Post Lamps – Đèn đường

Nhận thấy là mỗi đèn chỉ chiếu sang bên phải, nên thuật toán tối ưu để dùng ít đèn nhất là cứ tìm vị trí trái nhất chưa được chiếu sáng và đặt đèn vào cái vị trí phải nhất có thể chiếu được vào vị trí đó, lặp lại cho đến khi xong thì dừng.

Nhận thấy là có x vị trí liên tiếp không đặt được đèn thì ta cần phải có đèn có sức chiếu $l > x$ thì mới chiếu được cái đoạn đó.

Để tiện xử lí thì ta sẽ tính trước xem là sức chiếu bé nhất có thể để chiếu được cả đoạn là bao nhiêu và chỉ kiểm tra các sức chiếu mà có thể chiếu được cả đoạn.

Để kiểm tra thì ta có thể sử dụng lại thuật toán tham lam đã miêu tả ở trên.

Cụ thể, với mỗi điểm ta tính trước cái điểm phải nhất không vượt quá nó mà có thể đặt đèn. Nếu có thông tin đó thì ta có thể mô phỏng thuật toán trong $O(step)$ với $step$ là số bước cần để chiếu sáng hết cả đoạn.

Tính điểm phải nhất không vượt quá thì đơn giản là điểm phải nhất không vượt quá ở bước trước nếu điểm này có vật cản hoặc là chính điểm này nếu như điểm này không có vật cản.

Độ phức tạp thực tế là $O\big(nlog(k)\big)$. Để chứng minh được thì ta sẽ chứng minh là với mỗi sức chiếu x, độ phức tạp của quá trình trên không quá $O\left(\frac{n}{x}\right)$. Việc chứng minh tổng kia là $O\big(nlog(k)\big)$ thì các bạn cũng đã biết.

- Giả sử ta đặt đèn với sức chiếu x ở một điểm a. Có hai trường hợp có thể xảy ra:
    - $a + x$ không có vật cản, vì vậy sau bước này thì đoạn ta chiếu sáng được tăng lên x.
    - $a + x$ có vật cản, vậy thì ở bước tiếp theo ta sẽ đặt đèn vào vị trí b sao cho $a < b < a + x$ (theo giả thiết là có thể chiếu sáng được).
        - Từ b đến $a + x$ sẽ không có điểm nào không có vật cản, vì thuật toán sẽ chọn b không có vật cản lớn nhất.

        - Vậy thì khi ta đặt đèn ở b, dù cho $b + x$ có vật cản hay không thì tiếp theo ta cũng đặt đèn một điểm $c > b$ mà không có vật cản.
        - Vì từ $b + 1$ đến $a + x$ không có ô nào không có vật cản nên $c > a + x$
    - Vậy thì trong bất kì trường hợp nào, sau 2 bước ta chắc chắn sẽ di chuyển điểm đặt đèn được x vị trí.
    - Vậy thì sau $O\left(\frac{n}{x}\right)$ thao tác, ta sẽ có kết quả.

https://codeforces.com/contest/990/submission/43391209

Việc nhận ra rằng độ phức tạp là $O(nlog(k))$ chắc là phần khó hơn của bài này. Trong những bài có tính tăng giảm như thế này thì thường nó vẫn thỏa mãn là sau một thời gian nhất định, xu hướng sẽ luôn tăng. Nếu cái thời gian này mà bé, ví dụ như $O(1)$ hay $O(log)$ thì ta vẫn có thể bruteforce được.

Sum of Remainders – Tổng các số dư

Đầu tiên nhận xét về phép chia: khi ta chia n cho a, được thương q là số dư r thì ta có đẳng thức sau:

$$n = a \times q + r (0 \leq r < a)$$

Thì nhận thấy là ta có $a \times q \leq n$ nên $\min(a, q) \leq \sqrt{n}$.

Vậy thì ta có ý tưởng chia căn là xử lí riêng riêng trường hợp $a \leq \sqrt{n}$ và $q \leq \sqrt{n}$.

Trường hợp $a \leq \sqrt{n}$ thì rất đơn giản, ta chỉ duyệt a, chia lấy dư rồi cộng vào kết quả.

Trường hợp $q \leq \sqrt{n}$ thì ta cần phải xử lí các yếu tố sau:

- Với các giá trị a nào thì sinh ra giá trị q như thế:
  - Phép chia nguyên thực tế là floor của phép chia thực, nên thực ra, $q = \left\lfloor \frac{n}{a} \right\rfloor$.
  - Vậy thì $\frac{n}{a} \geq q$ và $\frac{n}{a} < q + 1$ (Tính chất của hàm floor).
  - Vậy thì $a \leq n/q$ và $a > \frac{n}{q+1}$. Từ các tính chất này và giới hạn $1 \leq a \leq m$, ta có thể có được một đoạn liên tiếp các số $a \in [l_q, r_q]$ thỏa mãn $n = q \times a + r_a$. Với r_a là số dư khi chia cho a.
- Biết được các giá trị l_q, r_q thì tính tổng số dư trong đoạn này như thế nào?
  - Nếu ta cộng hai vế đẳng thức chia thì ta được:

    $$\sum_{a=l_q}^{r_q} n = \sum_{a=l_q}^{r_q} a \times q + r_a$$

    Hay:

    $$(r_q - l_q + 1) \times n - q \times \sum_{a=l_q}^{r_q} a = \sum_{a=l_q}^{r_q} r_a$$

- Vậy thì ta cần tính tổng các số a mà $l_q \leq a \leq r_q$ thì sẽ tính được tổng số dư. Tính tổng này đơn giản có thể dùng công thức tính tổng các số tự nhiên.

