

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Introduction to Computation (CS1602) · Lecture 8

AI Programming and Linux

Instructor: Tao Huang

Part 3: Problem Solving with AI · Fall 2026

What Is in Our Course Folder?

As course files grow, **count them by file type**.

Course folder 课程文件夹

```
L01.PDF        L02.pdf
README         .env
notes.txt      data.csv
lab1.py        diagram.PNG
```

Target report 预期报告

```
(none): 2
csv: 1
pdf: 2
png: 1
py: 1
txt: 1
```

PDF 与 pdf 合并统计；原文件名保持不变。

Start with the Initial Function

We will **read and review** the first version.

`l08-classroom-demo/`

`candidates/first.py`	初版
`inventory.py`	参考
`check_inventory.py`	检查
`sample-files/`	样例

1. Download & unzip

示例项目 ZIP

2. Open the initial function

`candidates/first.py`

It Runs, but Does It Meet the Rules?

The initial function counts **PDF and pdf separately**.

Input: `["L01.PDF", "L02.pdf", "README"]`

Initial function 初版实跑

```
{'PDF': 1,
 'pdf': 1,
 '(none)': 1}
```

Expected 按规则应得到

```
{'pdf': 2, '(none)': 1}
```

Review code
Find the mismatch

Ask AI locally
Explain or suggest a fragment

Run checks
Verify the change yourself

> 先审查实际行为，再引入 AI 局部辅助；建议仍需自己验证。

How This Lecture Fits Together

Two parts: **AI + debugging**, then **Linux**.

Part 1 · AI + Debugging

理解模型，提供上下文，获得局部帮助。
自己运行、调试和核对建议。

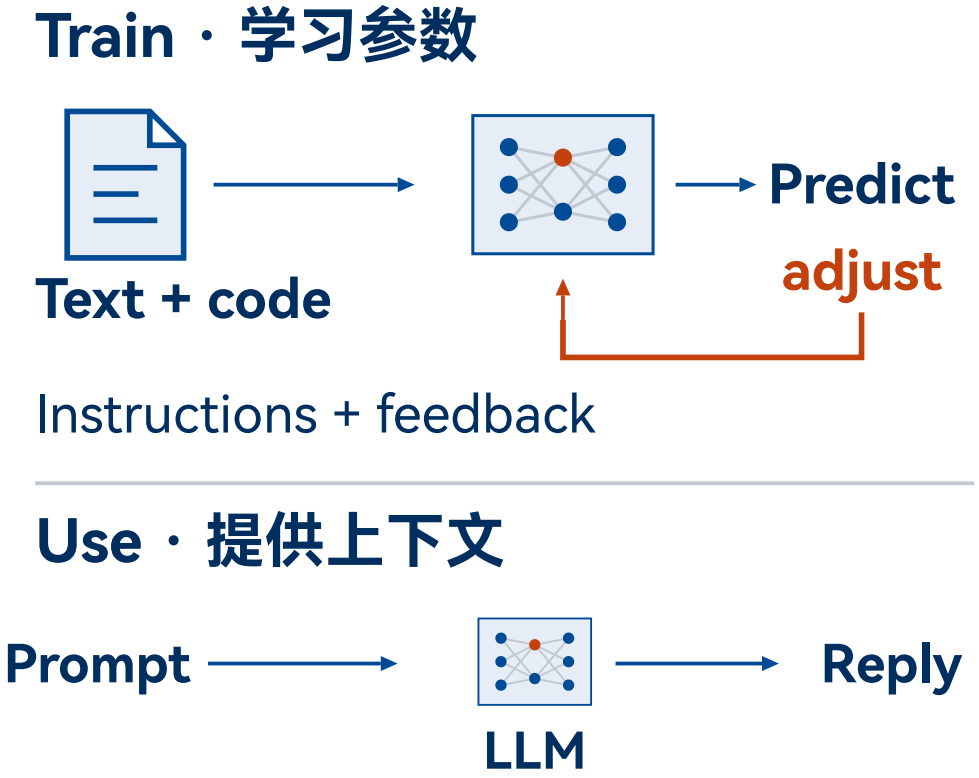
Part 2 · Linux

从历史、内核到发行版与 shell。
用几个小例子练习终端操作。

Why Can an LLM Answer Us?

Large Language Model（LLM，大语言模型）

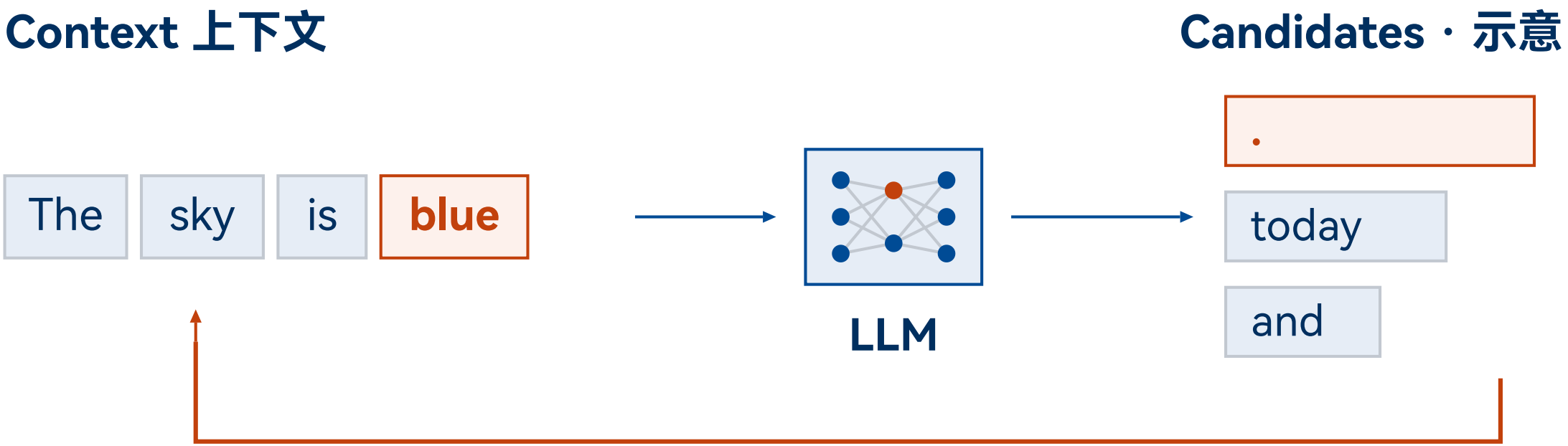




训练调整参数；提问提供上下文。

Text Is Generated One Token at a Time

Read the context → choose a token → append → **repeat**.

