

Creative Development

AP Computer Science Principles • Unit 1

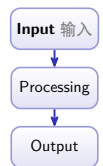
AP Computer Science Principles



Creative Development

What we will cover

- 1 Collaboration
- 2 Program function & purpose
- 3 Design & development
- 4 Finding & fixing errors



1 Collaboration

Creative Development

↳ Collaboration

Software is a team sport

A large **program** 程序 is designed, written, tested & fixed by a team. **Collaboration** 协作 means working together toward a shared goal.

Driver 驾驶员
types the code

Navigator 导航员
watches, thinks ahead, catches mistakes

Pair programming 结对编程: two people, one computer, swapping roles. Teams merge work with **version control** 版本控制 and explain it with **documentation** 文档.



Designed, written and tested by people

Creative Development

↳ Collaboration

Purpose, the IPO model, and why comments matter

every program has a purpose it solves a problem or pursues an interest – the Create task asks you to state yours

input 输入 what the program takes in – typed, clicked, sensed, or read from a file

processing what it does with that input

output 输出 what it produces – printed, drawn, stored or sent

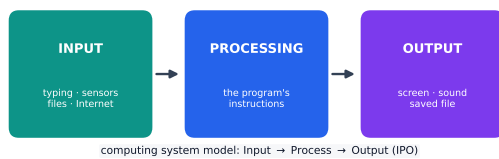
comments 注释 ignored by the machine, written for people: they record purpose, assumptions and anything non-obvious

Collaboration 协作 brings more perspectives and catches more errors, which is why the exam treats it as a **skill**, not a courtesy.

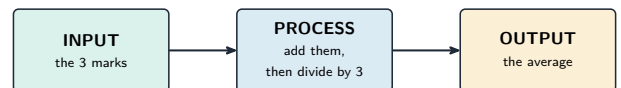
2 Function & purpose

Purpose, function & the IPO model

Purpose 目的 = **why** a program exists; **function** 功能 = **what** it does.
Every program fits **input** → **processing** → **output** (the IPO model).
Many are **event-driven** 事件驱动: they wait and react to **events** 事件 (a click) via an **event handler** 事件处理程序.



Every program decomposes into input



Every program decomposes into input, processing, and output

3 Design & development

The iterative development cycle

The process is **iterative** 迭代的, **not** a straight line: investigate, design, implement, test – **then repeat**.

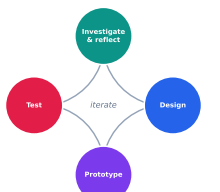
Development is **incremental**: build and test a **small piece**, then add the next.



Development is iterative and collaborative

Break a large problem into pieces (**decomposition** 分解); plan with a **program requirement** 需求 and **pseudocode** 伪代码; keep it readable with a **comment** 注释.

Software is built by an iterative – one

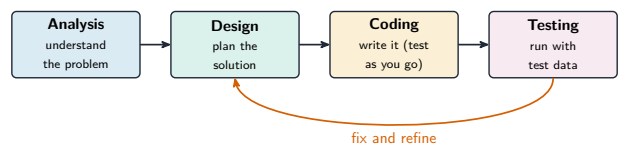


Iterative + incremental: refine after testing; ship working pieces early

Software is built by an iterative, incremental development process

Software is built by an iterative, incremental development process

Software is built by an iterative – two

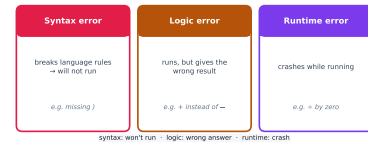


The stages of program development, with testing feeding back to fix and refine

The stages of program development, with testing feeding back to fix and refine

4 Errors

Four kinds of error



Syntax error 语法错误 – breaks the language rules; won't run

Logic error 逻辑错误 – runs, but wrong answer

Runtime error 运行时错误 – crashes mid-run (divide by zero)

Overflow error 溢出错误 – value too large for its storage

Creative Development
Errors

Name the error, then name the test that catches it

syntax error 语法错误	breaks the language's rules – caught by the compiler or interpreter before it runs
run-time error 运行时错误	crashes during execution – dividing by zero, indexing past the end
logic error 逻辑错误	runs and gives the wrong answer – the hardest, because nothing complains
overflow error 溢出错误	a value too large for the bits available
how to catch each	read the message for the first two; hand-trace or test with known values for a logic error

This pairing – error type plus a testing strategy that would find it – is a recurring multiple-choice and Create-task theme.

Creative Development
Errors

Finding & fixing bugs

A **bug 缺陷** is a mistake; **debugging 调试** is finding & fixing it.

test many inputs, incl. **edge cases 边界情况**

compare with the **expected output 预期输出**

divide & check; re-test after each fix

A logic error runs with no crash – “it ran” is not “it's correct”. Only comparing output to what you expected catches it.

Creative Development
Errors

Hand-tracing with a trace table

A **trace table 追踪表** records **each variable's value** as the program runs – line by line, by hand.

It is how you find a **logic error** that no crash will ever reveal.

count	total	output
1	5	
2	12	
3	20	20

each variable, at each step

Creative Development
Errors

Worked example – a logic error hiding in plain sight

A program should print the average of two numbers, but runs $\text{avg} = a + b / 2$

Order of operations: `/` runs **before** `+`, so it computes $a + \frac{b}{2}$ – not the average. Test with $a = 4$, $b = 6$: the bug gives $4 + 3 = 7$; the fix $\text{avg} = (a + b) / 2$ gives $\frac{10}{2} = 5$.

Testing with **known inputs** is exactly how you find and **confirm** a logic error – the program never crashed.

Key points

Collaboration + version control build better software

Every program is **input** → **processing** → **output**

Development is **iterative**: plan, prototype, test, refine

Syntax, logic, run-time & overflow **errors**

Exam tips

- Much of CSP is assessed through the **Create** and written performance tasks – explain your reasoning clearly, not just your result.
- Know the benefits of collaboration and how diverse perspectives reduce bias in a program.
- Use precise vocabulary (iterative development, program requirements) when you describe a design process.
- Give and take feedback constructively; credit collaborators and sources.
- Break a large problem into smaller modules that a team can build in parallel.

5

Recap

Unit 1 – key takeaways

1 Create task

explain reasoning, not just the result

2 Collaboration

diverse perspectives reduce bias

3 Vocabulary

iterative development, program requirements

4 Modules

break a large problem so a team can build in parallel

Check yourself

- 1 $avg = a + b / 2$ runs and prints a number. Which error is it?
- 2 Which error stops the program from running at all?
- 3 Name the three parts of the IPO model.
- 4 What does the navigator do in pair programming?

Answers 1. a logic error 2. a syntax error 3. input, processing, output 4. watches, thinks ahead, catches mistakes