

Vector operator+(const Vector& b) const;
函数声明中两个const的作用分别是什么？



在 `Vector operator+(const Vector& b) const;` 这个成员函数声明中，两个 `const` 的作用分别如下：

第一个 `const` (`const Vector& b`)

表示函数参数 `b` 是一个 **常量引用**，即函数内部不能通过 `b` 修改传入的 `Vector` 对象。

这是对输入参数的约束，保证 `operator+` 不会意外修改 `b` 的值。

第二个 `const` (在函数声明的末尾)

表示这个成员函数是一个 **常量成员函数 (const member function)**，即该函数不会修改调用它的对象 (`*this`)。

这意味着在该函数内部，不能修改当前对象的任何成员变量（除非成员变量被声明为 `mutable`）。

这种 `const` 成员函数可以被常量对象调用。

示例说明

cpp

复制 下载

```
class Vector {  
public:  
    Vector operator+(const Vector& b) const; // 两个 const 分别约束参数和成员函数  
    // ...  
};
```

如果 `v1` 和 `v2` 是 `Vector` 对象，`v1 + v2` 不会修改 `v1`（因为函数是 `const`）也不会修改 `v2`（因为参数是 `const Vector&`）。

这种设计符合直觉，因为加法运算通常不应该改变操作数本身，而是返回一个新的结果。

总结

`const Vector& b` → 保证不修改参数 `b`。

`const` 成员函数 → 保证不修改当前对象 (`*this`)。

这种写法既安全又清晰，是良好的 C++ 编程实践。

