



上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

Introduction to Computation (CS1602) · Lecture 8

AI-Assisted Programming I

Instructor: Tao Huang

Part 3: Problem Solving with AI · Fall 2026



| Code Review and Verification 代码审查

L07 → L08 · the same checks, now applied to another source.

- **Input assumptions 输入前提**: which inputs are allowed?
- **Termination 终止**: base cases and progress, as in L07
- **Results 结果**: returned values and state changes

| 候选代码、行为约定、检查证据要相互对应。



| A Candidate Implementation 候选实现

The sample passes. The code is short.

Is that enough to accept it? 样例通过后，还要检查什么？

Quiz 1 · Lists and Side Effects



扫码作答 · 一题

1 别名、remove 与输入变化

Predict both printed lists. Submit once.

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

| Lecture Structure

- | | | |
|----------|--------------------|---|
| 1 | Generate 生成 | Keep a candidate; do not confuse an explanation with execution. |
| 2 | Specify 约定 | Define inputs, results, boundaries and side effects. |
| 3 | Review 审查 | Read the implementation; check APIs and readability. |
| 4 | Verify 验证 | Test failures and scale; debug and recheck corrections. |
| 5 | Accept 采纳 | Record the source, changes and actual evidence. |
-

I • Generated Code and Evidence

A sample gives limited evidence.

What has actually been checked?

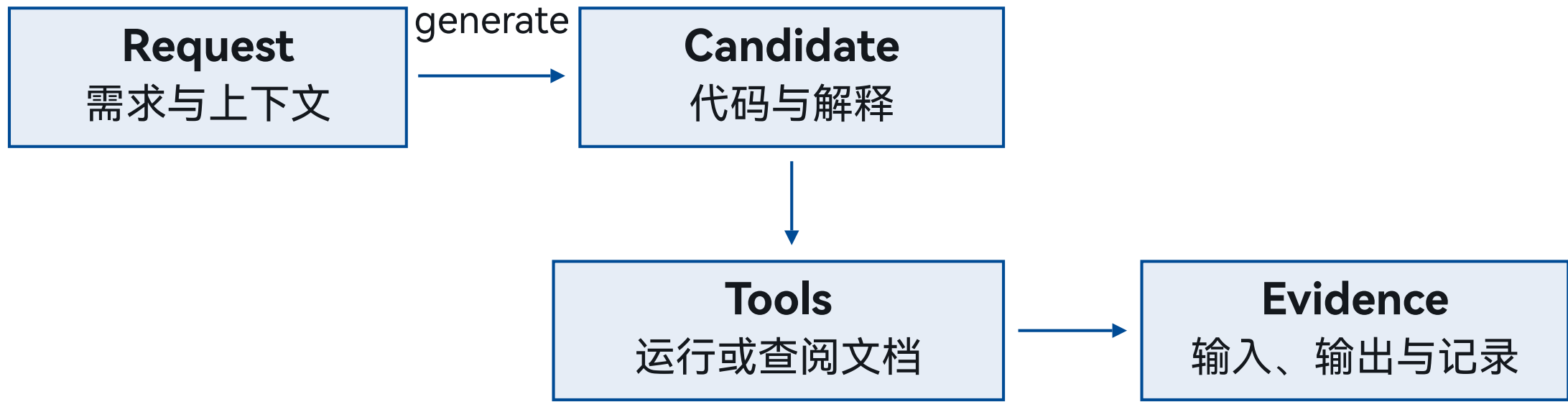


| Language Models 大语言模型

- **Token 文本单位**: a word, part of a word or punctuation
- **Generation 生成**: context → subsequent tokens → code and explanation
- **Evidence 证据**: recorded checks of that implementation

| 解释不能当作运行记录；工具实际执行的检查要单独核对。

| Generation and Tool Execution



Acceptance: compare the contract, code and evidence.