Vậy thì ta giải quyết được cả hai trường hợp trong $O(\sqrt{n})$, độ phức tạp là $O(\sqrt{n})$.

https://codeforces.com/contest/616/submission/44096653

Lưu ý: khi tính giá trị đầu cuối của a với thương q, không nên dùng chia số thực mà nên chia số nguyên rồi thử lại sẽ đảm bảo chính xác hơn.

Bài này thì điểm khó là nhìn ra khi số chia lớn lên thì số lượng thương là rất bé. Trên thực tế, chỉ có khoảng $2\sqrt{n}$ loại thương số khác nhau khi ta đem n đi chia có dư cho các số nguyên khác. Nếu các bạn gặp bài nào mà kiểu đếm tất cả các thương / tính tổng tất cả các thương khác nhau thì biết là nó không khó.

Earth Wind and Fire – Đất Gió và Lửa

Làm bài này thì chắc các bạn cũng đoán được là kết quả sẽ có kiểu đưa khối đá trái nhất về vị trí đá trái nhất, trái nhì về vị trí trái nhì, ...

Giả sử bây giờ ta đã sort lại tất cả các vị trí đích và vị trí đá, thì ta có thể chứng minh điều kiện trên như sau:

- Ta sẽ chứng minh rằng nếu tồn tịa cách ghép thì tồn tại cách mà ghép khối đá trái nhất với đích trái nhất. Nếu chứng minh được điều này thì ta có thể quy nạp chứng minh cho tất cả các khối còn lại.
- Khi ta thao tác vào bất kì hai khối đá nào thì $\min(s_i)$ không giảm và $\max(\mathrm{s_i})$ không tăng. Vì vậy nếu $\min(s_i) > \min(t_j)$ thì sẽ không có nghiệm.
- Vậy thì ta chỉ cần xét trường hợp mà $\min(\mathrm{s_i}) \leq \min(\mathrm{t}_j)$.
- Sớm hay muộn thì ta cũng cần kéo hết tất cả s_i có $s_i < t_j$ về $\min(\mathrm{t}_j)$ vì ta phải nối chúng với $t_k \geq \min(t_j)$.
- Vậy thì xem như ta đã kéo tất cả $s_i < \min(\mathrm{t_j})$ về $\min(t_j)$, ta phải giữ lại một tảng đá ở đây. Vì tất cả các tảng đá như nhau nên ta có thể chọn là sẽ giữ lại tảng có s_i bé nhất ban đầu.
- Vì quá trình trên bắt buộc phải làm nếu có nghiệm nên nếu có nghiệm thì ta có nghiệm sao cho ta ghép khối đá bé nhất với vị trí lớn nhất.
- Từ các khối sau, có thể có cả hai trường hợp đi từ trái sang phải và phải sang trái, tuy nhiên vẫn sẽ chứng minh tương tự được bằng cách là nếu ta di chuyển một khối đá đến vị trí đích bé nhất còn lại, ta sẽ di chuyển qua vị trí của khối trái nhất hoặc phải di chuyển khối trái nhất vào vị trí này, nên vẫn ghép được.

Vậy thì biết là với mỗi tảng đá thì nó cần di chuyển vào vị trí nào, làm sao kiểm tra được?

- Nhận thấy là có hai loại đá, di chuyển sang trái và di chuyển sang phải.
- Nếu chưa đạt được kết quả mà có kết quả, thì tồn tại một khối đá mà cần phải di chuyển sang bên phải (và một khối mà cần phải di chuyển sang bên trái).

- Vậy thì ta có thể có chiến thuật là ta sẽ di chuyển khối đá cần đi sang phải ở vị trí trái nhất mỗi lần.
- Mỗi khối đá đi sang phải thì phải được ghép với một khối đá đi sang trái.
- Để ghép được thì ta cần khối đi sang phải bé hơn khối đi sang trái.
- Vậy thì cách tốt nhất là ta ghép khối đi sang phải vị trí bé nhất với khối đi sang trái vị trí bé nhất, mỗi lần ta sẽ di chuyển sao cho một trong hai khối này đạt đến đích, nếu không di chuyển được thì là không làm được.
- Điều trên đúng là vì nếu ta ghép khối trái sang phải bé nhất với một khối khác thí sau này sẽ khó ghép khối phải sang trái bé nhất hơn vì tọa độ của khối trái sang phải luôn tăng.
- Kĩ càng hơn thì nhận xét trường hợp khối trái sang phải đến đích trước và phải sang trái đến đích trước.
    - Nếu trái sang phải đến đích trước thì khối phải sang trái vẫn là khối bé nhất, không có gì đặc biệt.
    - Nếu phải sang trái đến đích trước thì có nghĩa là khối trái sang phải chưa đến đích, do đó nếu nó có vượt qua khối nào khác cũng không sao vì không bị đổi thứ tự ghép.

https://codeforces.com/contest/1148/submission/54935136

Nhìn chung bài này là một bài tham lam khá phức tạp nếu chứng minh kĩ càng. Có thể dễ chứng minh sơ qua và tự tin là nó đúng. Trong các kì thi mà chấm online thì có thể làm thế được. Còn trong VOI thì thường chứng minh dễ hơn nhiều.

Việc nhận ra phải ghép đá theo thứ tự là phần cần thiết. Việc ghép thế nào thì có thể có nhiều cách để tham lam đúng được.