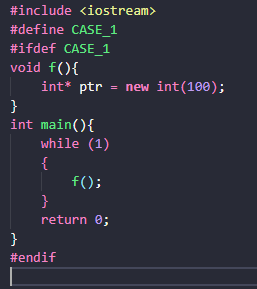
Smart Pointer:

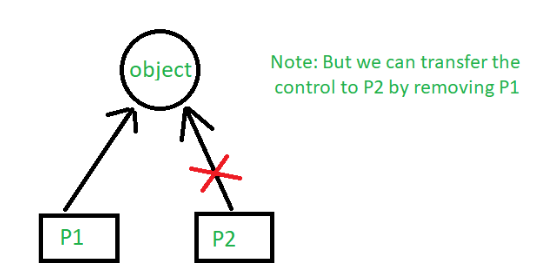
1. Vấn đề khi sử dụng Pointer thông thường:

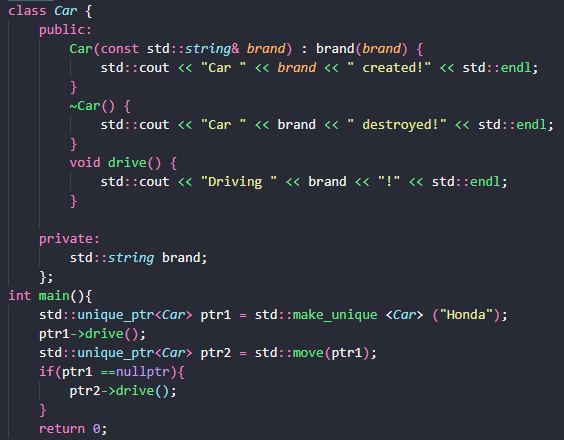


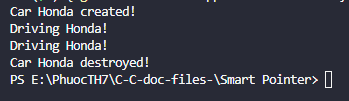
* CHương trình trên bị lỗi memory Leak vì vùng nhớ trên HEAP không được giải phóng.
* Ptr được lưu ở stack, nhưng vùng nhớ chứa kiểu int 100 nằm ở HEAP, khi kết thúc hàm thì ptr bị hủy tuy nhiên vùng nhớ trên HEAP không được giải phóng, khi chương trình chạy đủ lâu bộ nhớ Heap sẽ bị chiếm dụng hết
* C++ cung cấp thư viện chuẩn để sử dụng Smart Pointer, những Pointer này sẽ tự động giải phòng bộ nhớ heap khi con trỏ bị hủy.

Có 3 loại Smart Pointer thường dùng: (nhớ include <memory> )

1. **Unique Pointer:**

* Sử hữu độc quyền đối tượng được cấp phát động, có nghĩa là ngăn cấm việc copy, nghĩa là chỉ có 1 unique pointer duy nhất trỏ đến vùng nhớ đó.
* Tuy nhiên vẫn cho phép tranfer(move) miễn sau chỉ có 1 unique pointer trỏ đến vùng nhớ này
* 





Ở đây ta thấy 2 điều:

* Ta không cần phải giải phóng bộ nhớ bằng tay
* Đối tượng chỉ có 1 con trỏ trỏ đến, khi dùng move() để nhượng quyền, ptr1 sẽ thành Null.

Nên dùng std::make\_unique, make\_share khi dùng smart pointer thay vì new, mặc dù dùng new cũng không sai, tuy nhiên sẽ an toàn hơn khi khong dùng new:

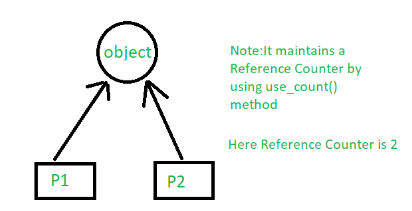
**std::unique\_ptr<MyClass> ptr(new MyClass());**

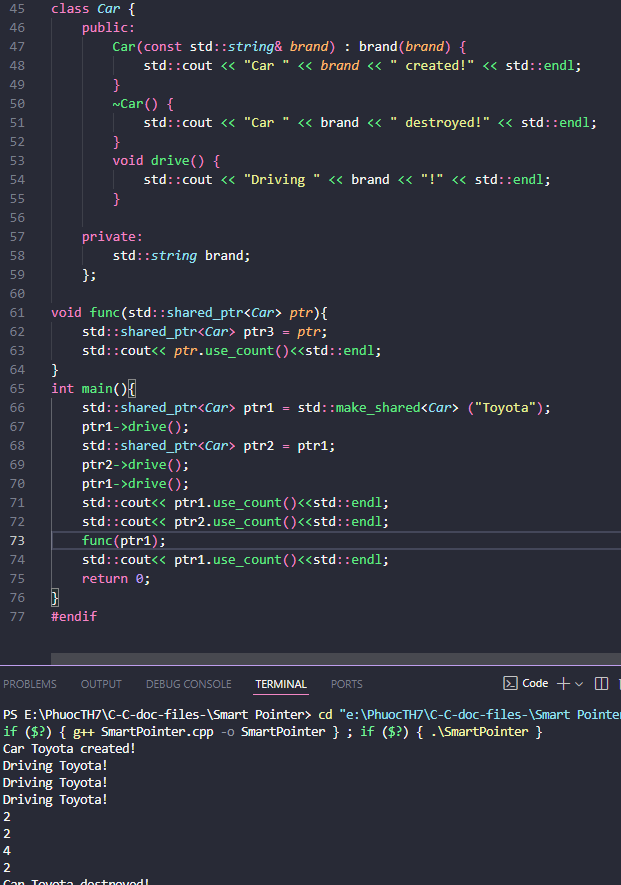
**someFunctionThatThrows();** // Nếu exception xảy ra ở đây, mà MyClass chưa được gán vào ptr → rò rỉ bộ nhớ!