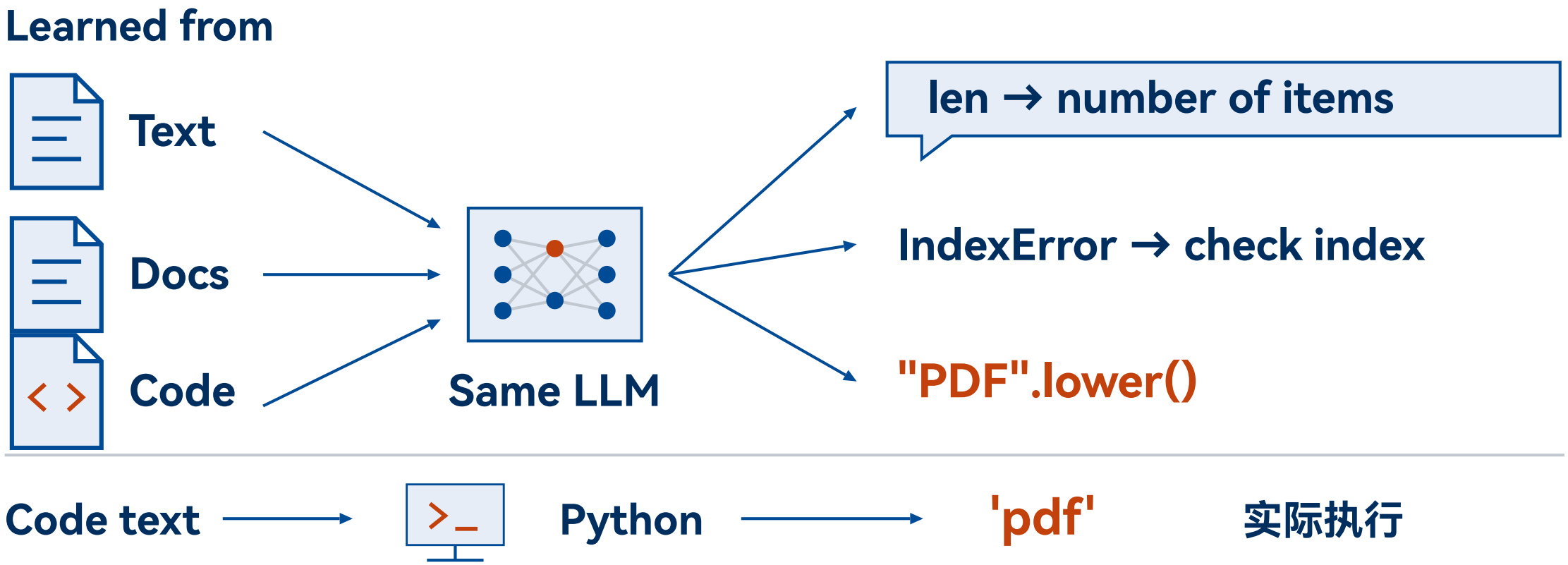


Full context + new token → next step

新片段接回完整上下文，再生成下一片段。

Why Can the Same Model Write Code?

Code is structured text; **generating it does not execute it**.



模型生成代码文本；解释器执行，测试检查行为。

A Fluent Answer Can Still Be Wrong

A plausible suggestion may use **an unavailable method**.

Teaching example 教学例子

```
1  s = "hello"
2  print(s.reverse())
```

- A familiar-looking API may not exist.
- Check the actual object and method.
- Use documentation and execution.

Actual Python error 实际错误

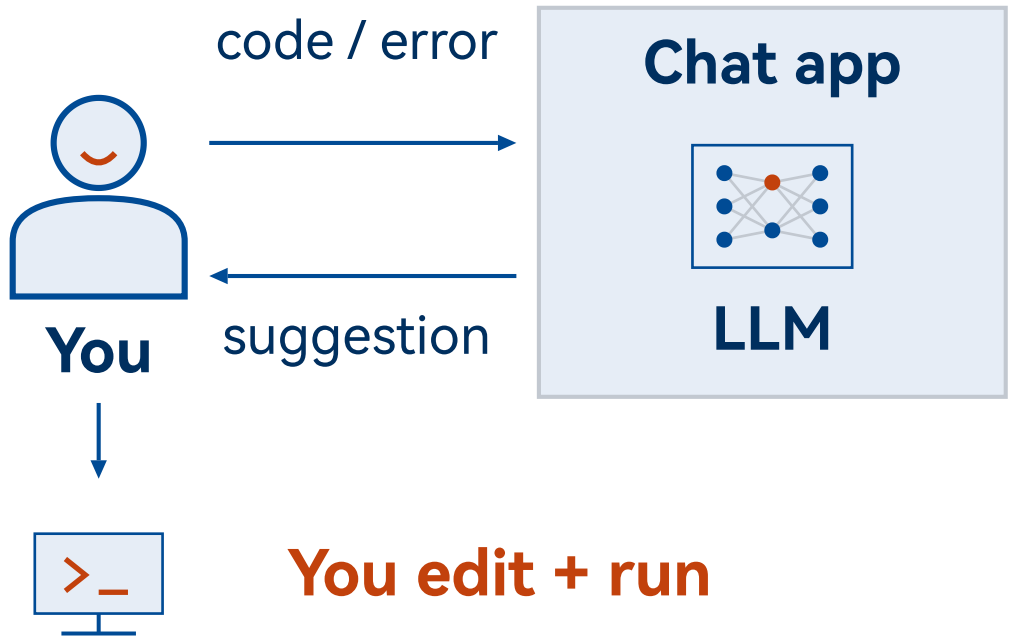
```
AttributeError: 'str' object has no attribute 'reverse'
```

