Chinh phục C++ Vector

Hành trình khám phá chiếc balô thần kỳ

Slide Learning C++

Ngày 14 tháng 1 năm 2026

Lộ trình: Chiếc Balô Thần Kỳ

Chào em! Hãy tưởng tượng chúng ta đang chuẩn bị cho một chuyến đi dã ngoại thú vị. Để hiểu về Vector, chúng ta sẽ đi qua 4 trạm dừng chân chính:

Các trạm dừng chân

1. **Trạm 1: Vector là gì?**
   Tại sao Mảng (Array) giống như cái khay trứng cố định, còn Vector lại là "Chiếc túi thần kỳ"?
2. **Trạm 2: Sắm túi và Xếp đồ (Create & Add)**
   Cách khai báo và nhét từng món đồ vào đáy túi (`push_back`).
3. **Trạm 3: Lục lọi và Tráo đổi (Access & Change)**
   Cách lấy đúng món đồ mình cần hoặc đổi món này lấy món kia.
4. **Trạm 4: Kiểm kê và Soi độ rộng (Loop & Size)**
   Cách rà soát lại toàn bộ đồ đạc và xem túi nặng bao nhiêu.

Trạm 1: Vector là gì? - Tạm biệt "Khay trứng"

Mảng (Array) - Khay trứng

- Kích thước cố định.
- 10 lỗ chỉ đựng được 10 quả.
- Quả thứ 11? Không vừa!
- Chỉ có 2 quả? Lãng phí chỗ.

Vector - Túi thần kỳ

- Co giãn linh hoạt (Mảng động).
- Lúc đầu xẹp lép.
- Nhét đồ vào $\rightarrow$ Phồng lên.
- Lấy đồ ra $\rightarrow$ Co lại gọn gàng.

Chốt lại: Vector chính là một **Mảng động (Dynamic Array)**.

Khởi động Chiếc Balô (Cú pháp)

Để sử dụng được chiếc túi thần kỳ này, chúng ta phải "gọi"nó ra từ thư viện C++.

```cpp
#include <iostream>
#include <vector>  // <--- Day la lenh goi thu vien Vector

using namespace std;

int main() {
  // Chung ta se tao chiec tui o day
  return 0;
}

```

Listing 1: Khai báo thư viện

Lưu ý: Dòng `#include <vector>` giống như việc hô to: *"Doremon ơi, cho tớ mượn cái túi thần kỳ!"*.

Kiểm tra nhanh (Checkpoint)

Câu hỏi

Nếu em muốn lập trình một danh sách **các bạn đăng ký đi tham quan**, nhưng em **không biết trước** sẽ có bao nhiêu bạn tham gia (có thể 5 bạn, có thể 50 bạn).
Em nên dùng **Mảng (Array)** hay **Vector**? Tại sao?

Kiểm tra nhanh (Checkpoint)

Câu hỏi

Nếu em muốn lập trình một danh sách **các bạn đăng ký đi tham quan**, nhưng em **không biết trước** sẽ có bao nhiêu bạn tham gia (có thể 5 bạn, có thể 50 bạn).
Em nên dùng **Mảng (Array)** hay **Vector**? Tại sao?

Đáp án

Vector!
Vì số lượng người tham gia thay đổi liên tục, Vector có thể co giãn để chứa vừa đủ số người, không lo thiếu chỗ hay thừa chỗ như Mảng.

Trạm 2: Sắm túi (Khai báo)

Em phải nói rõ cho máy tính biết túi đựng cái gì (Số nguyên hay Chữ).

Cú pháp

```
vector<Kiểu dữ liệu> Tên_túi;
```

Ví dụ:

- `vector<string> xe_hoi;` → Túi tên `xe_hoi`, đựng chữ.
- `vector<int> diem_so;` → Túi tên `diem_so`, đựng số.

Nhét đồ vào túi (push_back)

Câu thần chú: `.push_back()`.
Hãy hình dung hành động **xếp hàng**:

- Người vào sau phải đứng sau lưng người trước ("Back").

