

AI 代码审查与调试实战

A 段按明确规格修正五个函数，B 段完成 IPv4 的独立测试、候选审查与回归，C 段记录 Ruff、调试器和命令行操作。保留来源、修改及真实检查结果。

截止：小作业 3 · 11 月 17 日 周二 23:59

本次目标

- 按函数规格检查返回值、顺序、边界和输入变化
- 区分接口、功能、效率与可读性问题
- 独立设计测试，保留候选、反例和修正版
- 理解 Ruff 诊断，观察递归调用帧
- 据实填写 AI 使用声明

A、B、C 三段同时开放，建议分别用约 40、35、15 分钟。A 段占本周小作业分的 60%，B 段占 40%；C 段只登记完成情况，内容纳入期末考核。平台显示题库原始自动分，人工说明由助教评阅；不能把“已提交 / 待评”当作人工分已经完成。

本周实验与个人作业

本实验按 [Level 1](#) 执行：允许 AI 辅助，须记录实际工具、生成或采用范围和真实验证。未使用 AI 或采用教师候选均可据实说明。[个人作业 1](#) 仍按 **Level 0** 执行，普通与相邻杆汉诺塔须独立编写，占总成绩 5%。本实验的递归调试只用课堂阶乘，不让 AI 改写个人作业。

本周实验归入[小作业 3](#)。HW3 覆盖第 7–9 周，截止见其提交页，按北京时间 (GMT+8) 显示。顶栏按最早一周显示 Level 0，不能替代每周题面的具体要求：第七周任务仍独立完成，第八周按本页 Level 1 执行。

本地第八周题包和提交卡已准备，服务器尚未同步。正式发布后使用 HW3 的第八周入口；当前先保存本地文件，不误交第七周题目，也不把本地保存称为已经在线提交。

准备与检查方式

使用已经安装的 Python 3.13、VS Code 和系统终端，不重新安装操作系统。Windows 使用此前准备的 WSL；环境仍有问题时请助教协助。

下面的候选均为教学构造，不是真实模型回答记录。先读规格、自己确定预期，再检查实现；不要预设每段“恰好一个错误”。功能正确但慢或难读，也有审查与改进的必要，不应为了报告数量编造逻辑缺陷。

五道 A 题可分别保存为代码文件，也可把五个函数保存在同一文件中，分别提交到对应代码卡。每道代码题的提交说明填写该题 AI 使用声明；完整审查过程另交文字卡。截图不得包含密码、令牌、私人文件或他人代码。

A 段 · 规格、改错与行为保持

五道代码题各 4 分，自动分合计 20 分。A 段审查记录四问各人工 1 分，合计 4 分；A 段原始满分为 **24 分**。

8-1 第二大的不同值

实现 `second_largest(nums: list[int]) -> int`。输入是至少包含两个不同值的整数列表，长度不超过 1000；返回第二大的**不同整数值**，不得修改输入。少于两个不同值不在本题输入范围内；不要自行引入 None、异常或其他返回约定。

候选：

```
def second_largest(nums: list[int]) -> int:
    ordered = sorted(nums, reverse=True)
    return ordered[1]
```

```
second_largest([3, 1, 2])      → 2
second_largest([5, 5, 3])      → 3
second_largest([-8, -1, -3])   → -3
```

排序后第二项未必是第二个不同值。修改后核对重复最大值、负数及原列表是否保持不变。返回值应为 int 整数，不用布尔值替代数字。

提交代码到 `w08-second-largest`，审查证据填写 A 段文字卡第 1 问。

8-2 标题大小写接口核查

实现 `title_case(s: str) -> str`，行为严格对应 Python 3.13 的字符串

`title()`：保留空白与标点，返回新字符串。输入长度不超过 200。不额外要求处理英语标题中冠词、缩写等自然语言规则。

候选：

```
def title_case(s: str) -> str:
    return s.capitalize_words()
```

`title()` 未在课堂逐项演示，可按本讲的文档核查方法查看[字符串方法](#)。比较 `capitalize()` 与 `title()`，确认不存在的接口不能仅凭名称使用。

```
title_case("hello python")    → "Hello Python"
title_case("")                → ""
title_case(" python AND c ") → " Python And C "
title_case("they're here")    → "They'Re Here"
```