先核对接口和任务，再采纳建议。

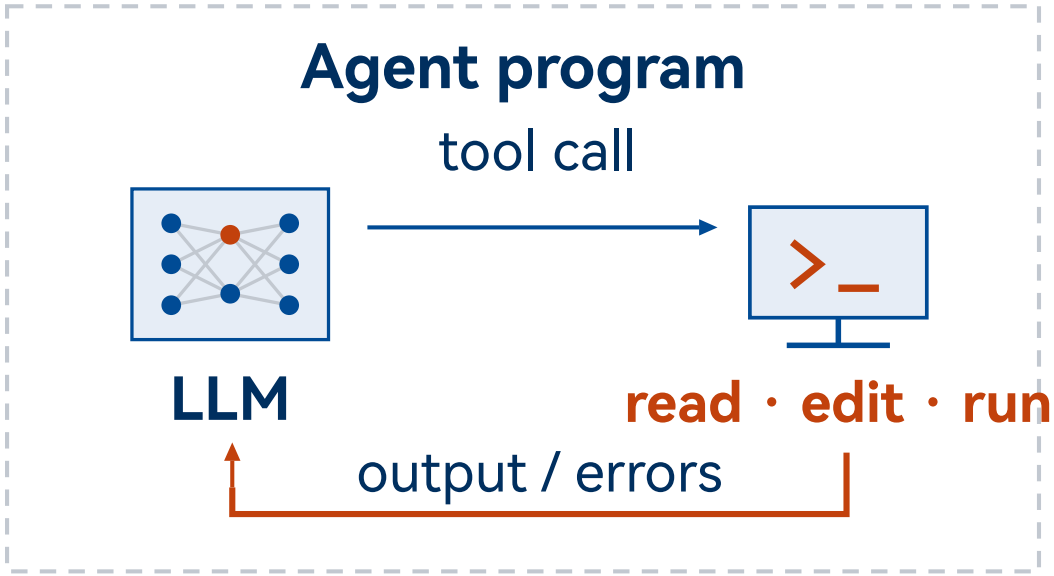
LLM, Chat App and Coding Agent

Who edits files and **runs the code**?

Level 1 · Chat · 本周



Level 2 · Coding agent



You review the changes

本周局部求助，自己修改运行；不让 AI 代写整题。

How to Use Chat AI

Our examples use **DeepSeek Chat**.

Chat AI tools

- ChatGPT
- Claude
- **DeepSeek**
- 豆包
- 千问
- 腾讯元宝
- Kimi
- 智谱 GLM
- MiniMax
- 小米 MiMo



https://chat.deepseek.com/

AI in Practice: Four Ways to Get Help

Use chat at **different points in your own work**.



Understand

读懂代码

Trace state changes.

Add a fragment

生成局部代码

Keep one local change.

Diagnose

分析报错

Use the real traceback.

Review checks

审查检查遗漏

Find a missing case.

先明确卡在哪里，再决定请 AI 帮哪一部分。

Give Evidence, Not “It Doesn’t Work”

Make the **same problem reproducible**.

Missing context 信息不足

```
It does not work.
Please fix it.
```

Useful context 有用信息

```
Python: 3.13
Input: "README"
Expected suffix: "(none)"
Actual: IndexError
Code: parts[1].lower()
Include the full traceback.
Ask: explain + local change.
```

让别人拿到材料后，可以复现同一个问题。

Start from Code and a Clear Contract

Decide **the expected behavior before asking**.

Classroom example 课堂示例

```
names → suffix handling
      → count with a dict
      → print the report
```

Classroom files · 示例项目 ZIP

inventory.py · 完整参考实现

> 任务与预期自己决定，AI 建议再据此检查。

- Merge PDF and pdf.
- Use the final valid suffix.
- README / .env → **(none)**.
- Keep the input names unchanged.

AI in Practice 1: Understand Code

Ask about **the part you do not understand**.

Sent to AI 发出的材料

这段代码我有点没看懂，get 后面的 0 是干嘛的？能拿第一次循环简单说说吗？

Code sent 一起发送的代码

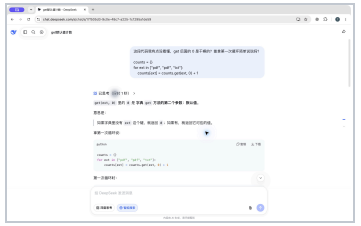
```
counts = {}
for ext in ["pdf", "pdf", "txt"]:
    counts[ext] = counts.get(ext, 0) + 1
```

AI reply 回答节选

> 如果字典里没有 ext 这个键，就返回 0；如果有，就返回它对应的值。

首轮实跑：get / counts

```
0
{'pdf': 1}
```



DeepSeek Chat
点击放大完整对话

> 问清楚一个具体点，再自己验证。

AI in Practice 2: Change One Line

Clarify the context, then choose **one local change**.

Sent to AI 发出的材料

前文要求：PDF / pdf 合并统计，其他部分先不动。

我后面就是用 ext 当字典的键，没有别的判断，文件名里的空格也要保留。代码是这两行：

Follow-up 补充的代码

```
ext = parts[1]
counts[ext] = counts.get(ext, 0) + 1
```

那我只加 lower() 可以吗?

AI reply 回答节选

> 这样 PDF 和 pdf 会合并成同一个键 pdf，同时不会动到文件名中的空格。

> 不要加 .strip()，因为你明确要保留空格。

Local suggestion 局部建议

```
ext = parts[1].lower()
```



DeepSeek Chat
点击放大完整对话

> 补充上下文，审查建议，只采纳符合任务的局部改动。

AI in Practice 2: Run and Check

Compare **actual outputs before and after**.

Before 真实结果

```
{'PDF': 1,
 'pdf': 1,
 '(none)': 1}
```

After 真实结果

```
{'pdf': 2, '(none)': 1}
```

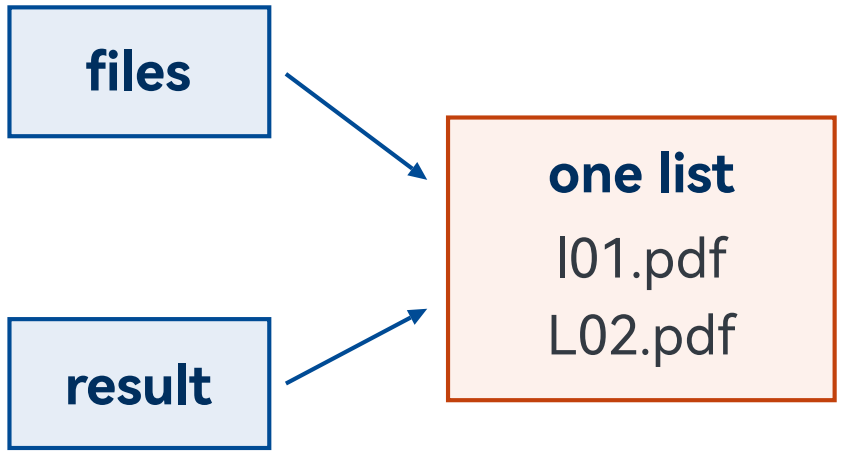
> 相同输入对比结果，再确认其他约定没有改变。

Why Did the Input List Change?

result = names points to the same list.

