

类的设计与实现

保留矩形、栈、词计数、温度、成绩簿重构和类的选择六项任务。先约定接口，再实现与核对；检查实例隔离、无效更新和返回值，提交 A/B/C 学习记录。

截止：小作业 3 · 11 月 17 日 周二 23:59

本次目标与提交入口

使用 [L09](#) 的初始化、实例方法、共享引用与接口检查方法，完成六项原任务。分为 A 基础类、B 成绩簿重构、C 设计判断与调试记录。

五道代码题与三张文字卡使用 [HW3 提交页](#)。先在本机保存、运行和核对，再分别提交对应入口；保存文件不等于已经在线提交。HW3 覆盖第 7–9 周，截止由统一校历计算，不另设“本周日”覆盖整份作业截止。

段	代码入口	文字入口	原始分
A	矩形、栈、词计数、温度，各 4 分	A 设计与验证记录，人工 8 分	24
B	Gradebook，自动 8 分	B 重构与回归记录，人工 4 分	12
C	不提交代码	C 选择、调试与操作记录，0 分	0

A/B 合计 36 原始分。自动测试与人工文字分分别处理，文字分不会因为自动测试通过而自动给出。C 仍需完成，但不增加分数。本周原始分汇入 HW3，不改变学期考核比例。

第 9 周：Level 1

可以使用 AI 辅助，须先自己明确接口、输入域、失败行为和验证方法，再审查候选实现。每道代码题的提交说明写简短使用声明；A/B 文字卡另保存完整记录。没有使用 AI 也明确写“未使用”。HW3 跨周顶栏按最早覆盖周显示较严格标签，每周实验要求仍按对应题面；第 7 周与个人作业一的独立实现要求不因此放宽。

本实验的类是独立程序接口，不能要求评测先运行某页课件或使用另一道题的全局变量。不要用 `input()` 读取题目测试数据，不写交互主程序。

所有公开方法写类型提示，接收者按惯例写 `self`，不需要额外为 `self` 写类型。签名描述接口，不替代运行时检查；评测以约定行为为主，文字记录说明设计与证据。

A 段 · 基础类与实例隔离

9-1 矩形 Rectangle

实现以下接口，宽和高保存在公开属性 `width`、`height`：

```
class Rectangle:
    def __init__(self, width: float, height: float) -> None: ...
    def area(self) -> float: ...
    def perimeter(self) -> float: ...
    def is_square(self) -> bool: ...
    def scale(self, factor: float) -> None: ...
```

规则：

- 宽、高及缩放因子为有限正数。零、负数、NaN 或正负无穷均不合法；构造或缩放抛 `ValueError`。
- `area()`、`perimeter()` 不修改尺寸；`is_square()` 判断两个保存尺寸是否相等，本题不要求模糊几何容差。
- `scale(factor)` 原地同时更新两个尺寸，返回 `None`，不返回新矩形。

- 更新后的两个尺寸也必须为有限正数；若浮点运算得到无穷或零，抛 `ValueError`，保持旧宽、高不变。
- 常规测试使用可表示面积和周长的数值。实数参数通过整数或浮点数传入，不测试其他对象类型。

```
r = Rectangle(3, 4)
r.area()           → 12
r.perimeter()      → 14
r.is_square()      → False
r.scale(2)         → None
r.width, r.height  → 6, 8
r.area()           → 48
```

核对两个实例、正方形、无效初值、缩放成功和失败后的完整尺寸状态。浮点结果使用合理误差比较，不把浮点计算误差误判为接口错误。

A 文字卡解释本题原地约定，并比较返回新对象的另一种接口设计；不能一边选择新对象，一边提交与本题 `-> None` 和原地要求不符的实现。

代码入口： `w09-rectangle`。

9-2 栈 Stack

实现后进先出栈，只保存本实例的数据：

```
class Stack:
    def __init__(self) -> None: ...
    def push(self, item: int) -> None: ...
    def pop(self) -> int: ...
    def peek(self) -> int: ...
    def is_empty(self) -> bool: ...
    def size(self) -> int: ...
```

- `push` 在栈顶加入整数，返回 `None`。
- `pop` 删除并返回栈顶； `peek` 返回栈顶但不删除。

- 空栈 `pop` 与 `peek` 统一抛 `IndexError`，操作后仍为空。不返回 `None`，也不要求评测先替实现检查。
- 初始为空、大小 0；两个栈的 `push/pop` 互不影响。

```
s.push(7); s.push(9)
s.peek()    → 9
s.size()    → 2
s.pop()     → 9
s.pop()     → 7
s.is_empty() → True
s.pop()     → IndexError
```

