

Chiếc Tủ Thần Kỳ: Bí mật tìm đồ trong chóp mắt

Khám phá `unordered_map` trong C++17/20

Slide Learning CPP

Ngày 15 tháng 1 năm 2026

Lộ trình Khám phá (The Map)

- ❶ **Chương 1:** unordered_map là gì? (Chiếc tủ có nhãn dán).
- ❷ **Chương 2:** Cỗ máy băm thức ăn (Hash Function).
- ❸ **Chương 3:** Các thao tác cơ bản (Thêm, sửa, xóa nhãn).
- ❹ **Chương 4:** Điểm mới trong C++17/20 (Ngăn kéo thông minh).
- ❺ **Chương 5:** Iterator - Ngón tay chỉ đường.
- ❻ **Tổng kết:** Ứng dụng trong Lập trình thi đấu (CP).

unordered_map là gì?

Ẩn dụ: Danh bạ điện thoại

Nó giống như danh bạ điện thoại. **Tên bạn bè** là cái nhãn (**Key**), còn **Số điện thoại** là món đồ (**Value**). Bạn chỉ cần nhớ tên là tìm được số ngay lập tức.

- **Mảng (vector):** Phải tìm từ đầu đến cuối (mệt và chậm).
- **unordered_map:** Dẫn thẳng bạn tới món đồ thông qua nhãn dán.
- **Tại sao "Unordered"?** Ưu tiên tốc độ tìm kiếm hơn là việc xếp hàng ngay ngắn.

Ví dụ sống động (Khai báo và Truy xuất)

```
1 #include <iostream>
2 #include <unordered_map>
3 #include <string>
4 using namespace std;
5
6 int main() {
7     // Key: Ten (string), Value: Diem so (int)
8     unordered_map<string, int> bang_diem;
9
10    // 1. Cat do vao tu (Them du lieu)
11    bang_diem["An"] = 9;
12    bang_diem["Ba"] = 7;
13    bang_diem["Chi"] = 10;
14
15    // 2. Lay do ra (Truy xuất)
16    cout << "Diem cua Chi la: " << bang_diem["Chi"] << endl;
17    return 0;
18 }
```

Chương 2: Cỗ máy băm (Hash Function)

Bí mật của tốc độ ánh sáng

Làm sao để biết chính xác ngăn kéo nào chứa tên "An"?

- 1 Bạn đưa nhãn "**An**" vào máy băm.
- 2 Máy băm chữ thành số (ví dụ: số **5**).
- 3 Chiếc tủ mở đúng ngăn số **5** để cất/lấy đồ.

Lưu ý về Key

Nhãn dán (Key) là **duy nhất**. Nếu bạn dán đè nhãn mới lên cùng một tên, giá trị cũ sẽ bị thay thế.

Thao tác Sửa, Xóa và Kiểm tra

```
1 // Kiểm tra tồn tại (C++20)
2 if (ds_lop.contains("Lan")) { ... }
3
4 // Kiểm tra tồn tại (Trước C++20)
5 if (ds_lop.count("Lan")) { ... }
6
7 // Sửa dữ liệu
8 ds_lop["An"] = "MA_MOI";
9
10 // Xóa dữ liệu
11 ds_lop.erase("An");
```

Chương 4: Siêu năng lực C++17 và C++20

C++17: Structured Bindings

Giúp lấy cả Nhãn và Đồ ra cùng lúc cực kỳ gọn gàng.

```
1 // Dung dau ngoac vuong [ten, ma]
2 for (auto const& [ten, ma] : ds_lop) {
3     cout << "Ban " << ten << " co ma: " << ma << endl;
4 }
```

C++20: Contains

Sử dụng `.contains("Key")` thay vì dùng `.find()` hoặc `.count()`.

Chương 5: Iterator - Ngón tay chỉ đường

- **Iterator (it):** Đóng vai trò như một con trỏ chỉ vào từng ngăn kéo.
- `it->first`: Truy cập vào cái Nhãn (**Key**).
- `it->second`: Truy cập vào món đồ (**Value**).

Tại sao dùng dấu `->`?

Vì `it` giống như con trỏ, ta dùng `->` để "chỉ tay" vào thuộc tính bên trong ngăn kéo thay vì dùng dấu chấm . thông thường.

Câu hỏi: Nếu có 2 người cùng tên "An", ta thực hiện:

- `ds_lop["An"] = "001";`
- `ds_lop["An"] = "002";`

Kết quả cuối cùng trong ngăn "An" là gì?

Câu hỏi: Nếu có 2 người cùng tên "An", ta thực hiện:

- `ds_lop["An"] = "001";`
- `ds_lop["An"] = "002";`

Kết quả cuối cùng trong ngăn "An" là gì?

Đáp án

Là **"002"**. Vì Key trong `unordered_map` là duy nhất, giá trị sau sẽ ghi đè giá trị trước.

Cạm bẫy trong Lập trình thi đấu (CP)

Anti-hash Test trên Codeforces

Kẻ xấu có thể tạo bộ dữ liệu gây xung đột băm (Collision), khiến tốc độ giảm từ $O(1)$ xuống $O(n)$, gây lỗi **TLE**.

Giải pháp: Custom Hash với Random Seed

```
1 struct custom_hash {
2     size_t operator()(uint64_t x) const {
3         static const uint64_t FIXED_RANDOM = chrono::steady_clock::
4             now().time_since_epoch().count();
5         return splitmix64(x + FIXED_RANDOM);
6     }
7 };
8 // Khai báo an toàn
9 unordered_map<long long, int, custom_hash> safe_map;
```

Thử thách cuối cùng: Inventory Game

Bài toán

Bạn nhất thêm 1 thanh "Kiem" trong khi đã có sẵn 2 thanh trong túi đồ (`unordered_map<string, int> inventory`). Bạn sẽ xử lý thế nào?

Thử thách cuối cùng: Inventory Game

Bài toán

Bạn nhất thêm 1 thanh "Kiem" trong khi đã có sẵn 2 thanh trong túi đồ (`unordered_map<string, int> inventory`). Bạn sẽ xử lý thế nào?

Gợi ý giải pháp

```
inventory["Kiem"] += 1;
```

Chiếc tủ sẽ tìm ngăn "Kiem", lấy số lượng cũ (2) cộng thêm 1 và cất lại số 3 vào ngăn đó!