Ví dụ minh họa với túi `xe_hoi`

1. `xe_hoi.push_back("Vinfast");`
   → Túi: [Vinfast]
2. `xe_hoi.push_back("Toyota");`
   → Túi: [Vinfast, Toyota]

Code thực tế: Mua túi và Xếp xe

```cpp
#include <iostream>
#include <vector> // Nho goi thu vien

using namespace std;

int main() {
  // 1. Mua cai tui ten la "xe_hoi" chi dung chuoi ky tu
  vector<string> xe_hoi;

  // 2. Nhet chiec xe dau tien vao
  xe_hoi.push_back("Vinfast");

  // 3. Nhet tiep chiec xe thu hai
  xe_hoi.push_back("BMW");

  // 4. Nhet chiec xe thu ba
  xe_hoi.push_back("Ford");

  // Luc nay trong tui dang la: Vinfast, BMW, Ford
  return 0;
}

```

Kiểm tra nhanh (Checkpoint)

Tình huống xếp hàng trà sữa

1. Bạn **Nam** đứng vào hàng đầu tiên.
2. Bạn **Lan** dùng lệnh `.push_back()`.
3. Bạn **Hùng** dùng lệnh `.push_back()`.

Hỏi: Ai đứng **đầu hàng** và ai đứng **cuối hàng**?

Kiểm tra nhanh (Checkpoint)

Tình huống xếp hàng trà sữa

1. Bạn **Nam** đứng vào hàng đầu tiên.
2. Bạn **Lan** dùng lệnh `.push_back()`.
3. Bạn **Hùng** dùng lệnh `.push_back()`.

Hỏi: Ai đứng **đầu hàng** và ai đứng **cuối hàng**?

Đáp án

- Đầu hàng (vị trí 0): **Nam**.
- Cuối hàng: **Hùng**.

Nguyên tắc: "Vào sau đứng sau".

Trạm 3: Lục lọi và Tráo đổi

Túi đang có: **Vinfast, BMW, Ford**.

Quy tắc vàng: Máy tính đếm từ 0!

Vector giống như dãy tủ gửi đồ có đánh số (Index):

- Ô đầu tiên: **Số 0**.
- Ô thứ hai: **Số 1**.
- Ô thứ ba: **Số 2**.

- `xe_hoi[0]` $\rightarrow$ Vinfast
- `xe_hoi[1]` $\rightarrow$ BMW
- `xe_hoi[2]` $\rightarrow$ Ford

Cách lấy đồ và Đổi đồ

1. Lấy đồ ra (Access)

- Dùng ngoặc vuông: `cout « xe_hoi[0];`
- Dùng lệnh `.at()`: `cout « xe_hoi.at(0);`

2. Tráo đổi đồ (Change)

- Chỉ đúng cái tủ và gán giá trị mới.
- `xe_hoi[0] = "Tesla";`
- **Kết quả:** Túi trở thành [Tesla, BMW, Ford].

Code thực tế: Truy cập và Thay đổi

```cpp
#include <iostream>
#include <vector>
#include <string>
using namespace std;

int main() {
  vector<string> xe_hoi;
  xe_hoi.push_back("Vinfast"); // Vi tri 0
  xe_hoi.push_back("BMW");     // Vi tri 1
  xe_hoi.push_back("Ford");    // Vi tri 2

  // 1. Lay do ra xem: In xe o vi tri 1 (chiec thu 2)
  cout << "Xe o vi tri 1 la: " << xe_hoi[1] << "\n";

  // 2. Trao doi do: Doi xe vi tri 0 thanh Opel
  xe_hoi[0] = "Opel";

  // In lai de kiem tra
  cout << "Xe o vi tri 0 bay gio la: " << xe_hoi[0] << "\n";
  return 0;
}

```

Kiểm tra nhanh (Checkpoint)

Câu hỏi mẹo

Túi có 3 món: `["Opel", "BMW", "Ford"]`.
Nếu em viết lệnh: `cout « xe_hoi[3];` chuyện gì sẽ xảy ra?
A. In ra "Ford".
B. In ra ô trống.
C. Báo lỗi hoặc in ra linh tinh.

