CHẶT NHỊ PHÂN KẾT QUẢ

Đề bài - Thuật toán - Code C++

1. Lý thuyết chung

Chặt nhị phân kết quả dùng khi đáp án nằm trong một khoảng [L, R] và ta viết được hàm check(x) trả về đúng/sai. Điều kiện quan trọng nhất là check(x) **phải đơn điệu**: khi x tăng thì trạng thái đúng/sai chỉ đổi một lần.

Dạng	**Dãy trạng thái**	**Cách cập nhật khi check(mid) đúng**
Tìm giá trị nhỏ nhất thỏa mãn	Sai, Sai, Đúng, Đúng	ans = mid, r = mid - 1
Tìm giá trị lớn nhất thỏa mãn	Đúng, Đúng, Sai, Sai	ans = mid, l = mid + 1

Độ phức tạp tổng quát: O(log(R-L+1) x chi phí của hàm check).

Bài 1. Chia sách cho học sinh

Mã bài: CHIASACH

Đề bài

Có n quyển sách đặt theo thứ tự từ 1 đến n. Quyển thứ i có a[i] trang. Cần chia n quyển sách thành k nhóm liên tiếp, mỗi nhóm giao cho một học sinh. Mỗi quyển sách phải thuộc đúng một nhóm. Hãy tìm số trang lớn nhất mà một học sinh phải đọc sao cho giá trị này là nhỏ nhất.

Dữ liệu vào

Dòng 1: n k. Dòng 2: n số nguyên a[1], a[2], ..., a[n].

Dữ liệu ra

In ra một số nguyên là số trang lớn nhất nhỏ nhất có thể.

Ví dụ

Input	Output
4 2 10 20 30 40	60

Nhận xét và thuật toán

Dạng tìm kiếm: Tìm nhỏ nhất thỏa mãn

Tính đơn điệu: Nếu x càng lớn thì càng dễ chia, vì giới hạn mỗi nhóm rộng hơn. check(x) có dạng Sai Sai Đúng Đúng. Ta tìm x nhỏ nhất thỏa mãn.

Miền tìm kiếm: L = max(a[i]), R = tổng a[i].

Hàm check: Duyệt từ trái sang phải, cộng dồn trang vào nhóm hiện tại. Nếu thêm a[i] làm tổng vượt x thì mở nhóm mới. Nếu số nhóm cần dùng không quá k thì x đạt.

Các bước: (1) Xác định [L,R]. (2) Viết check(mid). (3) Nếu tìm nhỏ nhất thỏa mãn thì check đúng sẽ đi về bên trái; nếu tìm lớn nhất thỏa mãn thì check đúng sẽ đi về bên phải. (4) In ans.

Độ phức tạp: O(n log M), trong đó M là độ rộng miền đáp án.

Code C++ tham khảo

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
cin.tie(nullptr);

    int n, k;     cin >> n >> k;
vector<long long> a(n);     long
long L = 0, R = 0;     for (int i
= 0; i < n; i++) {
        cin >> a[i];
```

```
        L = max(L, a[i]);
        R += a[i];
    }

    auto check = [&](long long x) {
        int groups = 1;          long
long sum = 0;          for (long long v
: a) {             if (sum + v <= x)
sum += v;
            else {
groups++;
sum = v;
            }
        }
        return groups <= k;
};

    long long ans = R;
while (L <= R) {
        long long mid = L + (R - L) / 2;
        if (check(mid)) {
ans = mid;              R
= mid - 1;
        } else {
            L = mid + 1;
        }
    }     cout
<< ans;
return 0; }
```

Bài 2. Koko ăn chuối

Mã bài: KOKO

Đề bài

Có n đống chuối, đống thứ i có a[i] quả. Mỗi giờ Koko chọn một đống và ăn tối đa x quả từ đống đó. Nếu đống còn ít hơn x quả thì ăn hết đống đó trong 1 giờ. Hãy tìm tốc độ ăn nhỏ nhất x để ăn hết tất cả chuối trong không quá H giờ.

Dữ liệu vào

Dòng 1: n H. Dòng 2: n số nguyên a[1], a[2], ..., a[n].

Dữ liệu ra

In ra tốc độ nhỏ nhất x.

Ví dụ

Input	Output
4 8 3 6 7 11	4

Nhận xét và thuật toán

Dạng tìm kiếm: Tìm nhỏ nhất thỏa mãn

Tính đơn điệu: x càng lớn thì tổng thời gian càng giảm hoặc không tăng. check(x) có dạng Sai Sai Đúng Đúng. Ta tìm x nhỏ nhất thỏa mãn.

Miền tìm kiếm: L = 1, R = max(a[i]).

Hàm check: Với tốc độ x, đống a[i] cần ceil(a[i] / x) giờ. Tổng thời gian không quá H thì x đạt. Trong C++ tính ceil(a[i]/x) bằng (a[i] + x - 1) / x.

Các bước: (1) Xác định [L,R]. (2) Viết check(mid). (3) Nếu tìm nhỏ nhất thỏa mãn thì check đúng sẽ đi về bên trái; nếu tìm lớn nhất thỏa mãn thì check đúng sẽ đi về bên phải. (4) In ans.

Độ phức tạp: O(n log M), trong đó M là độ rộng miền đáp án.

Code C++ tham khảo

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
cin.tie(nullptr);

    int n;     long long H;
cin >> n >> H;
vector<long long> a(n);
long long R = 0;
    for (int i = 0; i < n; i++) {
        cin >> a[i];
        R = max(R, a[i]);
    }

    auto check = [&](long long x) {
long long hours = 0;         for (long
long v : a) {             hours += (v +
x - 1) / x;             if (hours > H)
return false;
        }
        return hours <= H;
};

    long long L = 1, ans = R;
while (L <= R) {
        long long mid = L + (R - L) / 2;
        if (check(mid)) {
ans = mid;             R
= mid - 1;
        } else {
            L = mid + 1;
        }
    }     cout
<< ans;
return 0; }
```

