Lab 1 · Part 1 计算基础与 Python 入门 AI Level 0

环境搭建与第一个程序

在自己的电脑上装好 Python 和 VS Code，学会用终端运行 .py 文件，走通一次提交。B 段用十行代码演示混沌——初值差百万分之一，二十五轮之后面目全非。

截止：小作业 1 · 10 月 10 日 周六 23:59

本次目标

- 在自己的电脑上装好 Python 3.13 和 VS Code
- 能用两种方式运行程序：VS Code 里点运行、终端里敲命令
- 走通一次完整的提交流程
- 亲眼看见「初值的微小差别会被放大」这件事

AI 政策提醒

本周处于 **Level 0**：不得使用 AI 生成或补全代码。安装过程中遇到问题， 可以问 AI「这个报错是什么意思」，但作业代码要自己写。

这次实验课怎么安排

题目分三段，**同时开放**，不必按顺序做完：

- **A 段**（必做）——覆盖本周知识点，平台自动判
- **B 段**（深入）——一道大题，分几问递进
- **C 段**（基本功）——命令行、git、远程这些，每周一小段，**期末要考**

装环境会吃掉今天大半时间，所以本周三段都轻。以后每周都是这个结构。

装环境是会出问题的

七十多号人、各种型号的电脑和系统版本，一定会有人卡住。这很正常，不是你的问题。 助教全程在场，卡住了就举手。**不要一个人跟它耗一小时。**

一、安装 Python

课程网站上放了一份 **Python 3.13.15** 的官方安装包。校园网连 python.org 时快时慢，第一周九十分钟耗不起，**直接从这里下载**：

你的系统	下载	大小
Windows（绝大多数）	python-3.13.15-amd64.exe	28 MB
Windows on ARM（骁龙本）	python-3.13.15-arm64.exe	27 MB
macOS（Intel 与 Apple Silicon 通用）	python-3.13.15-macos11.pkg	69 MB

这三个文件和 python.org/downloads 上的完全一致， 只是存了一份在校内。

不要用 Microsoft Store 里的那个 Python

应用商店版本的路径和权限处理和官方安装包不一样，装库时经常出莫名其妙的问题， 往年每年都有同学卡在这上面。**用上面的安装包。**

同样地，Windows 上**装在 C 盘默认位置**，不要改到 D 盘或带中文的路径。

Windows

运行安装程序，**在第一个界面就勾选** `Add python.exe to PATH`，然后点 Install Now。

这一步漏了会很麻烦

`Add to PATH` 那个复选框在窗口底部，默认不勾选，很容易看漏。 漏了的话终端里敲 `python` 会提示「不是内部或外部命令」。 补救办法：重新运行一次安装程序，选 Modify，把它勾上。

macOS

下载 `.pkg` 安装包，一路下一步。

系统自带的那个 `python3` 版本通常比较旧（macOS 带的是给系统用的，不是给你用的）， 装完之后以官网这个为准。

Linux 上不用装

Ubuntu 之类的发行版自带 `python3`， `python3 --version` 看一眼版本够不够新即可。 真要装新版本是 `sudo apt install python3.13`。

但不要随手改掉系统默认的 python3 版本 —— Linux 上很多系统工具本身是 Python 写的，换掉解释器会引发一连串莫名其妙的故障。

验证

打开终端（Windows 用 PowerShell 或 Terminal，macOS 用「终端」），输入：

```
python --version
```

应该输出 `Python 3.13.x`。如果提示找不到命令，在 macOS 和 Linux 上试试 `python3 --version`； 在 Windows 上说明 PATH 没配好，回去重装并勾选那个选项。

`python` 和 `python3` 为什么是两个命令

Python 2 和 Python 3 互不兼容，共存过很多年。那些年里 `python` 指 2.x、 `python3` 指 3.x，是各系统的通行约定。Python 2 已于 2020 年停止维护， 但这个约定留了下来：**macOS 和 Linux 上通常只有 `python3`**， Windows 官方安装包两个名字都装。本课程的讲义一律写 `python`， 你自己的机器上是哪一个，验证这一步就知道了。

