如 “Read timed out”）：重试 2-3 次，或输入 conda install jieba（conda 管理器兼容性更强）；

③ 验证安装：安装完成后，输入 python 进入 Python 交互模式，输入 import jieba，无报错则安装成功（输入 exit()退出交互模式）；

4. 截图保存：Prompt 中 “Successfully installed jieba-x.x.x” 提示界面（命名为 “jieba 安装成功.png”）。

1. **Spyder 中验证库**
   1. 返回 Spyder，在编辑器中输入代码：

```
import jieba
text = "大数据分析是工业工程的重要工具"
words = jieba.lcut(text)
print("分词结果：", words)
```

2. 运行代码，IPython 控制台显示分词结果（如['大数据', '分析', '是', '工业工程', '的', '重要', '工具']），无 “ModuleNotFoundError” 则库可用。

模块 4：基础数据结构与流程控制编程

操作目标

掌握列表、字典的创建与操作，编写含选择 / 循环结构的代码，养成规范编码与细致调试的习惯。

操作任务与步骤（每任务需附代码、运行结果截图及问题记录）

任务 1：列表（List）创建与操作

1. 需求：创建混合类型列表，实现添加、插入、删除、切片操作；

2. 代码编写（需加注释）：

```
# 任务 1：列表操作实践
# 1. 创建混合类型列表（含整型、字符串、布尔值）
mix_list = [2025, "工业工程 2241", 3.8, True, ["Python", "Java"]]
print("初始列表：", mix_list)
print("列表长度：", len(mix_list))
# 2. 添加元素（末尾添加）
mix_list.append("大数据分析")
print("添加后列表：", mix_list)
# 3. 插入元素（指定位置插入，索引从 0 开始）
mix_list.insert(2, "实验 1") # 在第 3 个元素前插入
print("插入后列表：", mix_list)
# 4. 删除元素（按值删除）
mix_list.remove(True)
print("删除 True 后列表：", mix_list)
# 5. 切片操作（提取第 2-5 个元素，左闭右开）
slice_list = mix_list[1:5]
print("切片结果（索引 1-4）：", slice_list)
```

1. 运行与验证：
    1. 运行代码，查看 IPython 控制台输出是否与预期一致；
    2. 在变量资源管理器中查看 mix_list“Value” 列，确认元素变化；
    3. 细节排查：若出现 “ValueError: list.remove (x): x not in list”，检查删除的元素是否在列表中；若切片结果为空，检查索引范围是否正确。

任务 2：字典（Dictionary）创建与键值操作

1. 需求：创建学生信息字典，实现键值访问、添加、修改与遍历；
2. 代码编写：

```
# 任务 2：字典操作实践
# 1. 创建学生信息字典（键：姓名、年龄、专业、成绩）
student_dict = {
    "name": "Wang Wu",
    "age": 21,
    "major": "Industrial Engineering",
    "scores": [85, 92, 78]  # 值为列表类型
}
print("初始学生字典：", student_dict)
# 2. 访问键值（两种方式）
print("学生姓名：", student_dict["name"])  # 直接访问（键不存在报错）
print("学生年龄：", student_dict.get("age"))  # get 方法（键不存在返回 None）
# 3. 添加新键值对
student_dict["gender"] = "Male"
print("添加性别后字典：", student_dict)
# 4. 修改已有键值
student_dict["scores"][2] = 88  # 修改列表中的第三个成绩（索引 2）
print("修改成绩后字典：", student_dict)
# 5. 遍历字典（键-值对）
print("\n 遍历字典：")
for key, value in student_dict.items():
    print(f"{key}: {value}")
```

1. 运行与验证：
    1. 确认控制台输出 “修改成绩后字典” 中 scores 为[85, 92, 88]；
    2. 细节注意：字典的键**不可重复**（若添加重复键，会覆盖原有值），键需为不可变类型（如字符串、整型，不能为列表）。

任务 3：选择结构（if-else）代码编写

1. 需求：输入学生成绩，判断及格（≥60）/ 不及格，处理非数字输入异常；
2. 代码编写：

```
# 任务 3：选择结构实践（成绩判断）
# 1. 获取用户输入（input 默认返回字符串，需转换类型）
score_input = input("请输入学生成绩（0-100）：")
# 2. 异常处理（避免输入非数字导致报错）
try:
    score = float(score_input)  # 转换为浮点型
    # 3. 条件判断
    if score >= 90 and score <= 100:
        print(f"成绩：{score}，等级：优秀")
    elif score >= 60 and score < 90:
        print(f"成绩：{score}，等级：及格")
    elif score >= 0 and score < 60:
        print(f"成绩：{score}，等级：不及格")
    else:
        print("成绩超出 0-100 范围，请重新输入！")
except ValueError:
    print("输入错误！请输入数字（如 85、92.5）。")
```

1. 运行与测试：
    1. 测试 3 种场景：输入 “88”（输出 “及格”）、输入 “55”（输出 “不及格”）、输入 “abc”（输出 “输入错误”）；
    2. 细节要求：记录每种场景的输入与输出，确认异常处理生效（避免程序因输入错误崩溃）。

任务 4：循环结构（for/while）代码编写

1. 需求 1（for 循环）：计算 1-100 的偶数和；

代码：

```
# 任务 4-1：for 循环计算 1-100 偶数和
even_sum = 0
```

```
for num in range(2, 101, 2):  # range(起始, 结束+1, 步长)
    even_sum += num
print("1-100 偶数和：", even_sum)  # 预期结果：2550
```