| API Availability 接口是否存在

code

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

traceback

```
Traceback (most recent call last):  
  File "example.py", line 2, in <module>  
    print(s.reverse())  
          ^^^^^^^^^  
AttributeError: 'str' object has no attribute 'reverse'
```



| Documentation Checks 文档核对

code

```
1 s = "hello"  
2 print(s[::-1])  
3 print(s)
```

output

```
olleh  
hello
```

| Empty Inputs 空输入

code

```
1 def average(nums: list[float]) -> float:
2     return sum(nums) / len(nums)
3
4 print(average([1, 2, 3]))
5 print(average([]))
```

traceback

```
Traceback (most recent call last):
  File "example.py", line 5, in <module>
    print(average([]))
    ~~~~~^
  File "example.py", line 2, in average
    return sum(nums) / len(nums)
    ~~~~~^~~~~~
ZeroDivisionError: division by zero
```

| Failure Categories 错误类别

Check 检查	Concrete case 具体例子	Question 审查问题
API availability	<code>"hello".reverse()</code>	Does that method exist on a string?
Boundary behavior	<code>average([])</code>	Is empty input allowed, and with what behavior?
Runtime version	<code>print "hello"</code>	Is the syntax valid in Python 3.13?
Performance	A search through a growing list	Is the repeated work suitable for the size?
Readability	<code>f(l, D)</code>	Can the purpose and state be understood?

| 不报错，不等于已满足功能、规模和可读性要求。



| Runtime Compatibility 运行环境

Python 2 syntax · excerpt

```
print "hello"
```

Python 3 syntax

```
print("hello")
```

| Review Evidence 审查依据

Which evidence is missing when only an explanation is supplied?

- **Requirement 需求**: input domain and promised behavior
- **Implementation 实现**: bounds, state updates and side effects
- **Documentation 文档**: actual APIs and runtime version
- **Observed checks 检查记录**: test results and required scale

II · Function Specifications 函数规格

Failures often expose unstated assumptions.

Which behavior must be preserved?

I Ambiguous Requirements 不完整的需求

“Merge two lists, sorted.”

- Are both inputs **already sorted**?
- Must **duplicates** be kept?
- A **new list**, or may inputs be changed?
- Which **input sizes** must be handled?

| 先把未决定的行为写清楚，再评价实现。

| Signatures and Behavior 签名与行为

Code / excerpt 代码或片段

```
def merge_sorted(  
    a: list[int], b: list[int]  
) -> list[int]:  
    ...
```

- **Signature 签名**: parameter and result types
- **Not specified**: ordering, duplicates or mutation
- **Not enforced**: runtime checks from annotations alone

| ... 表示尚未实现；类型提示不自动完成行为验证。

| A Complete Contract 完整约定

Code / excerpt 代码或片段

```
def merge_sorted(  
    a: list[int], b: list[int]  
) -> list[int]:  
    """Inputs are ascending.  
    Return a new sorted list.  
    Keep every duplicate.  
    Leave inputs unchanged.  
    """  
    ...
```

- **Empty:** $([], []) \rightarrow []$
- **Merge:** $([1, 3], [2]) \rightarrow [1, 2, 3]$
- **Duplicates:** $([1, 1], [1]) \rightarrow [1, 1, 1]$
- **Mutation:** check each input separately

| 接口、文字约定与例子共同说明任务，不是已完成的实现。

| Side Effects 输入变化

code

```
1 def without_two(items):  
2     result = items  
3     result.remove(2)  
4     return result  
5  
6 values = [1, 2, 3, 2]  
7 print(without_two(values))  
8 print(values)
```

output

```
[1, 3, 2]  
[1, 3, 2]
```

| A New Result 新结果

code

```
1 def remove_all(  
2     items: list[int], value: int  
3 ) -> list[int]:  
4     return [x for x in items if x != value]  
5  
6 values = [1, 2, 3, 2]  
7 print(remove_all(values, 2))  
8 print(values)
```

output

```
[1, 3]  
[1, 2, 3, 2]
```

I Checking Input Preservation 输入保全

code

```
1 def remove_all(  
2     items: list[int], value: int  
3 ) -> list[int]:  
4     return [x for x in items if x != value]  
5  
6 values = [1, 2, 3, 2]  
7 before = values.copy()  
8 result = remove_all(values, 2)  
9 print(result == [1, 3])  
10 print(values == before)
```

output

```
True  
True
```

| Prompt Components 提问内容

count_words(text: str) → dict[str, int]

- **Normalize:** split on whitespace and ignore case
- **Punctuation:** strip only . , ! ? ; : from the two ends
- **Empty tokens:** do not count words that become empty

Example 规格例子

Expected 预期

"The cat. THE dog"