Input mutation 输入被修改

```
1  def prepare(names):
2      result = names
3      result[0] = result[0].lower()
4      return result
5
6  files = ["L01.PDF", "L02.pdf"]
7  print(prepare(files))
8  print(files)
```



Actual output 实际输出

```
['l01.pdf', 'L02.pdf']
['l01.pdf', 'L02.pdf']
```

同一列表被改动，两个名字都能看见变化。

AI in Practice 3: Read the Traceback

For **README**, splitting produces only one part.

Failing expression 失败位置

```
return parts[1].lower()
```

Full Python traceback 完整回溯

```
Traceback (most recent call last):
  File "debug_suffix.py", line 5, in <module>
    print(suffix("README"))
          ~~~~~~^^^^^^^^^^
  File "debug_suffix.py", line 3, in suffix
    return parts[1].lower()
           ~~~~~^^^
IndexError: list index out of range
```

保留最小输入、相关代码和完整错误。

AI in Practice 3: Diagnose an Error

Send the **code, input, expected result and traceback**.

Sent to AI 发出的材料

这几行一运行就报错了，README 为什么不行呀？它没有后缀，我希望返回 "(none)"。帮我看看这几行怎么改就好。

Code + input 一起发送的代码与输入

```
def suffix(name):
    parts = name.split(".")
    return parts[1].lower()

print(suffix("README"))
```

另附上一页完整回溯:

```
IndexError: list index out of range
```

> 诊断能解释报错后，再检查修改是否符合全部约定。

AI reply 回答节选

> 问题出在 "README".split(".") 返回的是 ["README"]（只有一个元素），所以 parts[1] 越界了。

Local condition 局部判断

```
if len(parts) == 1:
    return "(none)"
```



DeepSeek Chat
点击放大完整对话

AI in Practice 3: Test the Suggested Fix

No exception — does the **return value** meet the rule?

DeepSeek suggestion 第一版局部改法

```python
def suffix(name):
    parts = name.rsplit(".", 1)
    if len(parts) == 1:
        return "(none)"
    return parts[1].lower()
```

Input	Actual	Expected
README	'(none)'	'(none)'
notes.	''	'(none)'

异常消失后，继续核对输入和真实返回值。

AI in Practice 4: Find Missing Tests

Ask for **test ideas** using requirements and existing cases.

Sent to AI 发出的材料

我写了个统计文件后缀的小函数，PDF 和 pdf 要算一类，没有后缀的算 "(none)"，也不能改原来的文件名。

我目前只试过 ["a.PDF", "b.pdf"]，结果是 {"pdf": 2}。

这就算测全了吗？还应该先试哪两种情况呀？

> 本次未发送函数实现，只请 AI 补充测试思路。

AI reply 回答节选

> 所以“先试哪两种”：无后缀 和 隐藏文件/多点文件名。

Reference run 自己实跑参考实现

```
README → (none)
archive.tar.gz → gz
.bashrc → (none)
```



DeepSeek Chat
点击放大完整对话

> 测试思路可作补充；核验具体实现还需代码和运行证据。

AI in Practice: Review the Changes

Keep **the local change and its evidence** together.

Change + version 修改与版本

```
- ext = parts[1]
+ ext = parts[1].lower()

git diff -- inventory.py
git status
```

- Explain why the changed line fits.
- Rerun the independent checks.
- Save the intended working version.
- Disclose the local AI assistance.

看差异、跑检查、保存版本，并据实说明帮助范围。

Give AI Runtime Evidence

Check an AI explanation against **what your program actually does**.

