预处理器与限定符

C Reference

预处理器、const、static、enum

预处理器 (preprocessor) 在编译之前运行。`#define` 创建一个宏 (macro)— 在所有地方被替换的纯文本。`const` 创建一个不能改变的常量 (constant)。函数内部的 `static` 让变量在多次调用之间保留它的值。`enum`(枚举,enumeration) 给一组从 0 开始的整数起名字。

```c
#include <stdio.h>

#define MAX 100                     // macro: text replaced before
↪  compiling

enum Color { RED, GREEN, BLUE };    // named integers 0, 1, 2

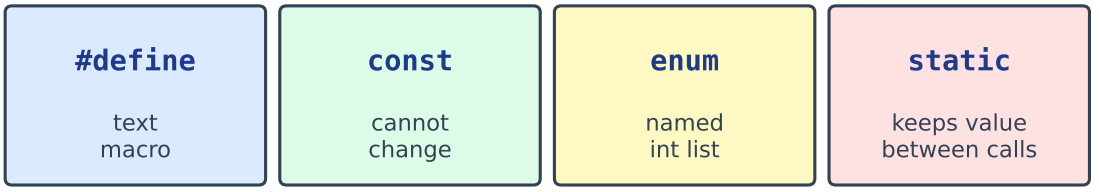
int next_id(void) {
    static int count = 0;           // keeps its value between calls
    count++;
    return count;
}

int main(void) {
    const double PI = 3.14;         // cannot be changed
    printf("%d %.2f\n", MAX, PI);           // 100 3.14
    printf("%d %d %d\n", RED, GREEN, BLUE);  // 0 1 2
    int a = next_id();
    int b = next_id();
    printf("%d %d\n", a, b);                // 1 2
    return 0;
}
```

更大的程序会拆成多个文件: 声明放进头文件 (header)`mine.h`, 再用 `#include "mine.h"` 引入 — 引号表示你自己的文件,`< >` 表示标准库。包含保护 (`#ifndef MINE_H` / `#define MINE_H` / `#endif`) 防止同一个头文件被引入两次。

常见错误

- `#define` 只做纯文本替换, 没有类型检查; 宏体和参数都要用括号包起来。
- `const` 值一旦设定就不能再改。
- `static` 局部变量在两次函数调用之间会保留它的值。



Exam tip: preprocessor runs first; enum values start at 0 unless set

#define, *const*, *enum* 和 *static* 使代码清晰