

TỔNG QUAN ĐỀ THI

Bài	Tên bài	Tệp chương trình	Tệp dữ liệu	Tệp kết quả	Bộ nhớ (MB)	Thời gian (giây)	Điểm
1	Căn bậc hai	rgc.*	rgc.inp	rgc.out	1024	1	6
2	Bóng đá	bal.*	bal.inp	bal.out	1024	1	6
3	Số lớn nhất	max.*	max.inp	max.out	1024	1	5
4	Dãy hài hòa	seq.*	seq.inp	seq.out	1024	1	3

Dấu * được thay thế bởi cpp hoặc py của ngôn ngữ lập trình được sử dụng tương ứng là C++ hoặc Python.

Hãy lập trình giải các bài toán sau:

Bài 1. Căn bậc hai (6 điểm)

Hôm nay, lớp của An làm bài tập môn Toán về nội dung đưa thừa số ra ngoài dấu căn. Cô giáo giao bài tập về nhà là cho trước số nguyên dương n , hãy tìm hai số nguyên dương x, y sao cho $\sqrt{n} = x \times \sqrt{y}$ với x lớn nhất.

Ví dụ:

- Với $n=18$: do $\sqrt{18}=3 \times \sqrt{2}$ nên $x=3, y=2$;
- Với $n=7$: do $\sqrt{7}=1 \times \sqrt{7}$ nên $x=1, y=7$;
- Với $n=4$: do $\sqrt{4}=2 \times \sqrt{1}$ nên $x=2, y=1$.

Dữ liệu: Vào từ tệp văn bản rgc.inp gồm một dòng chứa số nguyên $(1 \leq n \leq 10^{14})$.

Kết quả: Ghi ra tệp văn bản rgc.out gồm một dòng chứa hai số nguyên x và y .

Ví dụ:

rgc.inp	rgc.out
18	3 2
7	1 7
4	2 1

Ràng buộc:

- Có 30% số test ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^5$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^9$;
- 40% số test còn lại ứng với 40% số điểm: Không có thêm ràng buộc nào.

Bài 2. Bóng đá (6 điểm)

Trong một giải đấu bóng đá có n đội bóng tham gia. Mỗi đội bóng phải đá với mỗi đội bóng khác đúng 2 trận lượt đi và về, trong đó có 1 trận trên sân nhà và 1 trận trên sân khách.

Mỗi đội bóng i có 2 trang phục thi đấu, 1 trang phục dành cho sân nhà có màu được biểu diễn bởi số nguyên x_i , còn 1 trang phục dành cho sân khách có màu được biểu diễn bởi số nguyên y_i ($x_i \neq y_i$). Các

trang phục cùng màu thì được biểu diễn bởi cùng một số và các trang phục khác màu thì được biểu diễn bởi các số khác nhau.

Trong mỗi trận đấu, một đội mặc trang phục sân nhà, một đội mặc trang phục sân khách. Tuy nhiên, nếu hai đội cùng màu trang phục thì đội khách được phép thay trang phục sân nhà của mình.

Hãy tính xem trong giải đấu, mỗi đội mặc bao nhiêu lần trang phục sân nhà và bao nhiêu lần trang phục sân khách.

Dữ liệu: Vào từ tệp văn bản `bal.inp`. Dòng đầu tiên chứa số nguyên $(2 \leq n \leq 10^5)$. Dòng thứ i trong n dòng tiếp theo chứa hai số nguyên x_i, y_i ($1 \leq x_i, y_i \leq 10^9; x_i \neq y_i$).

Kết quả: Ghi ra tệp văn bản `bal.out` gồm n dòng, trong đó dòng thứ i chứa hai số nguyên tương ứng là số lần mặc trang phục sân nhà và sân khách của đội bóng i .

Ví dụ:

<code>bal.inp</code>	<code>bal.out</code>
3	3 1
4 7	4 0
7 4	2 2
4 2	

Ràng buộc:

- Có 40% số test ứng với 40% số điểm thỏa mãn: $2 \leq n \leq 1000$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq x_i, y_i \leq 10^5$;
- 30% số test còn lại ứng với 30% số điểm: Không có thêm ràng buộc nào.

Bài 3. Số lớn nhất (5 điểm)

Cho hai số nguyên dương a và b . Hãy tìm số c lớn nhất chứa tất cả các chữ số của a và b , trong đó thứ tự xuất hiện các chữ số của a và b được giữ nguyên.

Dữ liệu: Vào từ tệp văn bản `max.inp`. Dòng đầu tiên chứa số nguyên dương a và dòng thứ hai chứa số nguyên dương b (a, b có không quá 1000 chữ số).