A 文字卡比较返回 `None`、抛异常、调用前检查三种真实接口选择，但本代码入口采用固定异常契约。理由与实现应一致。

代码入口：`w09-stack`。

9-3 词计数 WordCounter

```
class WordCounter:
    def __init__(self) -> None: ...
    def add(self, word: str) -> None: ...
    def get(self, word: str) -> int: ...
    def total(self) -> int: ...
    def most_common(self, n: int) -> list[tuple[str, int]]: ...
```

- `add` 将给定字符串作为一个完整键计数，不拆词、不去标点、不改大小写；空字符串也是可计数键。
- 未见词 `get` 为 0；`total` 为所有词的次数之和，不是不同词数量。
- `most_common(n)` 返回至多 `n` 个（词，次数），次数降序，并列按 Python 字符串顺序升序。`n=0` 返回空列表，`n` 大于不同词数量时返回全部。
- `n` 为非负整数；负数抛 `ValueError`。本题不测试非整数 `n`。
- 查询与排序不修改计数；两个对象各有自己的计数字典。

- 若使用排序键，采用已学的具名函数，不必使用后面课程才专门讲的 lambda。

```
依次 add("pear"), add("apple"), add("pear"), add("banana")
get("pear")           → 2
get("missing")        → 0
total()               → 4
most_common(3)        → [("pear", 2), ("apple", 1), ("banana", 1)]
```

A 文字卡记录两实例隔离实验，分别给出操作、预期值与实际值，不只写“测试通过”。

代码入口： `w09-word-counter`。

9-4 温度 Temperature

内部保存摄氏度，使用单下划线字段 `self._celsius`：

```
class Temperature:
    def __init__(self, celsius: float) -> None: ...
    def celsius(self) -> float: ...
    def fahrenheit(self) -> float: ...
    def kelvin(self) -> float: ...
    def set_celsius(self, value: float) -> bool: ...
```

- 有效数值为有限实数，且不低于 -273.15°C ；输入通过整数或浮点数提供，不测试其他对象类型。
- 构造时初值无效，抛 `ValueError`。
- setter 无效时返回 **False**，不覆盖原值；有效时更新并返回 **True**。
- 三种查询不修改状态。华氏度为 `c * 9 / 5 + 32`，Kelvin 为 `c + 273.15`。
- 使用布尔结果报告 setter 成败，不在失败时把非法值存入字段再修正。

```
t = Temperature(25)
t.celsius()      → 25
t.fahrenheit()   → 77
t.kelvin()       → 298.15
t.set_celsius(-300) → False
t.celsius()      → 25
t.set_celsius(100) → True
```

核对绝对零度边界、略低于边界、NaN/无穷、拒绝后的三个查询及两个实例。A 文字卡解释构造与更新都要校验，也说明单下划线不提供强制访问保护。

代码入口： `w09-temperature`。

A 设计与验证记录

文字入口： `w09-a-design-notes`。四项各人工 2 分：

1. **原地与新对象**：解释 Rectangle 当前约定及另一方案的返回值、别名影响。
2. **空栈协议**：比较 None、异常、前置检查，解释本题选 IndexError 的一致性。
3. **实例隔离**：给出两个 WordCounter 的操作序列、预期/实际值与结论。
4. **有效状态**：给出 Temperature 非法构造和非法 setter 的证据，说明前后状态与下划线的作用。

另填 AI 使用声明，零分字段，不计入上述 8 分。若使用，写工具/版本、请求目标、采用或修改内容、验证依据；未使用写明未使用。不将 AI 输出或自己的口头结论当作测试证据。

B 段 · 重构与回归

9-5 成绩簿 Gradebook

保留原字典程序的任务：保存姓名、科目与成绩，查询平均分和最高均分学生。先审查下面候选，再设计类接口。

```
def add_score(book, name, subject, score):
    if name not in book:
        book[name] = {}
    book[name][subject] = score

def average(book, name):
    scores = book[name].values()
    return sum(scores) / len(scores)

def top_student(book):
    names = list(book)
    best = names[0]
    for name in names[1:]:
        if average(book, name) > average(book, best):
            best = name
    return best
```

