Contest 01

Bài 1: Cấp xố cộng

Giải thuật: Ta chỉ cần sort lại mảng a rồi kiểm tra xem mảng a có phải cấp số cộng không. Để là cấp số cộng thì mảng a phải có ít hơn 3 phần tử hoặc phải thỏa điều kiện sau:

$$a[i] - a[i - 1] == a[2] - a[1]\ \forall i \geq 3$$

Độ phức tạp: O(NlogN)

Bài 2: Xếp chỗ

Dễ thấy bài toán này được quy về bài toán tính chỉnh hợp. Hoặc ta có thể làm như sau:

- Học sinh 1 có M cách chọn chỗ ngồi.
- Học sinh 2 có M – 1 cách chọn chỗ ngồi.
- …
- Học sinh N có M – N + 1 cách chọn chỗ ngồi,

Vậy đáp án là $\prod_{i=M-N+1}^{M} i = M(M-1)(M-2)...(M-N+1)$

Độ phức tạp: O(N)

Bài 3: Bội chung nhỏ nhất

Làm như cách thông thường:

- Phân tích tất cả các số thành thừa số nguyên tố.
- Lập tích các thừa số nguyên tố, với mỗi thừa số nguyên tố, ta chọn số mũ lớn nhất của nó. Tích này sẽ là BCNN của cả dãy số.

Giải thuật:

- Sử dụng một mảng cnt[i] để lưu số mũ lớn nhất của thừa số nguyên tố i.
- Ta phân tích từng số $a[i] = p_{i1}^{k1}.p_{i2}^{k2}...p_{ij}^{kj}$ rồi cực đại hóa $cnt[p_{i1}] = \max(cnt[p_{i1}], k1)$, $cnt[p_{i2}] = \max(cnt[p_{i2}], k2)$, …, $cnt[p_{ij}] = \max(cnt[p_{ij}], kj)$
- Cuối cùng, ta lấy tích các $i^{cnt[i]}$, đây là kết quả bài toán.
- Lưu ý: để phân tích thừa số trong O(log(a[i])), ta cần sử dụng sàng.

Độ phức tạp: O(Nlog(a[i]))

Bài 4: Đột nhập phòng thí nghiệm

Giải thuật:

- Đầu tiên, ta lưu lại mảng flag[][] có ý nghĩa flag[u][v] = flag[v][u] = 1 nếu không được lấy đồng thời chất u và v.

- Ở đây ta sẽ dùng giải thuật sinh nhị phân. Giả sử dãy nhị phân được lưu trong b[i] thì sẽ có ý nghĩa là: b[i] = 0 thì ta không lấy chất thứ i, ngược lại nếu b[i] = 1 thì ta lấy chất i.
- Với mỗi dãy nhị phân sinh ra, ta kiểm tra xem dãy có hợp lệ hay không, nói cách khác là kiểm tra xem có tồn tại cặp u, v nào có b[u] == 1, b[v] == 1 và flag[u][v] == 1. Nếu không tồn tại cặp nào như vậy, ta lấy tổng của các a[u] có b[u] == 1 rồi cập nhật biến kết quả.

Độ phức tạp: $O(2^N . N^2)$

Bài 5: Mạng lưới giao thông

Giải thuật:

- Hai đỉnh không thể đến được nhau là hai đỉnh thuộc hai thành phần liên thông khác nhau.
- Ta sử dụng DFS hoặc BFS để duyệt qua toàn bộ thành phần liên thông, với mỗi thành phần liên thông ta lưu lại số đỉnh của nó.
- Kết quả là $V[1]$ * (n - $V[1]$) $+$ $V[2]$ * (n - $V[2]$) $+ \ldots +$ $V[k]$ * (n - $V[k]$), với $V[i]$ là số đỉnh thuộc thành phần liên thông thứ i và k là số thành phần liên thông của đồ thị.

Độ phức tạp: O(max(N, M))

Contest 02

Bài 1: Số nguyên tố gần nhất

Giải thuật:

- Ta tìm L, R tương ứng là số nguyên tố gần N nhất và $L \leq N, R \geq N$
- Để tìm L, R ta chỉ việc giảm dần L và tăng dần R cho đến khi L, R là hai số nguyên tố. Để kiểm tra số nguyên tố, ta sử dụng giải thuật kiểm tra trong O(sqrt(N))
- In ra kết quả như yêu cầu đề bài.