二、安装 VS Code

从 code.visualstudio.com 下载安装。装完之后要装一个扩展：

1. 点左侧栏的扩展图标（四个方块那个），或按 `Ctrl+Shift+X`（macOS 是 `Cmd+Shift+X`）
2. 搜索 `Python`
3. 装微软官方那个（作者是 Microsoft，下载量最高的）

这个扩展提供语法高亮、错误提示、运行按钮和调试器。

三、写第一个程序

1. 新建一个文件夹专门放这门课的代码。**路径里不要有中文和空格**——有些工具处理起来会出问题。比如 `D:\cs1602` 或者 `~/cs1602`。
2. 在 VS Code 里 File → Open Folder，打开这个文件夹。
3. 新建文件 `hello.py`，写进去：

```
print("Hello world")
```

4. 点右上角的 ▶ 运行按钮。下方会弹出一个终端，显示 `Hello world`。

也用终端运行一次

VS Code 里的运行按钮很方便，但你应该知道它背后干了什么。

按 `Ctrl+`` （反引号，键盘左上角 Esc 下面那个键）打开终端，输入：

```
python hello.py
```

同样的输出。**这才是运行 Python 程序的本来面目**——运行按钮只是替你敲了这行命令。

四、五个小实验

环境装好之后，在**交互式解释器**里跑一遍下面五件事。终端里直接敲 `python`（或 `python3`）就进去了，提示符是 `>>>`；退出敲 `exit()` 或按 `Ctrl+D`。

这五件事的答案要填到平台的「环境检查与第一次观察」那张答题卡里。**都是一行就能跑出来的，重点不是敲，是看结果和你想的一不一样。**

1. **两种引号**。分别跑 `print("上海交通大学")` 和 `print('上海交通大学')`，比较输出。
2. **Python 的信条**。跑 `import this`，读一遍。这是 Tim Peters 写的 19 条，Python 社区的审美纲领，往后这门课会不断回到它。
3. **一个很大的数**。算 `2 ** 100`。
4. **两种除法**。算 `10 / 3` 和 `10 // 3`。
5. **一次加法**。算 `0.1 + 0.2`。

> **第 5 个是这门课的一条暗线**
>
> `0.1 + 0.2` 不等于 `0.3`。这不是 Python 的 bug，也不是你的电脑坏了 —— 所有用 IEEE 754 浮点数的语言（C、Java、JavaScript、Excel）都是这个结果。**L2 会讲清楚为什么**，B 段那道题也是同一件事的另一面。

想玩玩的（不计分）

新建 `star.py`，跑一下：

```
import turtle

tur = turtle.Turtle()
tur.speed(6)
tur.getscreen().bgcolor("black")
tur.color("cyan")
tur.penup()
tur.goto(-200, 50)
tur.pendown()

def star(t, size):
    if size <= 10:
        return
    for _ in range(5):
        t.forward(size)
        star(t, size / 3)
        t.left(216)

star(tur, 360)
turtle.done()
```

一个五角星，每条边上又长出小五角星，小五角星上再长出更小的 —— 这叫**分形**。 画出这个图形的代码不到二十行，靠的是 `star` 把自己又叫了一遍。 这个写法叫**递归**，第 7 讲专门讲它。

现在不用懂，改改 `360` 和 `size / 3` 这两个数，看图形怎么变就行。

图形窗口卡住不动

`turtle.done()` 会一直等着你关窗口，这时候终端是没有反应的，不是死机。 关掉那个画图窗口，终端就回来了。 macOS 上如果报 `ModuleNotFoundError: No module named 'tkinter'`， 说明装的 Python 缺图形库 —— 这个实验跳过就行，不影响别的。