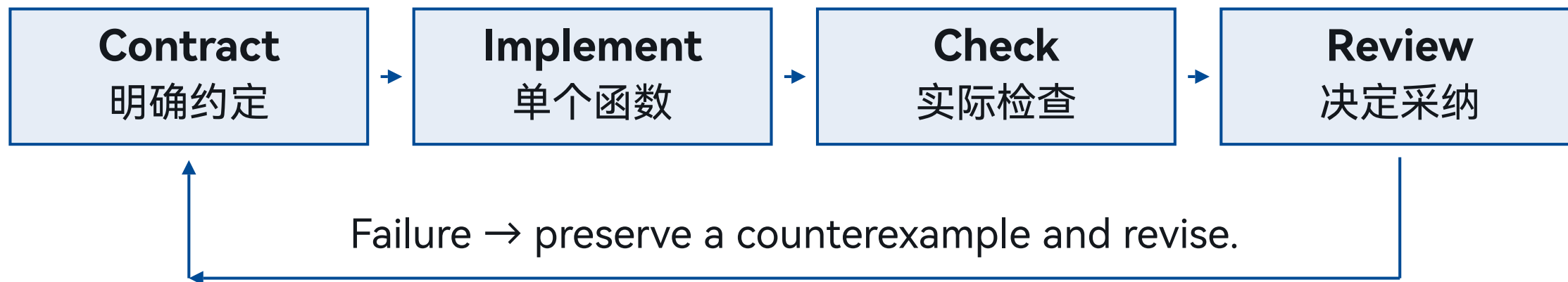
{"the": 2, "cat": 1, "dog": 1}

"" or "... !"

{}

| Python 3.13; 只用 L01–L07 已教语法，仍需检查生成的实现。

| Small Tasks 小规模任务



Keep the original candidate and recheck earlier cases.

| Failure Reports 失败报告

“It is wrong. Try again.” 缺少可复现信息

Record 记录项	A useful report 具体反馈
Input	[7]
Expected	0 · specified for fewer than two values
Actual	ValueError from max([])
Runtime	Python 3.13
Relevant code	The construction of gaps and final max call



III · Readable Code 可读代码

A contract defines what to do.

How can the implementation be reviewed?

| A Readability Review 可读性审查

Which keys are retained? 请先说明函数的行为。

```
1  def f(l, D):  
2      R = []  
3      for i in range(0, len(l)):  
4          if l[i] in D:  
5              if D[l[i]] > 0:  
6                  R.append(l[i])  
7      return R
```

| Names and Documentation 命名与说明

filter_positive(keys, counts) · a name that describes the task

- **Selection 选择**: present keys with positive counts
- **Order 顺序**: keep the input order
- **Duplicates 重复项**: keep repeated keys
- **Naming 命名**: snake_case for variables/functions; UPPER_CASE for constants

| Avoid `l`, `0`, `I` and unwanted built-in shadowing; annotations are not a PEP 8 mandate.

| Behavior-Preserving Refactoring 等价改写

code

```
1  def filter_positive(  
2      keys: list[str], counts: dict[str, int]  
3  ) -> list[str]:  
4      """Keep present keys with positive counts, in input order."""  
5      return [key for key in keys  
6              if counts.get(key, 0) > 0]  
7  
8  keys = ["a", "b", "c"]  
9  counts = {"a": 1, "b": 0, "c": 5}  
10 print(filter_positive(keys, counts))
```

output

```
['a', 'c']
```

| Whitespace and Layout 空格与布局

Recommended 推荐布局

```
spam(ham[1], {eggs: 2})
```

```
if x == 4:  
    print(x, y)
```

```
i = i + 1
```

Style issue · not necessarily syntax

```
spam( ham[ 1 ], { eggs: 2 } )
```

```
if x == 4 :  
    print(x , y)
```

```
i=i+1
```

| Continuation Lines 续行

Align or hang 参数续行

```
result = some_function(first,  
                        second)  
  
result = some_function(  
    first,  
    second,  
)
```

Operators and line length

```
income = (gross_wages  
          + interest  
          - deduction)
```

PEP 8 code limit: 79

Project/tool limits may differ.

| Formatters and Linters 格式与静态检查

Work 分工	Command 工具	What it does 作用
Format 格式化	<code>ruff format file.py</code>	Adjust layout and whitespace
Lint 静态检查	<code>ruff check file.py</code>	Report enabled rules
Test 行为检查	Inputs + expected results	Check a specified requirement