Bài 2: Tổng đoạn lớn nhất

Giải thuật:

- Đầu tiên ta tính tổng tiền tố của mảng a và lưu vào mảng sum.
- Với mỗi i, ta sẽ tìm đoạn kết thúc tại i và có tổng lớn nhất. Nhận thấy, tổng đoạn từ j đến i $(j \leq i) = \mathrm{sum}[i] - \mathrm{sum}[j-1]$. Để đoạn kết thúc tại i có tổng lớn nhất thì ta phải tìm được sum[j – 1] nhỏ nhất.
- Vậy ta chỉ cần lưu một biến Min chứa sum[j] nhỏ nhất (j < i). Với mỗi i, ta cực đại hóa biến kết quả: $\mathrm{res} = \max(\mathrm{res}, \mathrm{sum}[i] - \mathrm{Min})$, cực tiểu hóa biến Min: $\mathrm{Min} = \min(\mathrm{Min}, \mathrm{sum}[i])$.

Độ phức tạp: O(N).

Bài 3: Số gây nghiện

- Yêu cầu là ta cần chia chuỗi thành các số nguyên sao cho chúng thỏa điều kiện đề bài.
- Sử dụng giải thuật quay lui, xây dựng hàm backtrack(i, pos) với ý nghĩa là ta đang chia đến số thứ i, vị trí bắt đầu của số thứ i là pos. Sử dụng một mảng nhớ memo với memo[i] lưu độ dài (số chữ số) của số thứ i.
- Với mỗi hàm backtrack(i, pos), ta duyệt qua mọi độ dài len có thể, gán memo[i] = len và tiếp tục backtrack(i + 1, pos + len).
- Nhánh cận: nhận thấy rằng nếu số thứ i phải bằng tổng số thứ i – 1 và i – 2 thì memo[i] >= memo[i – 1] và memo[i] >= memo[i – 2] với mọi i > 2. Vậy ta chỉ chọn len khi len thỏa điều kiện trên.
- Khi đã có một cách chia, ta chỉ việc kiểm tra xem các số chia được có thỏa mãn hay không.

Bài 4: Kiểm tra chu trình

- Ta duyệt qua mọi thành phần liên thông của đồ thị.
- Với mỗi thành phần liên thông, ta sử dụng DFS(u, p) với ý nghĩa là duyệt sâu từ đỉnh u với đỉnh liền trước là p.
  - Đồ thị có chu trình khi tồn tại v là một đỉnh kề với u, v khác p và v đã được duyệt trước u.

Độ phức tạp: O(max(N, M)).

Bài 5: Trồng hoa

Giải thuật:

- Nhận thấy, ta không được trồng hai hàng nào có 3 loại hoa giống hệt nhau. Vậy:
    - Nếu hàng thứ i ta chỉ trồng 1 hoặc 2 loại hoa thì sẽ có số cách trồng là C(1, N) và C(2, N).
    - Nếu hàng thứ i ta trồng 3 loại hoa thì số cách trồng là C(3, N) – pre với pre là số hàng đất đã trồng 3 loại hoa trước đó. (Ta không được trồng trùng 3 loại nên phải trừ đi số cách trồng 3 loại ở trước đó.)
- Ta thực hiện bằng giải thuật đệ quy có nhớ:
    - Xây dựng hàm solve (pos, pre) với ý nghĩa là số cách trồng các hàng đất từ pos đến n với pre là số hàng đã trồng 3 loại hoa từ 1 đến pos – 1.
    - Lưu một biến kết quả res, ban đầu res = 0.
    - Nếu ta trồng 1 loại hoa trên hàng đất pos thì res += C(1, N) * solve(pos + 1, pre)
    - Nếu ta trồng 2 loại hoa trên hàng đất pos thì res += C(2, N) * solve(pos + 1, pre)
    - Nếu ta trồng 3 loại hoa trên hàng đất pos thì res += (C(3, N) – pre) * solve(pos + 1, pre + 1).
    - Gán lại vào mảng nhớ f[pos][pre] = res

Độ phức tạp: O(M^2)