Reuse L7 复用递归例子

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


print(factorial(3))
```

1 **Ask a concrete question**

What is n in each recursive call?

2 **Pause and inspect**

Observe the line, variables and stack.

3 **Follow up with evidence**

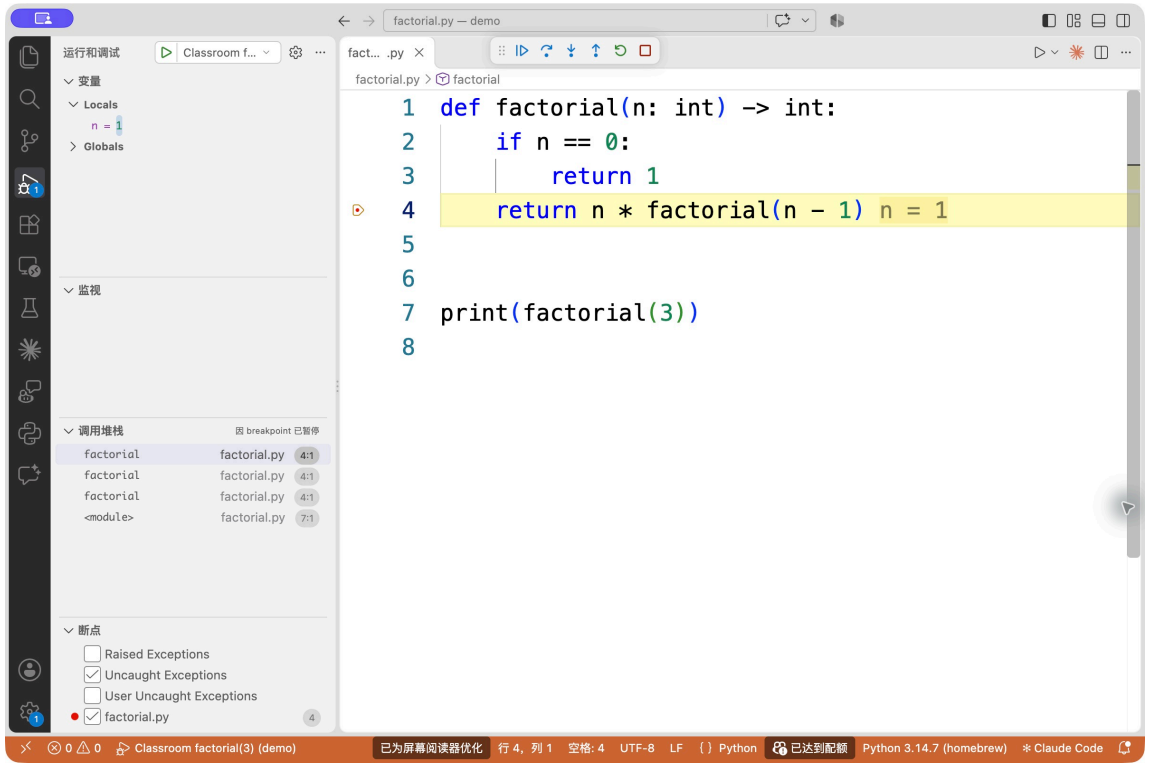
Send the code + paused state to chat.

AI 帮你分析；调试器提供实际状态；自己核对后继续追问。

Pause a Python Program in VS Code

Open **factorial.py** and pause before line 4 executes.

准备：安装 Microsoft Python 与 Python Debugger，打开解压后的项目文件夹。



1 Select Python 3.13

Python: Select Interpreter

2 Set a breakpoint

Click beside line 4 in factorial.py.

3 Start debugging · F5

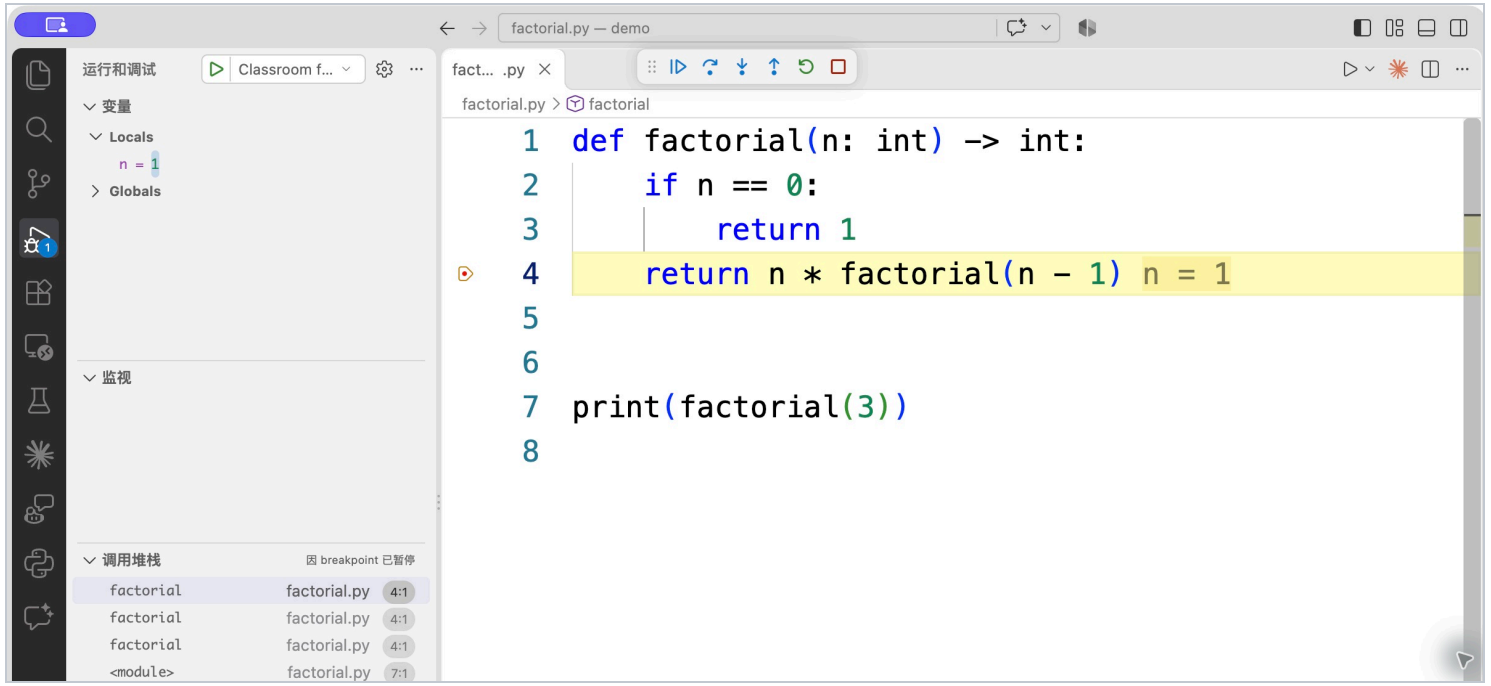
Run → Start Debugging

4 Continue twice · F5

Observe n: 3 → 2 → 1.

Inspect a Recursive Call in the Debugger

Each frame keeps **its own local n**.



- **Paused:** factorial(1), n = 1.
- **Waiting:** factorial(2), n = 2.
- **Waiting:** factorial(3), n = 3.
- Select a frame to inspect **its locals**.

Into · F11
进入调用

Over · F10
单步，不进入调用

Out · Shift+F11
运行至当前调用返回

Continue · F5
继续至断点或结束

Ruff and Runtime Tests Check Different Things

Choose evidence that matches **the question you are checking**.

Existing C-1 fixture

```python
import math

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

Rule	Actual finding
F401	math imported but unused
E741	Ambiguous variable name: l
F841	Local variable unused is assigned to but never used
F821	Undefined name missing_name

> 用实际工具结果检查建议，不只看 AI 的解释。

AI and Debugging: The Working Habit

Use **context, local help and your own checks**.

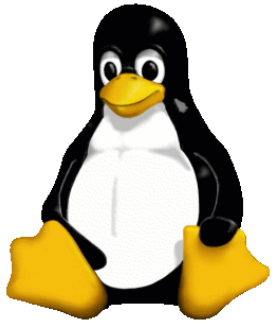


Input + expected + actual → a specific follow-up

谁为程序分配 CPU 时间、管理内存、提供文件访问?