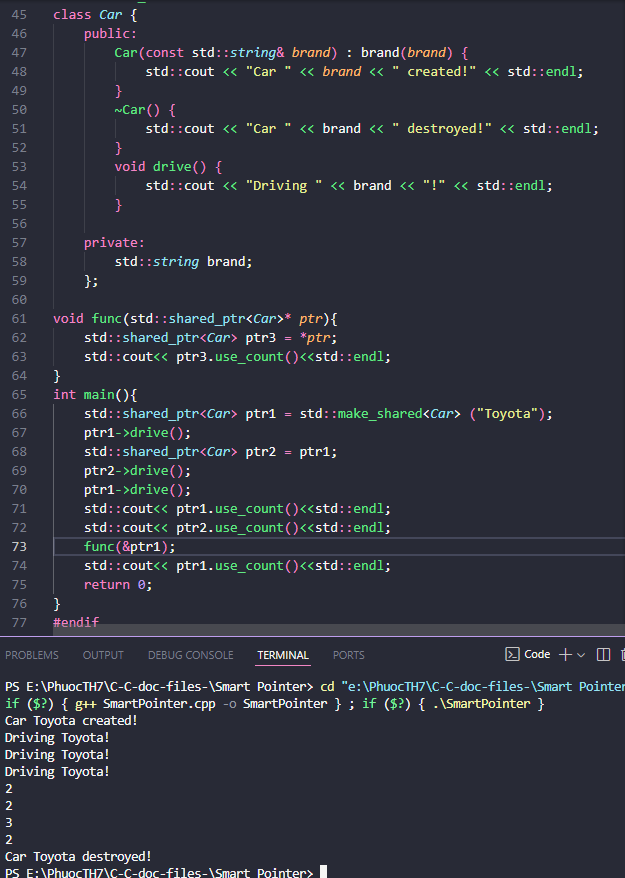
**auto ptr = std::make\_unique<MyClass>();**

**someFunctionThatThrows();** // Không sao cả, make\_unique sẽ đảm bảo không rò rỉ

**Share pointer:**

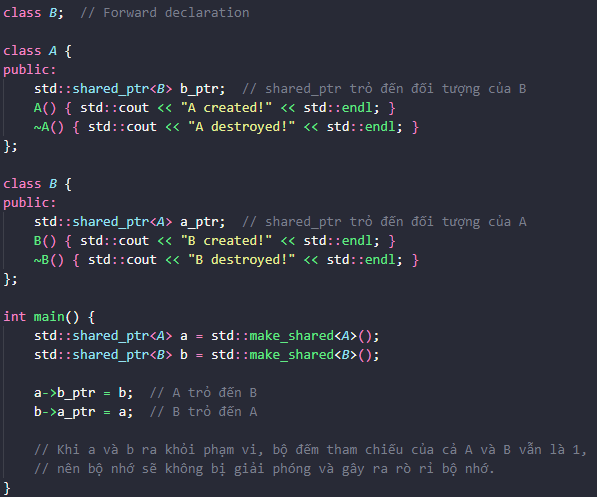
* 
* Smart Pointer cho phép chia sẻ quyền sở hữu của đối tượng duy nhất. Sử dụng bộ đếm tham chiếu (reference pointing) để theo dõi số lượng share\_ptr đang trỏ đến đối tượng. Khi mà số lượng = 0 nó sẽ tự giải phóng.
* Để biết số lượng ta dùng **use\_count()** method



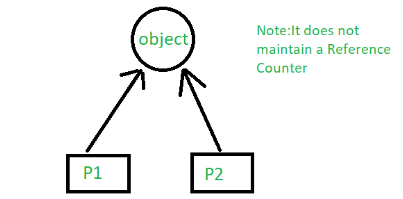


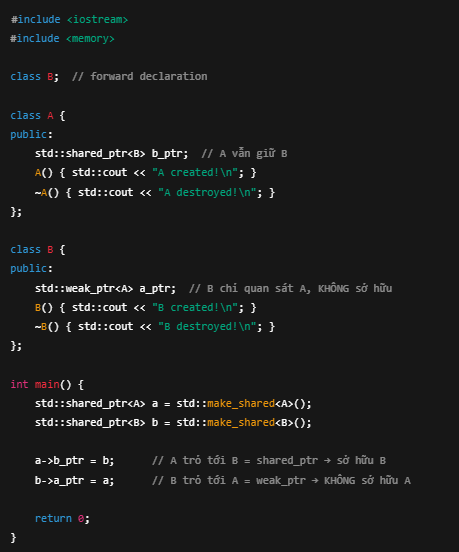
**Weak Pointer:**

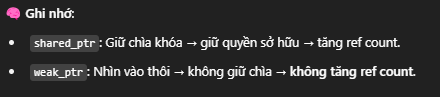
* Weak pointer sinh ra để giải quyết vấn đề **Cyclic Dependency,**điều này xảy ra khi chúng ta có 2 class chứa 2 share pointer trỏ vào nhau





* Weak Pointer không sở hữu đối tượng nó trỏ đến, nó không làm tăng bộ đếm tham chiếu như shared\_ptr
* 





* Để dùng weak\_ptr để truy cập vùng nhớ, ta phải dùng lock() để trả về 1 shared\_ptr. Nếu đã được giải phóng nó sẽ trả về NULL.