Kiểm tra nhanh (Checkpoint)

Câu hỏi mẹo

Túi có 3 món: `["Opel", "BMW", "Ford"]`.
Nếu em viết lệnh: `cout « xe_hoi[3];` chuyện gì sẽ xảy ra?
A. In ra "Ford".
B. In ra ô trống.
C. Báo lỗi hoặc in ra linh tinh.

Đáp án: C

Vì đếm từ 0, nên 3 món đồ nằm ở tủ số **0, 1, 2**.
Tủ số **3** là "tủ ma"(chưa được tạo), chọc vào sẽ bị lỗi!

Trạm 4: Kiểm kê và Soi độ rộng

1. Soi độ rộng (Size) - Túi nặng bao nhiêu?

- Lệnh: `.size()`
- Ví dụ: `cout « xe_hoi.size();` → Ra số 3.

2. Kiểm kê hàng loạt (Loop)

Cách 1: Truyền thống (Dùng chỉ số i)

```
for (int i = 0; i < xe_hoi.size(); i++) {
    cout << xe_hoi[i] << "\n";
}

```

Cách 2: Hiện đại (For-each) - "Quét mã vạch"

```
for (string xe : xe_hoi) {
    cout << xe << "\n";
}

```

Code thực tế: Vòng lặp

```cpp
// ... (Da khai bao vector va push_back)

  // 1. In ra kich thuoc tui
  cout << "Tong so xe la: " << xe_hoi.size() << "\n";

  // 2. Diem danh kieu Truyen thong (Dung i)
  cout << "--- Cach 1: Dung i ---\n";
  for (int i = 0; i < xe_hoi.size(); i++) {
    cout << xe_hoi[i] << "\n";
  }

  // 3. Diem danh kieu Hien dai (For-each) -> Khuyen khich!
  cout << "--- Cach 2: Quet tung cai ---\n";
  for (string x : xe_hoi) {
    cout << x << "\n";
  }

```

Kiểm tra tốt nghiệp (Final Exam)

Câu hỏi cuối cùng

Giả sử túi `xe_hoi` có **5** chiếc xe.

Anh dùng vòng lặp: `for (int i = 0; i < xe_hoi.size(); i++)`

1. Biến `i` sẽ bắt đầu từ số mấy?
2. Biến `i` sẽ kết thúc ở số mấy (số lớn nhất để lấy xe)?

Kiểm tra tốt nghiệp (Final Exam)

Câu hỏi cuối cùng

Giả sử túi `xe_hoi` có **5** chiếc xe.

Anh dùng vòng lặp: `for (int i = 0; i < xe_hoi.size(); i++)`

1. Biến `i` sẽ bắt đầu từ số mấy?
2. Biến `i` sẽ kết thúc ở số mấy (số lớn nhất để lấy xe)?

Đáp án

1. Bắt đầu từ **0**.
2. Kết thúc ở **4**.

(Vì kích thước là 5, các chỉ số hợp lệ là 0, 1, 2, 3, 4).

Tổng kết hành trình: Bùa hộ mệnh

Hành động	Code	Ví dụ đời thường
Mua túi	`vector<string> t;`	Mua balô rỗng.
Nhét đồ	`t.push_back("A");`	Nhét táo vào đáy.
Lấy đồ	`t[0]` / `t.at(0)`	Lấy món đầu tiên.
Đổi đồ	`t[0] = "B";`	Tráo táo thành lê.
Đếm đồ	`t.size()`	Xem túi nặng bao nhiêu.
Kiểm kê	`for (string x : t)`	Quét từng món.

Bước tiếp theo

Em đã tốt nghiệp khóa "Vector Cấp Tốc"! Em có muốn thử sức với bài tập thực hành nhỏ để test trình độ coding không?