这只是待审查片段，不是完整运行入口。新版本不靠 lambda 或生成器表达式完成重构。

```
class Gradebook:
    def __init__(self) -> None: ...
    def add(self, name: str, subject: str, score: float) -> None: ...
    def average(self, name: str) -> float: ...
    def top_student(self) -> str | None: ...
```

契约：

- 姓名、科目为非空字符串，按原字符串保存，不自动裁剪或改大小写；测试不会提供其他类型。
- 成绩为有限实数，0–100，含两端；无效成绩、空姓名或空科目抛 ValueError，**整个成绩簿保持原状态**，不留下新学生或空科目。
- 同姓名、同科目再次 add 覆盖旧分数；不同科目累积保存。add 返回 None。
- `average(name)` 返回该学生当前科目的算术平均；不存在的学生抛 KeyError，不创建学生。

- `top_student()` 空簿返回 **None**；否则返回最高平均分姓名，并列按 Python 字符串顺序升序选最小者，不能依赖加入顺序。
- 两个簿独立；查询不改变数据。

```
add("Alex", "math", 87)
add("Alex", "physics", 92)
add("Blake", "math", 65)
average("Alex")    → 89.5
top_student()      → "Alex"
空簿 top_student() → None
average("missing") → KeyError
```

原候选至少缺少空簿行为约定与缺学生错误说明，最高均分并列还依赖先加入者。新接口明确允许 `KeyError`，而不是把未说明的失败称为修好。回归既有正常例子，也测空簿、缺学生、并列、覆盖和拒绝后状态。

代码入口： `w09-gradebook` 。

B 重构与回归记录

文字入口： `w09-b-refactor-notes` 。

- **接口与修正，人工 2 分**：逐项说明候选原行为、新契约及修改理由，至少包含缺学生和空簿。
- **回归证据，人工 2 分**：正常例子、并列、覆盖、无效 `add` 前后状态，记录输入、预期、实际与结论。
- **AI 使用声明，0 分**：与实际代码提交对应；未使用明确写明。

说明文字不写进别道题的代码文件，也不把五道完整参考实现粘贴到文字卡。

C 段 · 设计判断与调试记录

9-6 类、函数或记录

保留四项原需求。为每项给出选择和依据，不按“这周学类”统一选类：

1. 将摄氏温度转换为华氏度。
2. 保存一副牌，支持洗牌、发牌和查询剩余张数。
3. 判断字符串是否为回文。
4. 保存对话历史，并据此组织后续回复的程序。

按状态是否跨调用保留、相关操作、生命周期与接口边界论证。无需连接模型服务或提交聊天数据。一个实例、一个方法或只读数据都不是自动排除类的规则。

调试器中的两个对象

使用本机自写 Point 或已经学过的类，不要求粘贴课件中的完整程序：

1. 创建两个实例，在实例方法中设断点。
2. 分别从两个对象调用，观察 self 与相关字段。
3. 原地更新一个实例，记录另一个是否保持。
4. 如有别名，说明与新构造实例的区别。

记录断点位置、调用顺序、self 选择与实际字段。不能启动调试器时，记录原因并用 print 给出同样的前后观察，不虚构截图。复习 L08 已教的调试动作，不新增工具安装要求。

文字入口：`w09-c-design-tools`。包含设计判断、调试记录、声明与操作复查三个零分字段；不提交可执行源码。零分表示不增加分数，不表示可以忽略任务。

提交前检查

- ☐ 五道代码分别提交正确入口，方法名和返回值遵守本题，而非另一课件例子。
- ☐ 必需类型提示完整，self 按惯例不额外注解。

- ☐ 两实例隔离，别名与共享输入的观察有实际记录。
- ☐ Rectangle 失败缩放保留两尺寸，Stack 空 pop/peek 都抛 IndexError。
- ☐ WordCounter 排序并列与 n=0、空状态正确，查询不改计数。
- ☐ Temperature 的构造、setter、有限性与绝对零度都核对。
- ☐ Gradebook 空簿、缺学生、并列、覆盖、拒绝原状态均核对。
- ☐ A/B 文字记录和每道代码说明包含真实使用声明；C 判断与调试记录已完成。

题面、骨架、测试与文字评分使用同一接口。自动分不代替人工设计评价，不改变 HW3 的课程权重。