- **Black** is an alternative formatter, not an extra requirement.
- **Ruff configuration** selects the rules; defaults are not all of PEP 8.

| 排版整齐或没有诊断，都不能代替功能验证。



| Rule Diagnostics 规则诊断

Checked file 检查文件

```
import os  
print(total)
```

Actual Ruff report 实际报告

```
ruff 0.16.10  
Rules: F401, F821  
  
F401 · line 1  
`os` imported but unused  
  
F821 · line 2  
Undefined name `total`
```

| Formatting and Behavior 格式与行为

Will formatting add empty-input behavior? 格式化能补上这个约定吗?

```
1 def average(nums: list[float]) -> float:  
2     return sum(nums) / len(nums)  
3  
4 print(average([]))
```

| Review Checklist 审查检查表

- Interface and contract 接口与约定
- Names and readable structure 命名与结构
- State updates, bounds and returns 状态与结果
- Unknown APIs and side effects 陌生接口与副作用
- Finite tests are evidence, not a universal proof.



IV · Tests and Debugging 测试与调试

Review identifies concrete questions.

Which input exposes a failure?

| Tests from the Contract 由需求设计测试

max_gap · sort for comparison, but **leave the input unchanged**

Input 输入	Expected 预期	Purpose 检查内容
[3, 9, 1, 15]	6	Ordinary unsorted values
[]	0	Empty input
[7]	0	One element
[2, 2, 2]	0	Repeated values
[-3, -1, -10]	7	Negative values

| A Minimal Counterexample 最小反例

code

```
1 def max_gap(nums: list[int]) -> int:
2     ordered = sorted(nums)
3     gaps = [ordered[i + 1] - ordered[i]
4             for i in range(len(ordered) - 1)]
5     return max(gaps)
6
7 print(max_gap([7]))
```

Error line

ValueError: max()
iterable argument
is empty

| A Guard and Regression Checks 边界与回归

code

```
1 def max_gap(nums: list[int]) -> int:
2     if len(nums) < 2:
3         return 0
4     ordered = sorted(nums)
5     gaps = [ordered[i + 1] - ordered[i]
6             for i in range(len(ordered) - 1)]
7     return max(gaps)
8
9 print(max_gap([3, 9, 1, 15]))
10 print(max_gap([7]))
11 print(max_gap([]))
```

output

```
6
0
0
```

| Coverage and Limitations 覆盖与限制

- **Boundaries 边界**: empty, one, duplicates, zero and negatives
- **State 状态**: output order and input preservation
- **Scale 规模**: the actual data size and performance requirement
- **Domain 输入范围**: wrong-type handling is contract-dependent

| 测试能暴露反例；有限测试通常不能证明所有输入正确。

| Reading a Traceback 阅读回溯

code

```
1 def read_number(text):  
2     return int(text)  
3  
4 read_number("abc")
```

traceback

```
Traceback (most recent call last):  
  File "example.py", line 4, in <module>  
    read_number("abc")  
    ~~~~~^  
  File "example.py", line 2, in read_number  
    return int(text)  
ValueError: invalid literal for int() with base 10: 'abc'
```

| A State-Update Failure 状态更新错误

Input: [1, 1, 0, 1, 1, 1]. What will be returned?

```
1  def longest_run(bits: list[int]) -> int:
2      current = 0
3      best = 0
4      for bit in bits:
5          if bit == 1:
6              current += 1
7          else:
8              best = max(best, current)
9              current = 0
10     return best
```

| Breakpoints 断点

Code · selected frame

```
else:  
    best = max(best, current)  
    current = 0  
return best
```

Variables · longest_run

```
best = 2  
current = 3
```

Paused before return; the final run has not updated best.