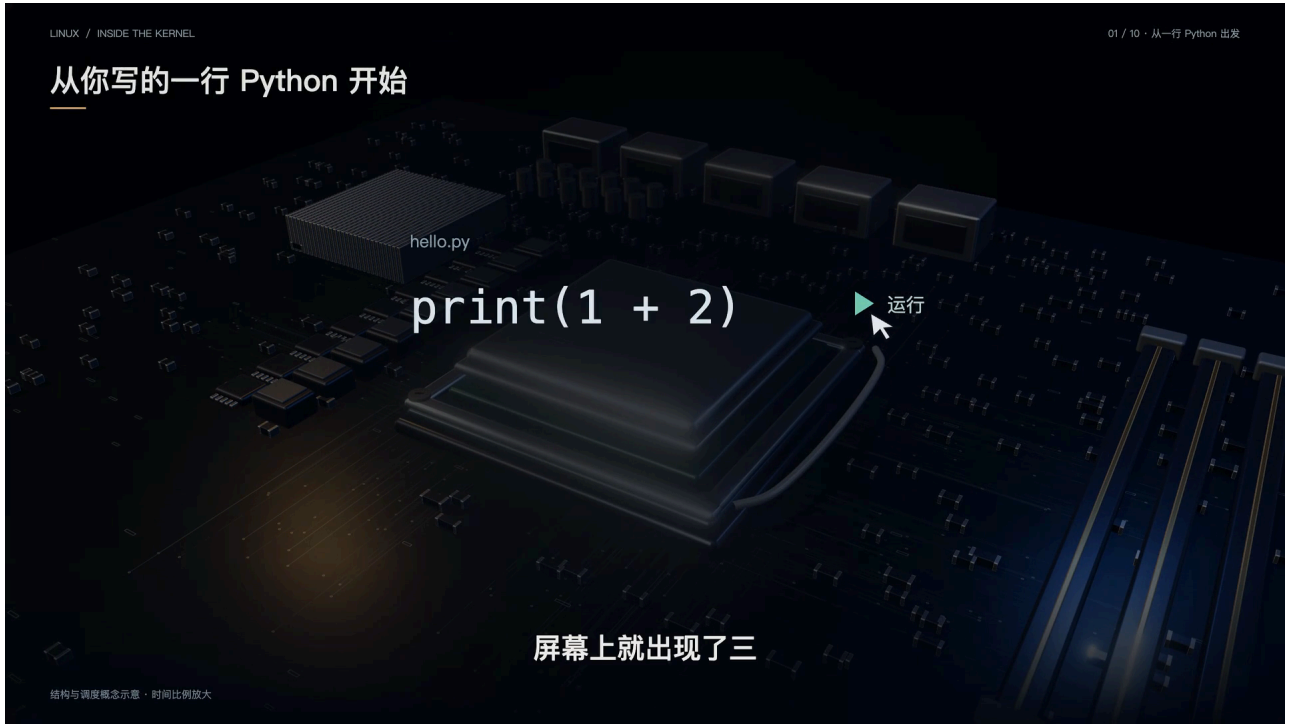
Linux: A Short Introduction

An **operating system** provides the services a program needs.



Tux

图：Larry Ewing



Linux / Inside the Kernel · 8:54 · MP4

从一行 Python，看看操作系统在幕后做什么。

What Does a Running Program Need?

A useful tool needs **services beyond its own code**.

- Read program code from a disk.
- Keep variables in memory.
- Get CPU time while other programs run.
- Write output to a terminal or file.

Unix at Bell Labs

Useful tools grew around **a shared computer**.



Bell Labs archive · early 1970s

- Thompson and Ritchie at a **PDP-11**.
- Unix began at Bell Labs in **1969**.
- Terminals connect people to the system.

> 从真实的使用需要出发，系统与工具逐步发展。

Tools and a Kernel Grow Together

A working system needs **more than a kernel**.



Unix ideas · GNU tools · Linux kernel

Linux 是独立编写的类 Unix 内核。

A Student’s Hobby Project

In **1991**, a student invited others to help.



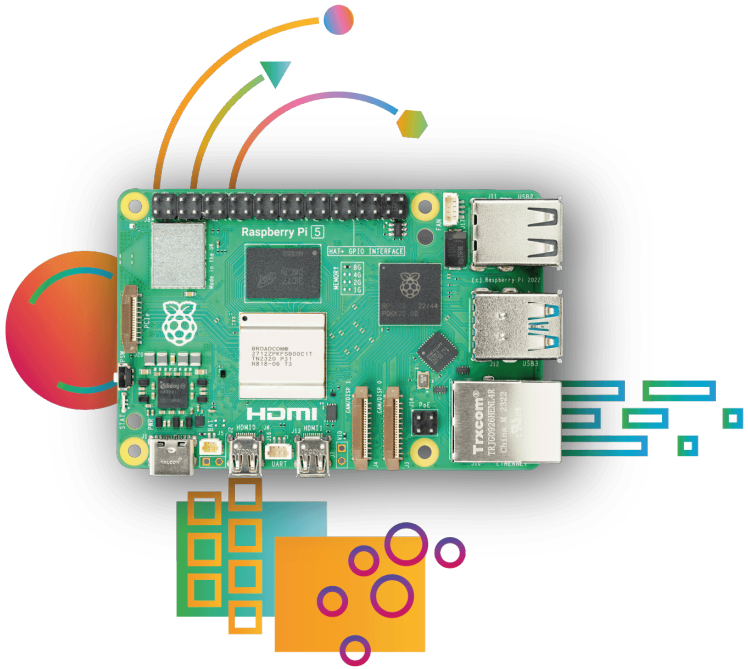
kuvaaja / Linuxmag.com · CC BY-SA 3.0

- Linus Torvalds started Linux as a student.
- The public announcement came in August 1991.
- Early release notes describe a small **i386 kernel**.

照片摄于 2004 年；项目起点是 1991 年。

Linux Beyond the Desktop

One kernel family supports **many kinds of devices**.