Kết quả: Ghi ra tệp văn bản `max.out` gồm một dòng chứa số c tìm được.

Ví dụ:

<code>max.inp</code>	<code>max.out</code>
20	421810
4181	

Ràng buộc:

- Có 40% số test ứng với 40% số điểm thỏa mãn: a có 1 chữ số;
- 30% số test khác ứng với 30% số điểm thỏa mãn: Không có chữ số nào của a xuất hiện trong b ;
- 30% số test còn lại ứng với 30% số điểm: Không có thêm ràng buộc nào.

Bài 4. Dãy hài hòa (3 điểm)

Quả sinh nhật của An là một bộ khối lập phương xếp hình. An xếp thành một dãy gồm n tháp, tháp thứ i có độ cao là a_i .

An rất có cảm tình với số nguyên k , vì vậy dãy liên tục các tháp được coi là **hài hòa** nếu chúng có độ cao trung bình là k .

Cho n, k và a_i với $i = 1, 2, \dots, n$. Hãy xác định dãy tháp hài hòa dài nhất: độ dài của dãy tìm được và số thứ tự của tháp đầu tiên. Nếu tồn tại nhiều dãy cùng độ dài thì chỉ ra dãy tháp có vị trí đầu nhỏ nhất. Nếu không tồn tại dãy tháp thì đưa ra một số 0.

Ví dụ với $n=5, k=3$ và độ cao của các tháp là 1,2,3,4,6. Dãy tháp hài hòa dài nhất là dãy các tháp từ vị trí thứ 2 đến 4, vì nó có độ cao trung bình là $\frac{2+3+4}{3}=3=k$ và không có dãy hài hòa nào có độ dài bằng hoặc lớn hơn nó. Dãy hài hòa này có độ dài bằng 3 và số thứ tự của tháp đầu tiên là 2.

Dữ liệu: Vào từ tệp văn bản `seq.inp`. Dòng đầu tiên chứa hai số nguyên n và $(1 \leq n \leq 10^5, 1 \leq k \leq 10^9)$. Dòng thứ hai chứa n số nguyên $a_1, a_2, \dots, a_n$.

Kết quả: Ghi ra tệp văn bản `seq.out` một dòng chứa hai số nguyên tương ứng là độ dài của dãy tìm được và số thứ tự của tháp đầu tiên hoặc một số 0 nếu không tồn tại dãy.

Ví dụ:

seq.inp	seq.out
5 3 1 2 3 4 6	3 2
3 2 2 1 3	3 1
4 3 1 2 5 6	0

Ràng buộc:

- Có 30% số test ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^2$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^3$;
- 40% số test còn lại ứng với 40% số điểm: Không có thêm ràng buộc nào.

----- HẾT -----

- *Thí sinh không được sử dụng tài liệu;*
- *Giám thị không giải thích gì thêm.*

Họ và tên thí sinh: Số báo danh:

**SỞ GIÁO DỤC VÀ ĐÀO TẠO
TỈNH QUẢNG NINH**

**HƯỚNG DẪN CHẤM THI CHỌN HỌC SINH GIỎI
CẤP TỈNH THCS**

Môn thi: **TIN HỌC** - Bảng A

ĐỀ THI MINH HỌA

(Hướng dẫn này có 06 trang)

I. HƯỚNG DẪN CHUNG

1. Giám khảo nghiên cứu đề, hướng dẫn chấm, test, code;
2. Chép bài của thí sinh vào máy chấm. Kiểm tra bài thí sinh trên giấy in và trên máy đảm bảo trùng khớp;
3. Thực hiện chấm bài trên máy chấm (bài thi của thí sinh được chấm trên máy tính bằng phần mềm chấm thi Themis, bản quyền của Cục Quản lý chất lượng);
4. Kiểm tra kết quả chấm bài (kiểm tra lại các bài có kết quả như không tồn tại bài, lỗi biên dịch, các bài bị 0 điểm, ...);
5. Tổng hợp kết quả chấm bài của thí sinh. Điểm bài thi được xuất từ phần mềm chấm thi, không quy tròn điểm của từng câu, điểm của bài thi;
6. Sau khi chấm bài xong, giám khảo lưu lại toàn bộ dữ liệu của chấm thi bằng phần mềm chấm thi Themis (Kỳ thi → Ghi kỳ thi), ghi ra đĩa CD và nộp cho hội đồng chấm thi để lưu.