调试界面示意 · verified state, not a screenshot

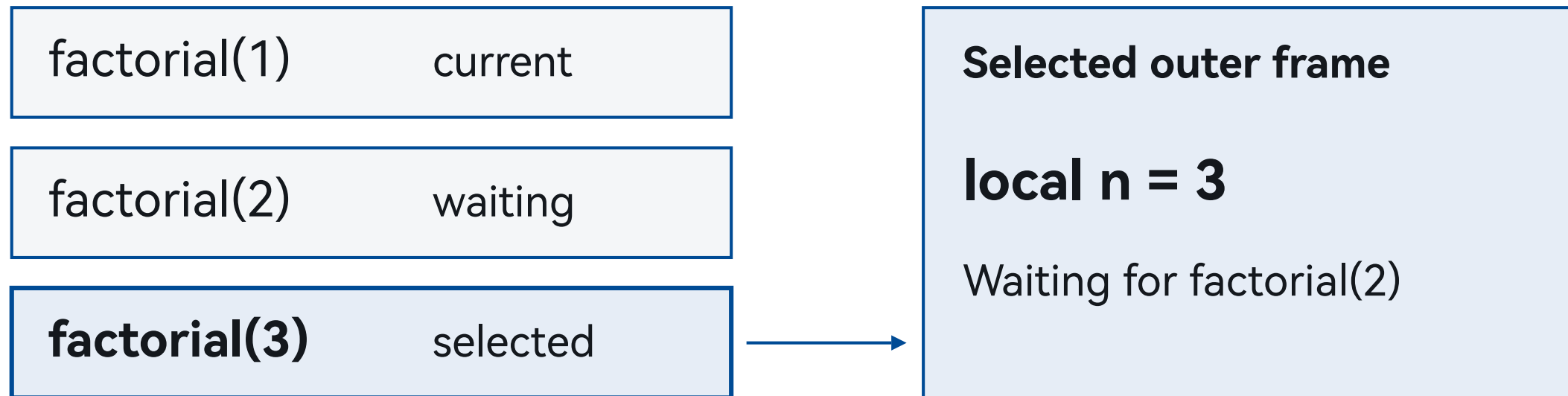
| Stepping and Watches 单步与观察

Action 动作	Common key 按键	Execution 执行效果
Step Over	F10	Execute this line without entering the call
Step Into	F11	Enter the function called on this line
Step Out	Shift+F11	Finish this function and return to its caller
Continue	F5	Resume until the next breakpoint or completion

| Variables / Watch use the **selected frame** while paused. Use toolbar actions if keys differ.

| Recursive Call Frames 递归调用帧

Call stack · $n = 1$ is current



局部变量属于各自调用 · logical frames, not memory addresses

| Fix and Recheck 修正与复查

code

```
1 def longest_run(bits: list[int]) -> int:
2     current = 0
3     best = 0
4     for bit in bits:
5         if bit == 1:
6             current += 1
7         else:
8             best = max(best, current)
9             current = 0
10    return max(best, current)
11
12 print(longest_run([]))
13 print(longest_run([1]))
14 print(longest_run([1, 1, 0, 1, 1, 1]))
```

output

```
0
1
3
```



V · Performance Checks 效率检查

Correct outputs are necessary.

How does work grow with input size?

I Repeated List Searches 反复列表查找

code

```
1 def dedup(items: list[int]) -> list[int]:
2     result = []
3     for x in items:
4         if x not in result:
5             result.append(x)
6     return result
7
8 print(dedup([3, 1, 3, 5, 1]))
```

output

```
[3, 1, 5]
```

I Operation Counts 操作计数

Distinct n	List comparisons	Set queries
10	100	20
100	10,000	200
1,000	1,000,000	2,000

Two ordered passes over n distinct integers. Different units.

| A Set and an Ordered Result 集合与顺序

Membership and output order need different structures.

```
1  def dedup(items: list[int]) -> list[int]:
2      seen = set()
3      result = []
4      for x in items:
5          if x not in seen:
6              seen.add(x)
7              result.append(x)
8      return result
```

| Equivalent Results 结果等价核对

Input 输入	List search 列表版	Set-assisted 集合版
[]	[]	[]
[3, 1, 3]	[3, 1]	[3, 1]
[-1, 0, -1]	[-1, 0]	[-1, 0]

- **Results match** on these recorded cases.
- **Inputs unchanged**: check separately, not from the result alone.

| 提速不能悄悄改变顺序、输入范围或重复项约定。

| Scale and Measurement 规模与测量

Measure 测量	Meaning 含义	Limit 限制
Operation count	Work performed on a chosen case	Identify the unit and input distribution
Elapsed seconds	Observed time in this environment	Hardware/runtime/background work affect it

- **Same task, data and environment** before comparing
- **Equivalent results and repeated measurements** before drawing conclusions

| 一次计时倍率不是通用规律；L11 再系统分析时间与空间。



VI · Review and Acceptance 审查与采纳

Behavior and scale have been checked.