最后一例是该 Python 接口的行为，不是自然语言标题建议。

提交代码到 `w08-title-case`，文档与真实测试填写 A 段文字卡第 2 问。

8-3 重复值检测与效率说明

实现 `has_duplicate(items: list[int]) -> bool`，判断整数列表中是否有重复值，不修改输入。输入长度不超过 1000；空列表和单元素返回 False，返回值必须是 bool。

候选：

```
def has_duplicate(items: list[int]) -> bool:
    for i in range(len(items)):
        for j in range(len(items)):
            if i != j and items[i] == items[j]:
                return True
    return False
```

```
has_duplicate([])          → False
has_duplicate([7])         → False
has_duplicate([3, 1, 2]) → False
has_duplicate([3, 1, 3]) → True
```

原功能逻辑可以正确，但对互异元素反复比较。使用 L06 的集合等已教方法减少重复工作，并保留功能和输入不变。

对 n 个互异整数，分别分析 `i != j` 和整数相等比较的求值次数；数的是源代码表达式，不是 CPU 指令。无需大规模计时，也不把两层循环的外观直接判为错误。自动测试核对功能，不声称已经证明复杂度；效率解释由助教查看。

提交代码到 `w08-has-duplicate`，分析与验证填写 A 段文字卡第 3 问。

8-4 字符频次与可读性

实现 `count_chars(s: str) -> dict[str, int]`，逐 Python 字符统计频次，区分大小写，空白和标点也计数。输入长度不超过 500，空字符串返回新空字典。每次调用的结果独立，不因后续调用或修改另一份结果而改变。

候选：

```
def count_chars(s):
    d = {}
    for i in range(len(s)):
        c = s[i]
        if c in d:
            d[c] = d[c] + 1
        else:
            d[c] = 1
    return d
```

```
count_chars("aba") → {"a": 2, "b": 1}
count_chars("AaA") → {"A": 2, "a": 1}
count_chars("") → {}
```

返回字典的键顺序不用于判分，频次为正 int 整数。这里统计字符，不统计 UTF-8 字节，也不把多个码点组成的视觉字形自动合并。

原行为未必有错。整理名称与遍历结构，再检查等价性；缺少类型提示或代码较长，不自动构成 PEP 8 违规或逻辑错误。

提交代码到 `w08-count-chars`，可读性与行为核对填写 A 段文字卡第 4 问。

8-5 最大相邻差与短输入

实现 `max_gap(nums: list[int]) -> int`：先按数值排序，返回相邻元素差的最大值，不修改输入。长度不超过 1000；少于两个元素返回整数 0，不用 False 替代 0。

候选：

```
def max_gap(nums: list[int]) -> int:
    ordered = sorted(nums)
    gaps = [ordered[i + 1] - ordered[i]
             for i in range(len(ordered) - 1)]
    return max(gaps)
```

```
max_gap([3, 9, 1, 15]) → 6
max_gap([]) → 0
max_gap([7]) → 0
max_gap([2, 2, 2]) → 0
max_gap([-3, -1, -10]) → 7
```

按规格处理短输入，复查普通、重复、负数与输入不变。可以使用已教的列表推导或普通循环，不必引入生成器表达式。

提交代码到 `w08-max-gap`，该题的来源、修改与测试填写其代码提交说明。

A 段审查记录

提交到 `w08-a-review-notes`，对应前四段候选，各 1 分：

1. 第二大的不同值：原候选反例、预期/实际、修改及输入不变。
2. 标题接口：文档依据、接口区别、空白或标点的实际测试。
3. 重复检测：原功能、两种检查的次数、修正版状态与功能核验。
4. 字符计数：原本正确的行为、命名/遍历改进、空串/大小写/重复调用结果。

每问都应有具体代码、输入或观察依据。不得把所有候选一律写为“不能运行”，也不以篇幅代替审查内容。

B 段 · IPv4 的规格、审查与回归

同一个问题分四步。修正版代码自动 8 分，说明卡四问各人工 1 分；B 段原始满分为 12 分。

B-1 规格与独立测试

接口为 `is_valid_ipv4(s: str) -> bool`，只考虑字符串输入。本题采用严格点分十进制：

- 恰好四段，以英文句点 `.` 分隔；

- 每段只含 ASCII `0–9`，数值在 0–255；
- 单个 `0` 合法，其他形式不允许前导零；
- 空段、正负号、前后或内部空白均非法；
- 对任意字符串按上述规则返回 bool，非法形式不抛出异常。