II. ĐÁP ÁN, BIỂU ĐIỂM

Bài 1. Căn bậc hai (6 điểm)

Phân bổ điểm

- Có 30% số test ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^5$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^9$;
- 40% số test còn lại ứng với 40% số điểm: Không có thêm ràng buộc nào.

Cấu hình bài thi

Hướng dẫn giải

Subtask 1 (30%): $1 \leq n \leq 10^5$

Ta có $\sqrt{n} = x\sqrt{y} \Leftrightarrow n = x^2 \cdot y$.

Duyệt các giá trị y từ 1 đến n và kiểm tra n có chia hết cho y hay không và $\frac{n}{y}$ có là số chính phương

hay không. Nếu có thì ta tính $x = \sqrt{\frac{n}{y}}$.

Vì ta cần tìm x lớn nhất, tức là y nhỏ nhất, nên khi tìm được cặp (x, y) đầu tiên ta sẽ ghi ra kết quả và kết thúc chương trình.

Độ phức tạp thuật toán $O(n)$.

Subtask 2 (30%): $1 \leq n \leq 10^9$

Nhận thấy rằng $1 \leq x \leq \sqrt{n}$ và cần tìm x lớn nhất, do đó ta chỉ cần duyệt các giá trị x từ $\lfloor \sqrt{n} \rfloor$ về 1. Với mỗi giá trị x , ta cần kiểm tra n có chia hết cho x^2 hay không. Nếu có thì ta sẽ tính $y = \frac{n}{x^2}$. Khi tìm được cặp (x, y) đầu tiên ta sẽ ghi ra kết quả và kết thúc chương trình.

Trong subtask này, x và y chỉ dùng kiểu số nguyên $\int$.

Độ phức tạp thuật toán $O(\sqrt{n})$.

Subtask 3 (40%): Không có thêm ràng buộc nào

Thuật toán trong subtask 3 này giống với subtask 2, nhưng cần khai báo x, y kiểu số nguyên *long long*.

Bài 2. Bóng đá (6 điểm)

Phân bổ điểm

- Có 40% số test ứng với 40% số điểm thỏa mãn: $2 \leq n \leq 1000$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq x_i, y_i \leq 10^5$;
- 30% số test còn lại ứng với 30% số điểm: Không có thêm ràng buộc nào.

Cấu hình bài thi

Hướng dẫn giải

Subtask 1 (40%): $2 \leq n \leq 1000$

Gọi $cnt1[i], cnt2[i]$ lần lượt là số lần mặc trang phục sân nhà, sân khách của đội $(i=1, 2, \dots, n)$. Ban đầu ta khởi tạo $cnt1[i] = cnt2[i] = 0$.

Với mỗi trận đấu giữa đội nhà i và đội khách $(i=1, 2, \dots, n; j=1, 2, \dots, n; i \neq j)$:

- Đội i luôn mặc trang phục sân nhà. Do đó tăng số lần mặc trang phục sân nhà của đội i lên 1, tức là tăng $cnt1[i]$ lên 1;
- Kiểm tra đội j mặc trang phục sân khách hay sân nhà:
 - Nếu $y[j] \neq x[i]$ thì đội j cần mặc trang phục sân khách, tức là tăng $cnt2[j]$ lên 1.
 - Nếu $y[j] = x[i]$ thì đội j cần mặc trang phục sân nhà, tức là $cnt1[j]$ lên 1.

Độ phức tạp thuật toán là $O(n^2)$.

Subtask 2 (30%): $1 \leq x_i, y_i \leq 10^5$

Đầu tiên ta nhận thấy rằng mỗi đội i sẽ thi đấu $(n-1)$ trận sân nhà. Trong các trận này, đội i sẽ mặc trang phục sân nhà.

Mặt khác đội i sẽ thi đấu $(n-1)$ trận sân khách với các đội bóng j còn lại. Nếu trang phục sân khách của đội i trùng với trang phục sân nhà của đội j thì đội i sẽ mặc trang phục sân nhà. Do đó nếu ta gọi $cnt[c]$ là số đội bóng có trang phục sân nhà màu c thì số lần đội i mặc trang phục sân nhà khi thi đấu trên sân khách là $cnt[y[i]]$. Mảng cnt này cần được tính toán từ trước.

Vậy tổng số lần đội i mặc trang phục sân nhà là $(n-1) + cnt[y[i]]$.

Vì mỗi đội sẽ thi đấu $2(n-1)$ trận nên số lần mặc trang phục sân khách của đội i là:

$$2(n-1) - ((n-1) + cnt[y[i]]) = (n-1) - cnt[y[i]]$$

Trong subtask này, do $1 \leq x_i, y_i \leq 10^5$ nên ta có thể dùng biến cnt kiểu mảng.

