M7 · 第 18 讲

用自然语言写第一个程序

把需求写成输入、处理、输出

阅读约 30 分钟

练习约 30 分钟

产出：一个能跑的小脚本

本节目录

开场：想做个工具，但不会写代码

本课目标

一、先写三行需求

二、脚本还是网页

三、给 AI 的提示词

四、一个最小例子

五、迭代：一次一个功能

六、安全底线

七、30 分钟练习

八、检查清单

九、延伸阅读

开场：想做个工具，但不会写代码

你每周要手动改 200 个文件名，或者把三张表格合成一张。你搜过脚本，复制过来一堆报错，最后放弃。

现在不需要先学语法。你只要能把“输入—处理—输出”说清楚，AI 就能写出第一版。真正的门槛是**定义问题**，不是敲代码。

本课目标

- 把一个需求写成“输入—处理—输出”；
- 判断做脚本还是做网页；
- 用一段可执行的提示词拿到第一版；
- 跑通、改一处、再跑通。

一、先写三行需求

不要写“帮我做个工具”，写三行：

> 输入：一个文件夹里的 200 张图片 处理：按拍摄日期重命名，格式 YYYY-MM-DD_序号 输出：重命名后的文件 + 一份对照表

三行不清，代码一定返工。

二、脚本还是网页

形态	适合	不适合
脚本	批处理、定时任务、本地文件	给别人用
网页	表单、展示、多人访问	读写本地文件
表格插件	在现有表格里加功能	复杂交互

第一次做，优先脚本：依赖少、能看到每一步、出错好改。

三、给 AI 的提示词

一段够用的模板：

> 用 Python 写一个脚本，环境是 macOS，Python 3.11。 输入：目录 /Users/me/photos 下的 jpg/png。 处理：读取 EXIF 拍摄时间；没有则用文件修改时间；按日期分组，重命名为 日期_序号。 输出：重命名文件，并生成 rename_log.csv（旧名、新名、时间）。 要求：先打印预览，加 --apply 参数才真正改名；跳过已符合命名规则的文件；异常文件记录到日志，不中断。 只给一个完整文件，附运行命令。

关键四件事：**环境、输入、规则、安全开关**。

四、一个最小例子

比如统计 CSV：

```
import csv
from collections import Counter

with open("orders.csv", newline="", encoding="utf-8") as f:
    rows = list(csv.DictReader(f))

counts = Counter(r["city"] for r in rows)
for city, n in counts.most_common(10):
    print(city, n)
```

运行：

```
python3 count.py
```

先让它跑通，再谈加功能。

五、迭代：一次一个功能

第一版只做核心路径。之后每轮只加一件事：

1. 先能读文件；
2. 再加规则；
3. 再加日志；
4. 最后加预览/确认。

每加一步都跑一次。一次加三个功能，出错时你不知道是哪一处。

六、安全底线

- 批量改文件前，先复制一份或只做预览；
- 删除操作必须二次确认；
- 密钥不要写进代码，用环境变量；
- 处理他人数据前，先脱敏。

七、30 分钟练习

1. 写一个三行需求（5 分钟）；
2. 让 AI 生成脚本（10 分钟）；
3. 用 3 个样本文件跑通（10 分钟）；
4. 加一个“预览模式”再跑（5 分钟）。

八、检查清单

- [ ] 输入、处理、输出三行写清；
- [ ] 说明了运行环境；
- [ ] 有预览/确认，不会一上来就改数据；
- [ ] 异常不会让整个任务中断；
- [ ] 代码能在一个文件里跑起来。

九、延伸阅读

- 知识库 2.0 AI 编程
- 知识库 3.2 工具：辅助学习工具和资源
- 知识库 3.1 案例：AI 产品案例和投稿

> 下一讲：读懂报错与迭代修复。你会把“红字”变成下一步动作。
