Phân tích Chiến lược và Giải thuật

Hệ thống Bài toán Vector trên Codeforces (Rating 800-1000)

Slide Learning C++

Ngày 15 tháng 1 năm 2026

Tổng quan

Vai trò của Vector trong CP

- Quản lý bộ nhớ linh hoạt (Dynamic Arrays).
- Tư duy về chỉ số (`indexing`) và truy cập ngẫu nhiên.
- Nền tảng cho các thuật toán tham lam (`greedy`) và mô phỏng.

Phân khúc Rating 800-1000

Tập trung vào khả năng chuyển đổi logic thành mã nguồn (`implementation`) và tư duy `ad-hoc`.

2.1. Next Round (158A)

Link: codeforces.com/problem/158/A

Đề bài

Xác định số lượng thí sinh lọt vào vòng tiếp theo. Điều kiện: Điểm số phải dương (> 0) và lớn hơn hoặc bằng điểm của người ở vị trí thứ k.

Chiến thuật: Tư duy Vector

- Chuyển đổi 1-based index sang 0-based: Ngưỡng điểm là `a[k-1]`.
- Duyệt qua `vector<int> a`: Kiểm tra `a[i] >= a[k-1] && a[i] > 0`.
- **Lưu ý:** Dừng sớm nếu gặp phần tử vi phạm vì mảng đã sắp xếp giảm dần.

2.2. Way Too Long Words (71A)

Link: codeforces.com/problem/71/A

Đề bài

Viết tắt từ có độ dài > 10 ký tự: Giữ ký tự đầu, cuối và thay phần giữa bằng số lượng ký tự bị lược bỏ.

Chiến thuật: String as Vector

- Kiểm tra `s.length() > 10`.
- Sử dụng `s.front()`, `s.back()` và `s.length() - 2`.

2.2. Way Too Long Words (71A)

Link: codeforces.com/problem/71/A

Đề bài

Viết tắt từ có độ dài > 10 ký tự: Giữ ký tự đầu, cuối và thay phần giữa bằng số lượng ký tự bị lược bỏ.

Chiến thuật: String as Vector

- Kiểm tra `s.length() > 10`.
- Sử dụng `s.front()`, `s.back()` và `s.length() - 2`.
- **Kết quả:** `s[0] + to_string(s.length()-2) + s.back()`.

2.3. Team (231A)

Link: codeforces.com/problem/231/A

Đề bài

Đếm số bài toán mà ít nhất 2 trong 3 người chắc chắn về lời giải.

Chiến thuật: Row Sum

- Coi mỗi bài toán là một hàng trong ma trận hoặc một tập hợp 3 số.
- Tính `sum = p + v + t`. Nếu `sum >= 2`, tăng biến đếm.
- Có thể xử lý trực tiếp (`online`) mà không cần lưu vector 2 chiều.

2.4. Bit++ (282A)

Link: codeforces.com/problem/282/A

Đề bài

Thực hiện các lệnh ++ và - trên biến X (khởi tạo bằng 0).

Chiến thuật: Pattern Matching

- Thay vì so sánh cả chuỗi, chỉ cần kiểm tra ký tự ở giữa `s[1]`.
- Nếu `s[1] == '+'` thì `X++`, ngược lại `X--`.

2.5. Domino Piling (50A)

Link: codeforces.com/problem/50/A

Đề bài

Tìm số quân domino 2×1 tối đa có thể đặt vào bảng $M \times N$.

Tư duy toán học

- Không cần dùng mảng/vector để mô phỏng lưới.
- Mỗi quân chiếm 2 đơn vị diện tích.
- Công thức: $\lfloor \frac{M \times N}{2} \rfloor$. Trong C++: `(m * n) / 2`.

3.1. Beautiful Matrix (263A)

Link: codeforces.com/problem/263/A

Đề bài

Tìm số bước tối thiểu đưa số 1 về tâm $(2, 2)$ của ma trận 5×5.

Chiến thuật: Khoảng cách Manhattan

- Tìm tọa độ (r, c) của số 1.
- Kết quả: $|r - 2| + |c - 2|$ (với 0-based index).
- Sử dụng hàm `abs()` trong thư viện `<cmath>`.

3.3. Helpful Maths (339A)

Link: codeforces.com/problem/339/A

Đề bài

Sắp xếp lại chuỗi phép cộng các số 1, 2, 3 theo thứ tự tăng dần.

Chiến thuật: Parsing và Sorting

- Trích xuất các số vào `vector<int> numbers`.
- Dùng `std::sort(numbers.begin(), numbers.end())`.
- In lại kèm dấu +. Lưu ý không in dấu dư ở cuối.

4.2. Nearly Lucky Number (110A)

Link: codeforces.com/problem/110/A

Đề bài

Đếm số chữ số may mắn (4, 7). Kiểm tra xem **số lượng** đó có phải là số may mắn không.

Cạm bẫy

- Dữ liệu đầu vào cực lớn: Phải đọc bằng `string`.
- Đếm xong mới kiểm tra `count == 4 || count == 7`.

5.1. Numbers Box (1447B) - Rating 1000

Link: codeforces.com/problem/1447/B

Đề bài

Đổi dấu 2 ô kề nhau tùy ý. Tìm tổng lớn nhất có thể của ma trận.

Tư duy kiến thiết (Constructive)

- Dấu trừ có thể "di chuyển"tự do.
- Nếu số lượng số âm là **Chẵn**: Tổng = Tổng trị tuyệt đối.
- Nếu số lượng số âm là **Lẻ**: Phải để lại 1 số mang dấu âm. Chọn số có trị tuyệt đối nhỏ nhất (`min_abs`).

5.1. Numbers Box (1447B) - Rating 1000

Link: codeforces.com/problem/1447/B

Đề bài

Đổi dấu 2 ô kề nhau tùy ý. Tìm tổng lớn nhất có thể của ma trận.

Tư duy kiến thiết (Constructive)

- Dấu trừ có thể "di chuyển"tự do.
- Nếu số lượng số âm là **Chẵn**: Tổng = Tổng trị tuyệt đối.
- Nếu số lượng số âm là **Lẻ**: Phải để lại 1 số mang dấu âm. Chọn số có trị tuyệt đối nhỏ nhất (`min_abs`).
- **Công thức:** `total_abs_sum - 2 * min_abs`.

5.8. Twins (160A) - Rating 900

Link: codeforces.com/problem/160/A

Đề bài

Lấy ít đồng xu nhất sao cho tổng tiền của bạn lớn hơn tổng tiền còn lại.

Chiến thuật: Greedy

- Sắp xếp vector giảm dần: `sort(v.rbegin(), v.rend())`.
- Lấy dần các đồng xu lớn nhất.
- Dừng lại khi `my_sum > total_sum / 2`.

6.3. Cạm bẫy và Tối ưu hóa

Lưu ý kỹ thuật

- **Integer Overflow:** Dùng `long long` cho các bài tính tổng hoặc nhân (ví dụ: Soldier and Bananas).
- **Indexing:** Luôn cẩn thận với lỗi lệch 1 (Off-by-one).

Fast I/O trong C++

```
int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    // Code logic here
    return 0;
}

```

Lời kết

Cảm ơn bạn đã theo dõi!

Hãy luyện tập thường xuyên để biến tư duy Vector thành phản xạ.

Slide created for CP Learning Path.