1. 需求 2（while 循环）：猜数字游戏（随机生成 1-100 的数，用户猜，提示 “大了”/“小了”，直到猜对）；

代码：

```
# 任务 4-2：while 循环猜数字游戏
import random
target_num = random.randint(1, 100)  # 生成 1-100 随机数
guess_count = 0  # 记录猜测次数
print("欢迎参与猜数字游戏！数字范围 1-100")
while True:
    guess_input = input("请输入你猜的数字：")
    try:
        guess_num = int(guess_input)
        guess_count += 1
        if guess_num > target_num:
            print("大了！再试试~")
        elif guess_num < target_num:
            print("小了！再试试~")
        else:
            print(f"恭喜猜对！数字是{target_num}，你猜了{guess_count}次")
            break  # 猜对退出循环
    except ValueError:
        print("输入错误！请输入整数。")
```

1. 运行与验证：
    1. 确保 for 循环输出 “2550”，while 循环能正常提示 “大了 / 小了”，直至猜对；
    2. 细节排查：若 while 循环出现死循环（如未加 break），立即停止运行（Spyder 工具栏「Stop」按钮），检查循环退出条件。

五、实验注意事项

1. **环境配置细节**
    1. Anaconda 安装路径**禁止含中文、空格或特殊字符**（如 “D:\ 软件 \Anaconda” 会导致 Spyder 启动失败）；
    2. Spyder 中新建文件必须先保存为.py 格式，否则点击 “Run” 会提示 “Save file before running”。
1. **库安装细节**
    1. 安装库前先通过 pip list/conda list 检查是否已安装，避免重复安装；
    2. 若出现 “Permission denied”（权限不足），关闭 Anaconda Prompt 后重新以 “管理员身份” 运行。
1. **代码编写细节**
    1. 缩进统一：使用 4 个空格作为 1 级缩进，**禁止 Tab 键与空格混用**（会导致 “IndentationError”）；
    2. 变量命名：遵循 Python 规范（以字母 / 下划线开头，不使用关键字如 if/for，建议 “小写 + 下划线” 如 student_score）；
    3. 注释规范：关键代码块加单行注释（# 注释内容），复杂逻辑加多行注释（""" 注释内容"""），便于后续调试与他人阅读。
1. **调试排查细节**
    1. 遇到代码报错，先查看 IPython 控制台 “Traceback (most recent call last)” 信息，定位 “File” 后的文件及 “line” 后的行号，重点检查该行语法（如括号不匹配、逗号遗漏）；
    2. 变量值异常时，在变量资源管理器中查看变量类型与值，或在代码中添加 print(变量名)打印中间结果，逐步排查。

六、实验报告要求

1. **报告结构**

需包含 “实验目的”“实验环境”“实验步骤与结果（附关键截图）”“问题与解决方法”“实验总结” 5 个核心部分。

2. **关键截图要求**
    1. 必附截图：Spyder 完整界面、jieba 库安装成功提示、任务 1-4 的代码运行结果（每张截图需标注 “图 1-Spyder 界面”“图 2-jieba 安装成功” 等，截图需清晰显示关键信息）。

1. **“问题与解决方法”要求**

需真实记录实验中遇到的至少 2 个问题（如“Spyder 启动失败”“库安装超时”“代码缩进报错”），并详细说明解决过程（例：“问题 1：Spyder 启动后变量资源管理器空白；解决：点击菜单栏「View」→「Panes」→「Variable Explorer」，勾选显示，重启 Spyder 后恢复”）。

2. **“实验总结”要求**

    1. 技术总结：提炼 Anaconda 环境搭建、数据结构操作、流程控制的核心要点（如“列表用 append 添加元素，字典用 get 访问键值更安全”）；
    2. 品质反思：反思实验中是否因细节疏忽导致问题（如“因文件名含中文导致代码无法运行，后续需严格遵循命名规范”“未检查输入类型导致程序崩溃，需重视异常处理”），提出改进方向。

七、思考题

1. 列表（List）和元组（Tuple）的核心区别是什么？结合实验中“修改列表元素”的操作，说明元组为何适合存储不可变数据（如学生学号、身份证号）。
2. 为什么工业工程专业学习大数据分析时，推荐使用 Anaconda 而非单独安装 Python？（从环境统一性、库兼容性、科研协作角度分析）。
3. 若实验中需安装的库在 pip 和 conda 管理器中均安装失败，你会尝试哪些备用方案？（提示：本地安装.whl 文件、更换 Python 版本）。
4. 代码中“缩进错误”属于语法错误还是逻辑错误？在科研编程中，如何避免此类低级错误影响实验进度？（提示：使用 Spyder 的“代码格式化”功能、养成“写一行运行一行”的习惯）。

八、实验考核标准

考核维度	考核要点（总分 100 分）	分值
环境搭建（20 分）	Anaconda 与 Spyder 启动正常（10 分）；jieba 库安装成功并验证可用（10 分）	20
代码完成（40 分）	任务 1-4 代码完整且运行正确（每任务 10 分，语法错误扣 5 分 / 处，逻辑错误扣 3 分 / 处）	40

报告质量（30 分）	结构完整（10 分）；截图清晰且标注规范（10 分）；“问题与解决方法”真实详细（10 分）	30
科研品质（10 分）	代码注释规范（3 分）；变量命名符合规范（3 分）；实验记录细致（4 分，如问题排查记录）	10