Bài 3. Xếp bò xa nhau nhất

Mã bài: XEPBO

Đề bài

Có n chuồng bò nằm trên một trục thẳng, chuồng thứ i có vị trí x[i]. Cần đặt C con bò vào C chuồng khác nhau sao cho khoảng cách nhỏ nhất giữa hai con bò bất kỳ là lớn nhất. Hãy tìm giá trị khoảng cách lớn nhất đó.

Dữ liệu vào

Dòng 1: n C. Dòng 2: n số nguyên x[1], x[2], ..., x[n].

Dữ liệu ra

In ra khoảng cách nhỏ nhất lớn nhất có thể.

Ví dụ

Input	Output
5 3 1 2 4 8 9	3

Nhận xét và thuật toán

Dạng tìm kiếm: Tìm lớn nhất thỏa mãn

Tính đơn điệu: Nếu yêu cầu khoảng cách d càng lớn thì việc xếp bò càng khó. check(d) có dạng Đúng Đúng Sai Sai. Ta tìm d lớn nhất thỏa mãn.

Miền tìm kiếm: Sau khi sắp xếp vị trí: L = 0, R = x[n] - x[1].

Hàm check: Đặt bò đầu tiên ở chuồng trái nhất. Sau đó duyệt từ trái sang phải, gặp chuồng cách vị trí con bò vừa đặt ít nhất d thì đặt thêm một con. Nếu đặt được ít nhất C con thì d đạt.

Các bước: (1) Xác định [L,R]. (2) Viết check(mid). (3) Nếu tìm nhỏ nhất thỏa mãn thì check đúng sẽ đi về bên trái; nếu tìm lớn nhất thỏa mãn thì check đúng sẽ đi về bên phải. (4) In ans.

Độ phức tạp: O(n log M), trong đó M là độ rộng miền đáp án.

Code C++ tham khảo

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
cin.tie(nullptr);

    int n, C;     cin >> n
>> C;     vector<long long>
x(n);
```

```
    for (int i = 0; i < n; i++) cin >> x[i];
    sort(x.begin(), x.end());
    auto check = [&](long long d) {
        int cnt = 1;          long
long last = x[0];          for (int i
= 1; i < n; i++) {              if
(x[i] - last >= d) {
                cnt++;
last = x[i];
                if (cnt >= C) return true;
            }
        }
        return cnt >= C;
};

    long long L = 0, R = x.back() - x.front(), ans = 0;
    while (L <= R) {
        long long mid = L + (R - L) / 2;
        if (check(mid)) {
ans = mid;              L
= mid + 1;
        } else {
            R = mid - 1;
        }
    }     cout
<< ans;
return 0; }
```

Bài 4. Cắt dây

Mã bài: CATDAY

Đề bài

Có n sợi dây, sợi thứ i có độ dài nguyên a[i]. Cần cắt các sợi dây thành ít nhất k đoạn có cùng độ dài nguyên L. Có thể bỏ phần thừa. Hãy tìm L lớn nhất. Nếu không thể cắt được đoạn nào có độ dài dương thì in 0.

Dữ liệu vào

Dòng 1: n k. Dòng 2: n số nguyên a[1], a[2], ..., a[n].

Dữ liệu ra

In ra độ dài nguyên lớn nhất L.

Ví dụ

Input	Output
4 11 802 743 457 539	200

Nhận xét và thuật toán

Dạng tìm kiếm: Tìm lớn nhất thỏa mãn

Tính đơn điệu: Nếu L càng lớn thì số đoạn cắt được càng ít. check(L) có dạng Đúng Đúng Sai Sai. Ta tìm L lớn nhất thỏa mãn.

Miền tìm kiếm: L = 1, R = max(a[i]); ans ban đầu bằng 0 để xử lý trường hợp không cắt đủ k đoạn.

Hàm check: Với độ dài thử L, mỗi sợi a[i] tạo được floor(a[i] / L) đoạn. Nếu tổng số đoạn không nhỏ hơn k thì L đạt.

Các bước: (1) Xác định [L,R]. (2) Viết check(mid). (3) Nếu tìm nhỏ nhất thỏa mãn thì check đúng sẽ đi về bên trái; nếu tìm lớn nhất thỏa mãn thì check đúng sẽ đi về bên phải. (4) In ans.

Độ phức tạp: O(n log M), trong đó M là độ rộng miền đáp án.

Code C++ tham khảo

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
cin.tie(nullptr);

    int n;     long long k;
cin >> n >> k;     vector<long
long> a(n);     long long R = 0;
for (int i = 0; i < n; i++) {
cin >> a[i];
        R = max(R, a[i]);
    }

    auto check = [&](long long L) {
long long pieces = 0;         for
(long long v : a) {
pieces += v / L;
            if (pieces >= k) return true;
        }
        return pieces >= k;
};

    long long l = 1, r = R, ans = 0;
    while (l <= r) {
        long long mid = l + (r - l) / 2;
        if (check(mid)) {
ans = mid;             l
= mid + 1;
        } else {
r = mid - 1;
        }
    }     cout
<< ans;
return 0; }
```