五、提交

打开 课程评测平台，**用你的学号注册**（不需要口令，学号在选课名单里就能注册），然后进「小作业 1」。

那一页上有两种入口：

入口	交什么	怎么算分
代码题	一个函数	平台当场跑测试，立刻出分
答题卡	选择题 + 问答题	选择题当场判；问答题写进评测报告，助教看

A 段的编程题是代码题，本周的五个小实验和 B、C 两段的问题都是答题卡 —— **你在网页上直接答，不用另外交文档**。写完的答案会一并写进评测报告， 学期末下载报告交到 Canvas 就行。

从这周开始养成习惯：每次实验课结束前提交一次，哪怕没做完。平台记录你的最好成绩， 答题卡也可以反复改了再交。

A 段 · 必做

1-1 自我介绍

写一个程序，用 `print()` 输出三行：你的姓名、学号、以及一句你想说的话。

```
张三
526030910001
希望这学期能写出让自己满意的代码
```

1-2 进制转换

用 `bin()`、`oct()`、`hex()` 输出你的学号在二进制、八进制、十六进制下的样子。

> **提示**
>
> 学号是一串数字，但注意——如果你直接写 `bin("526030910001")` 会报错， 因为引号里的是字符串不是数。想想应该怎么写。

1-3 算一算

用 `print()` 输出下面几个式子的结果，**先自己猜一遍再运行**：

```
7 / 2
7 // 2
7 % 2
2 ** 10
```

先自己写下来，再运行验证。**猜错的那个填到答题卡「A 段 · 算一算与读报错」里**， 说说原来以为是多少。这四个运算符 L2 会正式讲。

1-4 故意写错

分别制造下面三个错误，把完整的报错信息复制到答题卡里：

1. 把 `print` 拼成 `pirnt`
2. 用中文全角括号 `print（"hello"）`
3. 字符串只写一半的引号 `print("hello)`

然后回答：这三个报错的**第一行**分别是什么？它们指向的行号对不对？

> **这道题不是凑数的**
>
> 接下来十五周你会看到成百上千次报错。**能不能快速读懂报错，直接决定你写代码的速度。** 从今天起，看到红字第一反应应该是读它，不是慌。

B 段 · 深入：十行代码里的混沌

这一段不要求你懂语法——照着敲进去、运行、观察就行。它要给你看的是 **计算这件事本身的一个性质**，和 Python 无关。

新建 `chaos.py`：

```
x = float(input("输入一个 0 到 1 之间的小数："))

for i in range(25):
    x = 3.9 * x * (1 - x)
    print(f"{i + 1:3}  {x:.17f}")
```

注意第 4、5 行开头要多缩进四个空格——这是 Python 表示「这两行属于循环」的方式，L4 会正式讲。

B-1 跑一遍

输入 `0.25`，运行，把最后一行记下来 —— 下一问要拿它对照。

这个式子叫 Logistic 映射，是生态学里用来描述种群数量的模型：`x` 是当前种群占环境容量的比例，每一轮算出下一代。系数 3.9 表示繁殖率。

B-2 把初值改动百万分之一

再跑一次，这次输入 `0.250001`。

两次的初值只差 **0.000001**。请对照两次的输出回答：

1. 第 10 轮时，两个结果差多少？
2. 第 25 轮时，两个结果差多少？
3. 从第几轮开始，你已经看不出这两个序列有什么关系了？

> **这就是「蝴蝶效应」**
>
> 1961 年，气象学家 Lorenz 在重跑一次天气模拟时，为了省事把中间结果 从 0.506127 抄成了 0.506——他以为这点差别无所谓。结果两次模拟的天气 完全不同。这个意外催生了混沌理论。
>
> 它的现实后果是：**长期天气预报在原理上就是做不到的**， 不是因为计算机不够快，而是因为你永远无法把初始条件测得足够准。