What should be recorded before submission?

| An IPv4 Specification IPv4 规格

is_valid_ipv4(s: str) → bool

- **Structure 结构**: exactly four parts separated by .
- **Characters 字符**: ASCII digits only, each value **0–255**
- **Leading zeros 前导零**: only the single digit 0 is allowed
- **Reject 排除**: signs, empty parts and any whitespace

| 输入为字符串；先按这份规格写测试，不给出实验完整实现。



I Independent Test Cases 独立用例

Three valid cases: 0.0.0.0 · 192.168.1.1 · 255.255.255.255

Invalid input 非法输入

Contract boundary 对应约定

""

Empty input

"1.2.3"

Part count

"256.1.1.1"

Value range

"01.2.3.4"

Leading zero

" 1 .2.3.4"

ASCII only · full-width digit

"1.2.3.4 "

No whitespace · trailing space

| isdigit() 和 int() 的接受范围不自动等于本题规格。



| An AI Usage Declaration 使用声明

1	Source 来源	Actual tool/model or instructor candidate; unknown model → say unknown
2	Scope 范围	Name the functions or blocks generated/adopted.
3	Changes 修改	Record manual corrections; say if none were made.
4	Checks 验证	Actual inputs, expected results, observations and regressions

| 未使用 AI 就据实说明; “AI 保证正确”不是测试记录。

| Responsibility and Safe Use 责任与安全

- **Understand 理解**: explain each adopted block
- **Preserve 保留**: original code, counterexamples and corrections
- **Inspect 审查**: unfamiliar file, install and network operations
- **Protect 保护**: remove private information, passwords, keys and tokens

| 沿用现有课程政策，本页不另订处罚。

Quiz 2 · Review and Verification



扫码作答 · 五题

- | | |
|--------|--------|
| 1 函数签名 | 2 最小反例 |
| 3 操作计数 | 4 调用帧 |
| 5 使用声明 | |

All topics have been covered. Submit once.

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



I Quiz Review · Contracts and Evidence

- | | | |
|----------|----------------------|---|
| 1 | Q2 · contract | Annotations do not enforce sorting or input preservation. |
| <hr/> | | |
| 2 | Q3 · failure | [1]: candidate 0 , required 1 ; the final run was omitted. |
| <hr/> | | |
| 3 | Q4 · counts | This input: n^2 comparisons vs $2n$ member queries ; different units. |
| <hr/> | | |

| 源代码检查次数不能当作固定 CPU 指令数。

I Quiz Review · Frames and Declaration

Q5 · select factorial(3): local n is still 3

- **Current inner call:** `factorial(1)` has its own `n = 1`.
- **Selected outer call:** `factorial(3)` retains `n = 3`.
- **Q6 · declaration:** actual source, adopted scope and real verification
- **Never invent:** tool/model names, conversations or test results



| Lab 8 · A, B and C 实验三段

Part 实验段	Work 学习任务	Marks 计分
A · 40 min	Five functions + four review records	Raw 24 · weekly 60%
B · 35 min	IPv4: specification → tests → review → correction	Raw 12 · weekly 40%
C · 15 min	Ruff / recursive frames / command-line recap	Completion only

| **6 code cards + 3 text cards**, prepared locally; server synchronization pending.



| Solo Assignment 1 · Independent Work

Task 任务	AI rule 要求	Submission 提交
Solo assignment 1	Level 0 · independent Hanoi implementation	5% of course grade · due 10 Nov, 23:59 GMT+8
Lab08	Level 1 · assistance with a truthful declaration	Use the week-eight HW3 entries after publication

| 跨周作业顶栏角标不能替代题目要求；不要让 AI 生成或改写个人作业 1。

| Data and Operations 数据与操作

- **Before acceptance:** contract, code, documentation and environment
- **Recorded evidence:** boundaries, mutation, scale and corrections

L09 · Data and operations 数据与操作

How can similar records and their operations be organized together?



