Creative Development

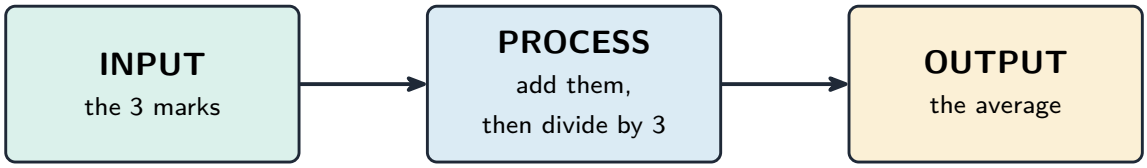
AP Computer Science Principles

Collaboration

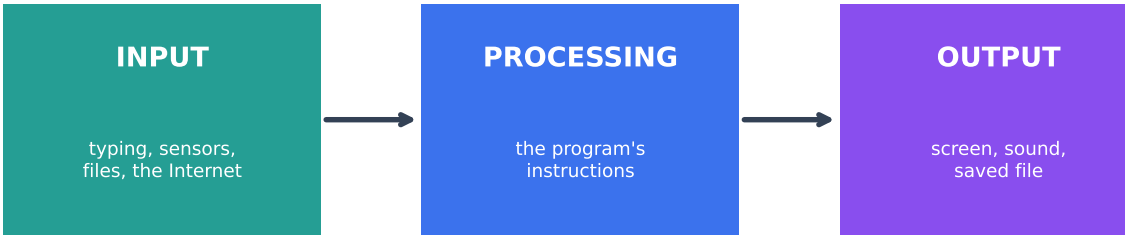
Computing is a **collaborative** 协作 activity. Working in a team brings more perspectives, catches more errors, and produces better programs than working alone. Good collaboration uses **consensus building**, clear communication, and each member's strengths. **Pair programming** 结对编程–two people at one computer, one typing and one reviewing – is a common practice. On the exam, you should be able to explain how collaboration improved a program (more ideas, fewer bugs, wider testing).

Program Function and Purpose

Every program is written for a **purpose** –it solves a problem or pursues an interest. A program takes **input** 输入, processes it, and produces **output** 输出. Inputs can come from a user, a device, a file, or another program; outputs can be visual, audible, textual, or a signal to a device. Being able to state a program's purpose, and describe its inputs and outputs clearly, is a core skill (and part of the Create performance task).



Every program decomposes into input, processing, and output

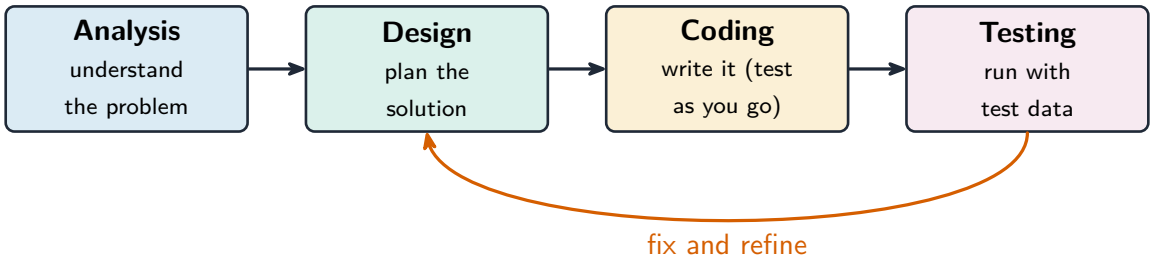


Every program follows the input-processing-output model

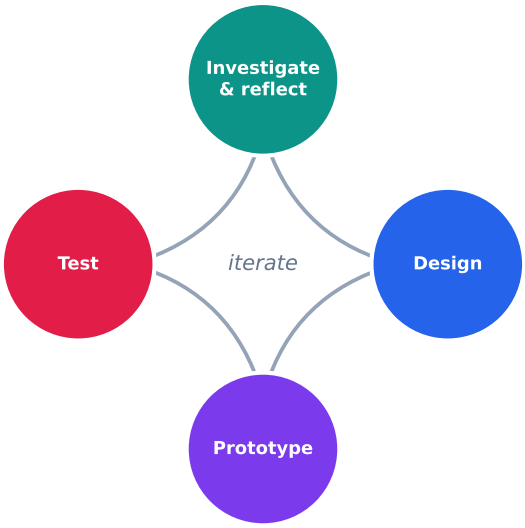
Program Design and Development

Programs are built through an **iterative** 迭代 process, not in one straight line: investigate the problem and users, design (often with a **diagram** or written plan), implement in code,

and test –then repeat. A large problem is broken into smaller pieces (**decomposition** 分解). **Comments** 注释 and clear naming document the design so others (and your future self) can understand it. Development is **incremental** –build and test a small piece, then add the next.



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



Software is built by an iterative, incremental development process

Identifying and Correcting Errors

A **bug** is an error in a program; **debugging** 调试 is finding and fixing it. Three kinds:

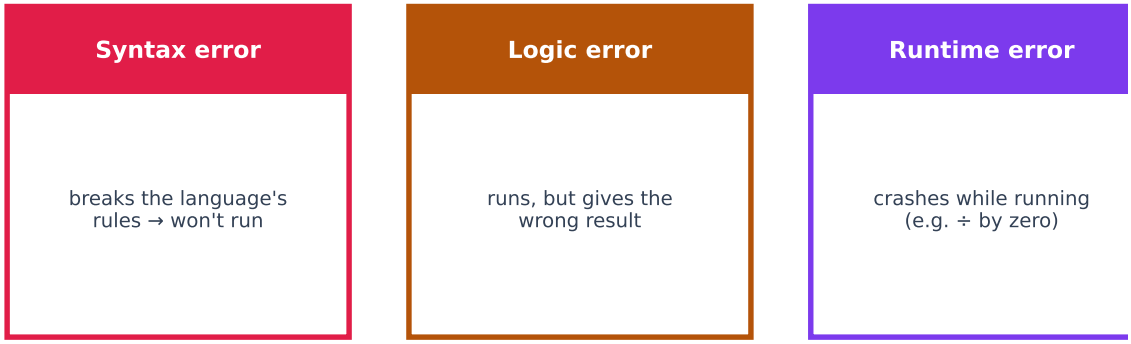
count	total	output
1	5	
2	12	
3	20	20

A trace table records each variable's value as the program runs, to find bugs

- a **syntax error** 语法错误 breaks the language's rules, so the program will not run;
- a **runtime error** 运行时错误 crashes the program while it runs (e.g. dividing by zero);
- a **logic error** 逻辑错误 lets it run but gives the wrong result.

Find bugs by **testing** with different inputs (including edge cases), adding **print statements** to see values, and hand-tracing the code. Fixing one bug at a time and re-testing is the reliable method.

Exam skill: be able to name the type of an error and describe a testing strategy that would catch it –a recurring multiple-choice and Create-task theme.



Three kinds of programming error: syntax, logic, and runtime

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.