B-3 同一个初值，两种写法

把循环里那一行换成下面这个，**初值仍然输入 0.25**：

```
x = 3.9 * x - 3.9 * x * x
```

`3.9 * x * (1 - x)` 和 `3.9 * x - 3.9 * x * x` 在数学上**完全等价**，你可以自己展开验证。

请回答：

1. 两种写法跑出来的第 25 轮结果，**一样吗**？
2. 从第几轮开始出现差别？
3. 你觉得这是为什么？（猜一猜就行，答案在 **L2 第四段**）

> **B-3 比 B-2 更值得想**
>
> B-2 里两次输入不同，结果不同，还算讲得通。 B-3 里输入完全相同、公式在数学上完全相同，结果却不同—— **这说明计算机算的并不是你写的那个数学式子。** 这门课第二讲会解释它。

C 段 · 基本功：第一次打开终端

> **这一段为什么存在**
>
> 这门课叫**计算导论**，不只是 Python。接下来十五周，每次实验课都有 一小段计算机基本功：命令行、版本控制、远程登录、权限、脚本。 每周十五分钟，攒到学期末就是一套完整的开发者基本技能。**期末考试会考。**

将来你会在没有图形界面的服务器上跑程序，那时候只有终端。而且很多工具 （装库、版本管理、跑测试）本来就是命令行的。

四个命令

命令	作用	记法
`pwd`	我现在在哪个目录	**p**rint **w**orking **d**irectory
`ls`	列出当前目录里有什么	**l**i**s**t
`cd 目录名`	进入某个目录	**c**hange **d**irectory
`cd ..`	回到上一级目录	`..` 表示上一级

Windows 的 PowerShell 这四个都能用（`ls` 和 `pwd` 是别名）。

> **按 Tab 键**
>
> 敲 `cd cs` 然后按 Tab，终端会帮你补全成 `cd cs1602`。 **这是用命令行最重要的一个习惯**——省时间，而且不会拼错。

C-1 走一遍

在终端里依次完成，把**每一步的命令和输出**从终端复制出来，填进平台上的答题卡（不必截图）：

1. `pwd` 看看自己在哪
2. `cd` 到你放这门课代码的文件夹
3. `ls` 确认 `hello.py` 和 `chaos.py` 都在
4. `python hello.py` 跑一次
5. `cd ..` 退到上一级，再 `pwd` 确认位置变了

> **「找不到文件」几乎总是路径问题**
>
> `python hello.py` 报 `can't open file`，原因基本都是你当前不在 `hello.py` 所在的目录。先 `pwd` 看自己在哪，再 `ls` 看这个目录里有没有那个文件。

检查清单

交之前对一遍：

- [ ] `python --version` 输出 3.13.x
- [ ] VS Code 装好了 Python 扩展
- [ ] 能用运行按钮跑通 `hello.py`
- [ ] 能用 `python hello.py` 跑通同一个文件
- [ ] 五个小实验跑过了，答题卡「环境检查与第一次观察」交了
- [ ] A 段四道题都提交了
- [ ] 答题卡「B 段 · 十行代码里的混沌」交了
- [ ] 答题卡「C 段 · 第一次打开终端」交了

- ☐ 课程群加了

装不上怎么办

症状	多半是	怎么办
`python` 不是内部或外部命令	Windows 上 PATH 没配	重跑安装程序 → Modify → 勾上 Add to PATH
`zsh: command not found: python`	macOS 上命令名不同	试 `python3`；或检查是否装到了别的位置
VS Code 运行按钮是灰的	没装 Python 扩展	装微软官方的 Python 扩展
VS Code 找不到解释器	没选解释器	`Ctrl+Shift+P` → 输入 `Python: Select Interpreter` → 选 3.13
运行时报 `can't open file`	终端当前目录不对	`pwd` 看位置，`cd` 到文件所在目录
`chaos.py` 报 `IndentationError`	循环里那两行没缩进	那两行前面各加四个空格

还是不行——举手叫助教，或者发到课程群里。**发问的时候把完整报错截图贴上，说清你的系统版本。**