AFTER CLASS

Reference 参考

Errors, conventions and debugging

Legacy exercises and complete records

| Common Exceptions 常见异常

重点区分：类型 / 值

- **TypeError**

Meaning 原因

Wrong **type** for an operation

Example 例子

```
"3" + 1
```

- **ValueError**

Accepted type, invalid **value**

```
int("abc")
```

其他常见异常 · 逐项查阅

Meaning 原因

Example 例子

- **SyntaxError**

Invalid **syntax**

```
if True · missing :
```

- **IndentationError**

Invalid **indentation**

```
if body not indented
```

- **NameError**

Name not defined

```
print(total) · undefined
```

- **IndexError**

Index out of range

```
[1, 2][2]
```

- **KeyError**

Dictionary **key** not found

```
{ }["id"]
```

- **ZeroDivisionError**

Division by **zero**

```
1 / 0
```

- **AttributeError**

Attribute not available

```
"abc".reverse()
```

- **RecursionError**

Recursion **depth limit** reached

```
f() keeps calling f()
```

阅读顺序：异常名 → 原因 → 例子；回溯中的**失败语句**另行核对。

| Naming and Documentation 命名与文档

Object 对象	Convention 约定	Example 例子
Variables / functions	snake_case	student_count, filter_positive
Constants	UPPER_CASE	MAX_RETRY
Classes	CapWords	BankAccount · introduced later
Modules / packages	Short lowercase names	utils, mypackage
Internal names	A leading underscore	_helper · convention, not access enforcement

| 文档补充任务、参数与结果；避免混淆名字与内置名覆盖，后续规范见 PEP 8。

| Layout Reference 布局规范

Code / excerpt 代码或片段

```
def f(x, y=2):  
    ...
```

```
def g(x: int, y: int = 2) -> int:  
    ...
```

Avoid crowded statements.

- **Indentation:** four spaces per level
- **Continuation:** align or use a hanging indent
- **Separation:** two blank lines at top level
- **Consistency:** operator breaks; no trailing whitespace

| Annotated defaults: `y: int = 2`; unannotated defaults: `y=2`. Avoid crowded statements.

| Imports and Comments 导入与注释

Code / excerpt 代码或片段

```
import os
import sys

import numpy as np

from mypackage import helper
```

- **Group imports:** standard → third-party → project
- **Blank lines** separate the groups.
- **Avoid wildcard imports** and unknown shadowing.
- **Docstrings** describe interfaces; keep comments current.

Python 3 source defaults to UTF-8. Third-party/project imports here are formatting excerpts.

| Formatting Tools 格式工具

Command 命令

Effect 用途

```
python3 -m pip install ruff
```

Install into that interpreter's environment

```
ruff check file.py
```

Lint under the selected rules

```
ruff format --check file.py
```

Check formatting without rewriting

- **Editor extension** is another route; Black is an alternative formatter.
- **Record version and rules**; do not force administrator pip.

| 没有 lint 报告，仍不能证明行为正确。



| Debugging Reference 调试速查

- | | | |
|----------|--------------------|--|
| 1 | Pause 暂停 | Breakpoint: stop before a statement executes. |
| 2 | Inspect 查看 | Variables / Watch: the selected-frame context while paused |
| 3 | Navigate 切换 | Call stack: choose a call; Over / Into / Out / Continue |
| 4 | Record 记录 | Minimal input, expected/actual, source and runtime; print remains useful. |
-

| The Zen of Python Python 之禅

- **Explicit** is better than implicit.
- **Readability** counts.
- In the face of **ambiguity**, refuse the temptation to guess.
- If the implementation is **hard to explain**, it's a bad idea.

| **Tim Peters** · design principles for explicitness, readability and explanation; not a correctness certificate.

| Leap-Year Tests 闰年测试

Code / excerpt 代码或片段

```
def is_leap_year(  
    year: int  
) -> bool:  
    return (  
        (year % 4 == 0  
         and year % 100 != 0)  
        or year % 400 == 0  
    )
```

Actual output 实跑输出

```
1900 False  
2000 True  
2024 True  
2100 False
```

- **Century boundary:** 1900 / 2100 are not leap years.
- **400-year rule:** 2000 is a leap year.
- **Positive Gregorian years;** all three versions remain in the notes.

| Prime-Number Boundaries 素数边界

Code / excerpt 代码或片段

```
def is_prime(n: int) -> bool:
    if n < 2:
        return False
    divisor = 2
    while divisor * divisor <= n:
        if n % divisor == 0:
            return False
        divisor += 1
    return True
```

Actual output 实跑输出

```
-1 False
0 False
1 False
2 True
4 False
9 False
97 True
```

- **$n < 2 \rightarrow \text{False}$** : include negative values, 0 and 1.
- **$\text{divisor}^2 \leq n$** : keep the equality; both versions are in the notes.