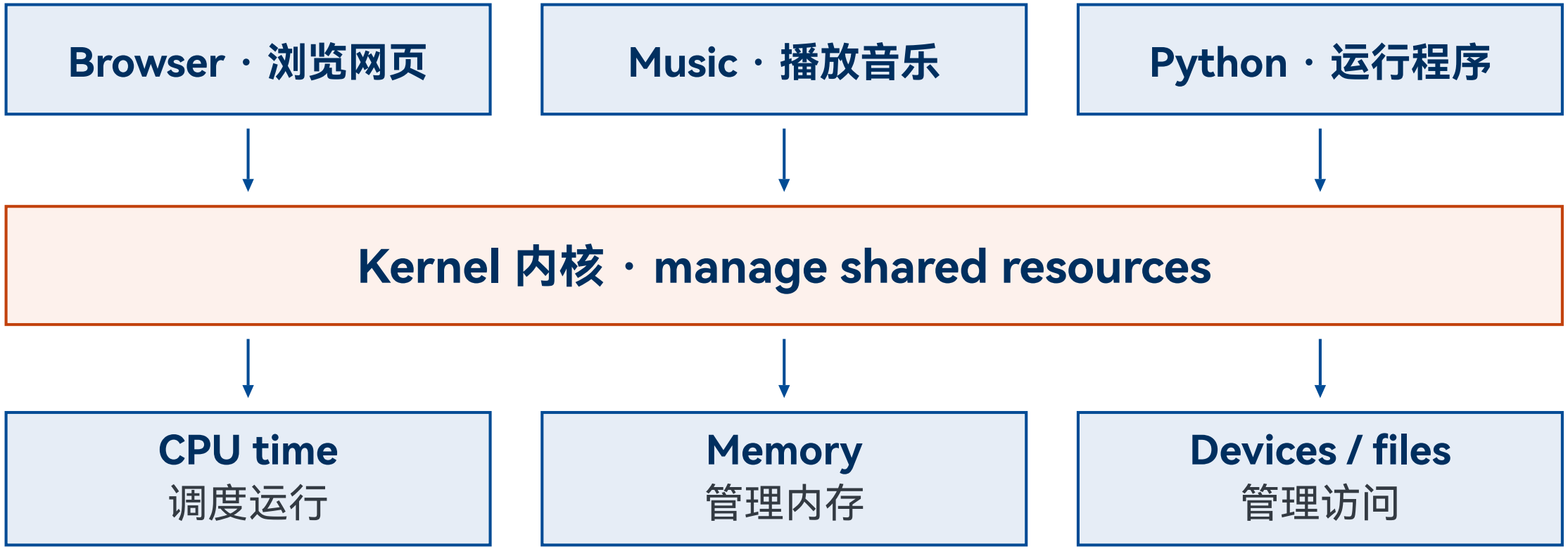
Raspberry Pi · official image

- A small board can run a Linux distribution.
- Servers run Linux-based systems too.
- The kernel and the desktop are different layers.

界面不同，程序仍需要系统管理资源。

The Kernel Manages Shared Resources

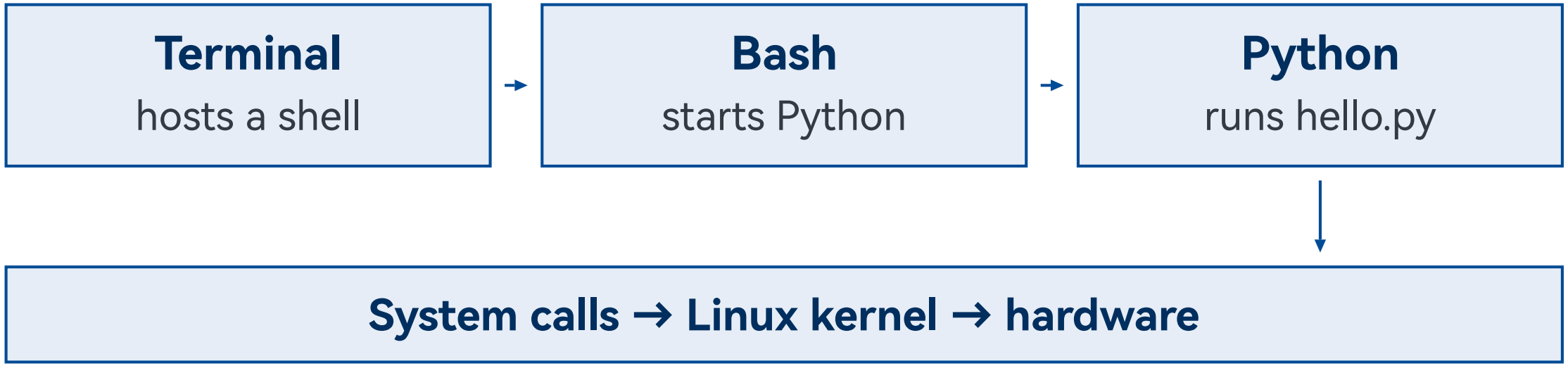
Applications share hardware through **managed services**.



Follow One Program Launch

Bash starts Python; Python requests **system services**.

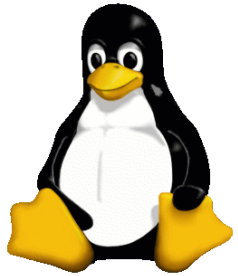
For this terminal launch 这次命令的启动过程



终端是界面；shell 解释命令；Python 执行代码。

Ubuntu Is a Linux Distribution

A usable system needs **more than the kernel**.



Linux kernel 内核

CPU · memory · devices

→

Ubuntu

A Linux distribution 发行版

Applications 应用	Editors · Python · …
Tools 工具	Bash · libraries · apt
Linux kernel 内核	CPU · memory · device access

Other Linux distributions: Debian GNU/Linux · Fedora Linux

> Ubuntu 是一种 Linux 发行版，Linux 内核是它的组成部分。

Linux on a Windows or Mac Laptop

Use the environment you have **actually started**.

- Windows: **WSL 2** runs a Linux kernel in a lightweight VM.
- macOS: **XNU** kernel; many Unix-style tools are similar.
- Check the shell, file location and Python interpreter.

> 命令相似，不代表内核相同。

Know Which Shell You Are Using

Commands depend on a **shell and filesystem context**.

Windows · usual defaults

C:\Users\Ana\project

Report.txt ≈ report.txt

PowerShell / cmd

Linux · typical home FS

/home/ana/project

Report.txt ≠ report.txt

Bash / other shells

大小写行为还取决于文件系统与配置。

Create a Folder and Run Python

Start with **one small program**.