先独立写至少八个测试，至少三个合法、五个非法。保存输入、预期与理由，再查看候选。除数值之外，还要覆盖段数、空段、前导零及字符范围。

这里练习用已教的字符串、循环与判断实现检查，不直接调用 `ipaddress` 等成品校验接口，也不使用 L12 才介绍的正则表达式。自动用例只核对功能，方法约束由人工审查。

B-2 候选来源与运行结果

在线 AI 可用时，提交本题完整规格，保留实际工具/型号、提示与原代码；型号不明则注明。工具不可用或不愿使用外部服务时，可采用下面教师候选，写明未使用在线 AI，不要求付费账号。

用自己的测试运行候选，记录返回值或异常，指出它与规格的差别。先写测试能减少被实现带偏的影响，看完代码后也可以补充检查；不把测试顺序当作发现全部遗漏的保证。

▼ 教师候选：先完成 B-1 的独立测试，再查看

这是教学构造的候选，有意保留审查问题，不是真实 AI 输出或完整正确解：

```
def is_valid_ipv4(s: str) -> bool:
    parts = s.split(".")
    if len(parts) != 4:
        return False
    for part in parts:
        if not part.isdigit():
            return False
        if not 0 <= int(part) <= 255:
            return False
    return True
```

B-3 修正与回归

先尝试自己修正，按需再讨论。若候选恰好通过已有测试，说明覆盖范围并补充检查，不故意伪造失败。

保留原候选与修正版；复查普通输入、0/255、段数、前导零、非 ASCII 数字、符号与空白。针对字符串范围返回 bool，不以 None 的假值充当 False。

把完整修正版提交到 `w08-b-is-valid-ipv4`。该代码卡的说明填写来源、修改与实际检查；详细测试和候选记录填写 B 段文字卡。

B-4 说明与使用声明

提交到 `w08-b-ipv4-notes`，四问依次对应 B-1 至 B-4：

1. 规格与至少八个独立测试及预期。
2. 实际来源、原代码与检查结果。
3. 修改、修正版与回归结果。
4. 全实验使用声明，以及有限测试能支持什么、不能证明什么。

声明包含实际工具/型号、生成或采用范围、人工修改、真实验证与独立完成部分。未使用 AI 或采用教师候选同等据实说明；不编造模型、聊天记录或测试结果。不得发送密码、密钥、令牌或个人作业 1 代码。

C 段 · Ruff、调试器与命令行

提交到 `w08-c-tools-notes`，三个问答均为 0 分，由助教登记完成情况。卡片只接收文字，**不能上传图片**；截图保留本地供助教查看，填写文件名与观察值，不把本地文件名说成平台已经收到图片。

C-1 Ruff 诊断

使用 VS Code 的 Ruff 扩展，或在课程 Python 环境安装：

```
python3 -m pip install ruff
ruff check .
```

记录版本、自己的检查路径/规则、实际诊断数量和分类，解释三项不熟悉的诊断及处理决定。不足三项时，用下面练习文件补充，不编造警告：

```
import math

def f(l):
    unused = 1
    return missing_name + len(l)
```

```
ruff check --select F401,F821,F841,E741 practice.py
```

这是显式选择四项规则的命令，不代表 Ruff 默认全部启用。静态检查无警告不证明行为正确，不未经理解就自动修复。若 pip 报系统管理环境错误，不用管理员权限强制安装；选择课程解释器或编辑器扩展，虚拟环境在 L10 介绍。

C-2 递归调用帧

只使用课堂阶乘，不使用个人作业 1：

```
def factorial(n: int) -> int:
    if n == 0:
        return 1
    return n * factorial(n - 1)

print(factorial(3))
```

1. 启用 Python 与 Python Debugger，选中 Python 3.13；在递归调用行设断点。
2. 通过 Run and Debug 或 F5 启动，用 Step Into 进入调用，暂停到 $n=1$ 。
3. 分别选中三个帧，记录局部 n ；选择外层 `factorial(3)`，核对其 n 仍为 3。
4. 保存调用栈和局部变量截图，记录文件名；解释它与 L07 的缩进追踪怎样对应。
5. 试用 Over、Into、Out、Continue，记录每种动作。