[Complete runnable example 完整程序](#)

| Digit-Square Sums 数位平方和

Code / excerpt 代码或片段

```
def sum_digit_squares(  
    n: int  
) -> int:  
    if n < 10:  
        return n * n  
    smaller = sum_digit_squares(  
        n // 10)  
    return (n % 10) ** 2 + smaller
```

Actual output 实跑输出

```
0 0  
2 4  
9 81  
19 82  
29 85
```

- **Old base case** returned n , not $n*n$.
- **Input 2** exposes it; input 19 happens to pass.
- **Loop / recursive / string** versions are retained.

I Happy Numbers 快乐数

Code / excerpt 代码或片段

```
def is_happy(n: int) -> bool:
    seen = set()
    while n != 1:
        if n in seen:
            return False
        seen.add(n)
        n = sum_digit_squares(n)
    return True
```

Actual output 实跑输出

True
False

- **Requires sum_digit_squares** from the complete program.
- **Positive integers only**; 0 is outside this contract.
- **Seen set** detects repeated states, not their counts.

[Complete runnable example](#) 完整程序

| Consecutive Ones 连续的 1

Code / excerpt 代码或片段

```
if bits[i] == 1:
    j = i
    while (j < len(bits)
           and bits[j] == 1):
        j += 1
    best = max(best, j - i)
    i = j
else:
    i += 1
```

Actual output 实跑输出

0
0
1
3

- **Outer i moves to j**, skipping the scanned run.
- **Nesting alone** does not imply quadratic work.
- **Empty / zero / one / final run** are separately checked.

[Complete runnable example 完整程序](#)

| Continuous Increasing Runs 连续递增段

Code / excerpt 代码或片段

```
def longest_increasing_run(
    nums: list[int]
) -> int:
    if not nums:
        return 0
    current = 1
    best = 1
    for i in range(1, len(nums)):
        if nums[i] > nums[i - 1]:
            current += 1
        else:
            current = 1
        best = max(best, current)
    return best
```

Actual output 实跑输出

```
0
3
1
1
```

- **Continuous:** do not skip positions.
- **Strictly increasing:** an equal value resets the run.
- **Empty input → 0;** ordinary and repeated values are checked.

[Complete runnable example](#) 完整程序

| Anagrams and Multiplicity 字符次数

Code / excerpt 代码或片段

```
def is_anagram(  
    s: str, t: str  
) -> bool:  
    left = char_counts(s)  
    right = char_counts(t)  
    return left == right
```

Actual output 实跑输出

```
True  
False  
True
```

- **Requires char_counts(s)** from the complete program.
- **Lowercase ASCII**; empty strings match.
- **set("aab") == set("abb")** loses the unequal character counts.

[Complete runnable example 完整程序](#)

| Boolean Conditions 布尔条件

Value 对象	bool(value) 真假值	value == True 比较
False	False	False
True	True	True
2	True	False
[1]	True	False

| 实际运行结果：真假值不等于和 True 比较；+= / + 也不能机械互换。



| Development and Review 开发与审查

- | | | |
|---|------------|---|
| 1 | Plan 设计 | Define meaningful state and interfaces . |
| 2 | Build 实现 | Implement and test small units , not a large opaque block. |
| 3 | Review 核对 | Reuse results and check unfamiliar operations. |
| 4 | Improve 修正 | Preserve counterexamples, changes and regressions . |
-

| 函数长度、嵌套层数或代码简短，都不是正确性证明。



| Review Records and Submission 审查记录

1	Contract 规格	The task and the permitted inputs/results
2	Source 来源	Original candidate, tool/model or instructor source
3	Evidence 证据	Actual input / expected / observed records
4	Correction 修正	Changes, regressions and a truthful declaration

9 week-eight cards are ready locally; not deployed. Six lecture exercises remain; Solo1 stays Level 0.