Command 命令

```
mkdir -p terminal-demo
cd terminal-demo
touch notes.txt
# Save print("hello") in hello.py
python3 hello.py
```

Actual output 实跑结果

```
hello
```

先保存 hello.py，再在它所在的目录运行。

Paths, Completion and the Interpreter

Know **where you are** and **which Python runs**.

Useful checks 确认环境

```
pwd
cd "course files"
python3 --version
python3 -m pip --version
```

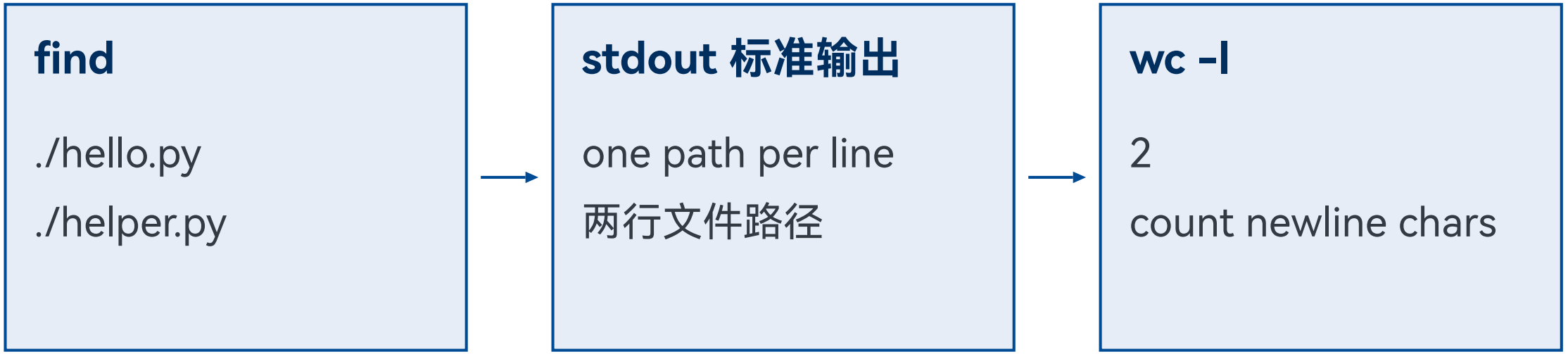
- Quotes keep a spaced path together.
- Tab completes a known name.
- python3 -m pip uses that Python.
- apt and pip manage different software.

先确认当前位置和解释器，再判断命令为什么出错。

Count Files with a Pipe

find selects paths; **wc -l** counts their lines.

```
find . -type f -name '*.py' | wc -l
```



管道传文本：两行路径交给 wc -l，得到 2。

Save Command Output

> sends standard output to a file.

Command 命令

```
python3 hello.py > output.txt
cat output.txt
```

Actual output 实跑结果

```
hello
```

> 覆盖；>> 追加；cat 查看保存的内容。

Find Text in a File

Use **grep** to locate a line in your source.

Command 命令

```
grep -n -F 'print(' hello.py
```

Actual output 实跑结果

```
1:print("hello")
```

先找一个文件中的一行，再扩展到目录搜索。

Stop, Search and Get Help

Use a few shortcuts to **keep working**.

- **Ctrl+C**: interrupt a foreground command.
- **Up / Ctrl+R**: recall or search history.
- **Tab**: complete a path or command.
- **man / --help**: inspect syntax; **q** leaves a pager.

> 会找帮助，比背完整命令表更实用。

Predict: How Many Python Files?



扫码作答

https://taohuang.info/cs1602/zh/quiz/8/

Linux: What You Can Now Explain

Quiz 1: **2 Python files** — find selects, wc counts.

- **Kernel** manages CPU, memory and devices.
- **Distribution** combines a kernel, tools and applications.
- **Shell** reads commands and starts programs.

> 能解释系统各层，也能在终端完成几件小事。

Reference · Requests and Failure Reports

Use facts that another person can **reproduce**.

- Request: local issue + relevant code + constraints.
- Environment: OS, shell, Python and directory.
- Failure: command + input + expected + actual.
- Follow-up: cause and smallest relevant change.

> 例如：解释 PDF/pdf 的差异，只建议 ext 的修改。

Reference · Inspect, Restore and Disclose

Preserve a known version before a **small revision**.

Inspect before restoring

```
git status
git diff
git diff -- inventory.py
git restore -- inventory.py
```

- Read the diff before committing.
- restore discards uncommitted changes.
- Save a copy if those changes matter.
- Declare actual generation and checks.

恢复之前，先确认要丢弃哪些改动。

Reference · Terminal Quick Guide

Connect each command to **one observable action**.

Action	Command	Action	Command
Locate	pwd / ls / cd	Count	find … | wc -l
Create	mkdir -p / touch	Save	command > report.out
Read	cat / head / less	Interrupt	Ctrl+C
Search	grep -rn -F	Help	man / --help

> 从当前目录查找，先看清命令选择的文件类型。

Reference · Windows, WSL and macOS

Match paths to the **actual environment**.

Environment	Distinction
Windows	PowerShell / cmd; drive paths
WSL 2	Linux kernel in a lightweight VM
macOS	XNU; Unix-style tools

> 相似命令不等于相同内核与文件系统。

- C: is typically **/mnt/c** in WSL.
- Linux home: commonly **/home/name**.
- Case behavior depends on filesystem/config.
- Prepare WSL and course Python before class.

Reference · Traceback and Debugging

Combine the traceback with **paused state**.

- Follow calls to the failing operation.
- Read exception type and message.
- print checks a few known values.
- A breakpoint pauses before execution.

Action	Meaning
Into	enter called function
Over	execute current line
Out	run until this call returns
Continue	run to breakpoint/end

暂停位置与选中的调用帧要分别看。

Reference · Tools and Environment Checks

Check the environment **before changing system software**.

Environment checks

```
python3 --version
python3 -m pip --version
ruff --version
ruff check practice.py
ruff format --check practice.py
```

- The Ruff extension is another entry.
- Enabled rules determine findings.
- Use course Python, not sudo pip.
- Retest behavior after repairs.

> 没有 lint 警告，也要运行功能检查。