系统和键盘可能需要 Fn，或直接使用工具栏动作。工具暂不可用时，保存真实 print 追踪并说明 GUI 截图未完成，不伪造截图；平台不会自动检查截图是否存在。

C-3 命令行复习

记录原有六步的实际命令与输出：建立家目录下 `cs1602/lab08`、进入并创建 `a.py/b.py/c.txt`、`ls -l`、运行包含 `print("hello")` 的 `a.py`、统计前七周 `.py` 普通文件数、查找自己的源码中的字面 `TODO` 或 `print(`。

```
mkdir -p ~/cs1602/lab08
cd ~/cs1602/lab08
touch a.py b.py c.txt
ls -l
# Use the editor to write print("hello") into a.py first.
python3 a.py

# Replace the path with a directory containing only your Labs 1-7 code.
find /path/to/labs1-7 -type f -name '*.py' | wc -l
grep -rn -F -e 'TODO' -e 'print(' /path/to/labs1-7
```

统计目录必须确实对应前七周，不把新建的 Lab08 文件混入。文件名不含换行符； `ls`

`-l | wc -l` 会把 total 等非文件行计入，不能当精确文件数。仅搜索自己的代码。

命令行复习速查

保留原有知识点，按需查阅，不在本次重新安装 Linux。

```
pwd
ls
ls -l
ls -a
ls -lh
cd /path/to/dir
cd ..
cd ~
cd -
mkdir mydir
mkdir -p a/b/c
touch file.txt
cp a.txt b.txt
cp -r dir1 dir2
mv a.txt b.txt
rm -i scratch.txt
rm -ri scratch-dir
```

Tab 补全取决于 shell。touch 创建缺失空文件，已存在时更新时间而不清空内容。复制或移动可能覆盖目标，先核对路径；rm 不经回收站，只用于自己创建的练习文件，不示范系统目录删除。Windows 未准备 WSL 时可按安装指引使用 `wsl --install`，需要系统条件、管理员权限及可能的重启，不保证立即可用。

```
cat file.txt
head file.txt
head -n 20 file.txt
tail file.txt
tail -f log.txt
less file.txt
wc -l file.txt
grep 'error' log.txt
grep -i 'error' log.txt
grep -n 'error' log.txt
grep -r 'TODO' .
find . -name '*.py'
find . -name '*.py' -size +10k
grep '2026' data.txt | wc -l
history | grep git
```

管道把标准输出交给下一命令。less 用 q 退出、/ 搜索，tail -f 可用 Ctrl+C 停止。

```
python3 hello.py
python3 -c 'print(2**10)'
python3 -m pip --version
which python3
man ls
```

`python3 -m pip` 使用该解释器对应的 pip。正式测试框架在 L11 介绍，安装后可用 `python3 -m pytest`。以下原有命令只供后续查阅：进程在第九周，权限在第十一周展开。

```
ps aux
ps aux | grep python
top
df -h
echo $PATH
chmod +x script.sh
```

grep 管道可能列出 grep 自己，不用它作精确进程数。Ctrl+C 中断前台程序。

计分与提交清单

本周原始满分：A 自动 20 + 人工 4 = 24；B 自动 8 + 人工 4 = 12；C 登记完成。助教汇总为：

本周小作业分 = $60 \times A\text{得分}/24 + 40 \times B\text{得分}/12$

分母包含对应人工记录，人工分未评时不能把自动得分直接当完整周成绩。增加题库原始分不改变学期考核比例。

正式发布后在 HW3 第八周分别使用以下入口：

提交内容	题目 ID
A 五个函数	w08-second-largest、w08-title-case、w08-has-duplicate、w08-count-chars、w08-max-gap
A 四份审查记录	w08-a-review-notes
B 修正版代码	w08-b-is-valid-ipv4
B 四步记录与全实验声明	w08-b-ipv4-notes
C 三项工具记录	w08-c-tools-notes

- ☐ 五道 A 题按规格核对，输入不变；各代码说明据实填写 AI 声明
- ☐ A 文字卡四问有反例、文档或分析与复查依据
- ☐ B 保留八个独立测试、原候选、修正与回归；代码和说明分别填写
- ☐ C 记录真实诊断、调用帧和六步命令；截图是否完成据实说明
- ☐ 本地文件已保存；发布前未误交旧题，发布后逐项确认对应卡片
- ☐ 人工作答显示待评，不当作已取得人工分
- ☐ 个人作业 1 按 Level 0 独立完成，截止见其提交页