Độ phức tạp thuật toán là $O(n)$.

Subtask 3 (30%): Không có thêm ràng buộc nào

Thuật toán trong subtask 3 này giống như subtask 2, nhưng có một điểm khác là do $1 \leq x_i, y_i \leq 10^9$ nên ta không thể dùng biến cnt kiểu mảng mà cần dùng kiểu *map* thay thế.

Độ phức tạp thuật toán là $O(n)$.

Bài 3. Số lớn nhất (5 điểm)**Phân bổ điểm**

- Có 40% số test ứng với 40% số điểm thỏa mãn: a có 1 chữ số;
- 30% số test khác ứng với 30% số điểm thỏa mãn: Không có chữ số nào của a xuất hiện trong b ;
- 30% số test còn lại ứng với 30% số điểm: Không có thêm ràng buộc nào.

Cấu hình bài thi
Hướng dẫn giải

Do các số a, b có tối đa 10^5 chữ số nên ta cần lưu trữ các số a, b bằng kiểu xâu.

Gọi m, n lần lượt là số chữ số của a, b .

Subtask 1 (40%): a có 1 chữ số

Ta tìm chữ số đầu tiên trong số b mà nhỏ hơn chữ số duy nhất của a . Sau đó chèn chữ số của số a vào

vị trí này trong số b . Trường hợp nếu không có chữ số nào trong b nhỏ hơn chữ số của a thì ta chèn chữ số của a vào cuối b .

Độ phức tạp thuật toán là $O(n)$.

Subtask 2 (30%): Không có chữ số nào của a xuất hiện trong b

Ta sử dụng hai con trỏ i, j để duyệt qua từng chữ số của số a, b .

Do không có chữ số nào của a xuất hiện trong b nên chỉ xảy ra hai trường hợp sau:

- $a[i] > b[j]$: Thêm chữ số $a[i]$ vào cuối số c hiện tại và tăng con trỏ i lên 1;
- $a[i] < b[j]$: Thêm chữ số $b[j]$ vào cuối số c hiện tại và tăng con trỏ j lên 1;

Việc duyệt trên sẽ dừng khi i hoặc j đến chữ số cuối cùng của a và b . Cuối cùng ta thêm các chữ số còn lại chưa được xét của a hoặc b vào cuối số c .

```
string a, b, c = "";
cin >> a >> b;
int m = a.size(), n = b.size(), i = 0, j = 0;
while (i < m && j < n) {
    if (a[i] > b[j])
        c += a[i++];
    else
        c += b[j++];
}
while (i < m)
    c += a[i++];
while (j < n)
    c += b[j++];
cout << c << "\n";
```

Độ phức tạp thuật toán là $O(m+n)$.

Subtask 3 (30%): Không có thêm ràng buộc nào

Ta sử dụng thuật toán đã trình bày trong subtask 2 nhưng bây giờ phải xét thêm trường hợp khi $a[i] = b[j]$. Khi đó ta cần tiếp tục so sánh các chữ số tiếp theo cho đến khi tìm thấy chữ số đầu tiên của a và b khác nhau hoặc đến khi một trong hai xâu kết thúc. Nếu một trong hai xâu có chữ số lớn hơn, thì thêm chữ số ở xâu đó vào kết quả. Ta có thể thay thế việc tìm kiếm này bằng so sánh xâu con của a từ vị trí i đến hết xâu và xâu con của b từ vị trí j đến hết xâu.

```
string a, b, c = "";
cin >> a >> b;
int m = a.size(), n = b.size(), i = 0, j = 0;
while (i < m && j < n) {
    if (a.substr(i) >= b.substr(j))
        c += a[i++];
    else
        c += b[j++];
}
while (i < m)
    c += a[i++];
while (j < n)
    c += b[j++];
cout << c << "\n";
```

Độ phức tạp thuật toán là $O(m^2 + n^2)$.

Bài 4. Dãy hài hòa (3 điểm)

Phân bổ điểm

- Có 30% số test ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^2$;
- 30% số test khác ứng với 30% số điểm thỏa mãn: $1 \leq n \leq 10^3$;
- 40% số test còn lại ứng với 40% số điểm: Không có thêm ràng buộc nào.

Cấu hình bài thi

Hướng dẫn giải

Subtask 1 (30%): $1 \leq n \leq 10^2$

Xét mọi dãy con gồm các tháp liên tiếp $a_l, a_{l+1}, \dots, a_r$ với $1 \leq l \leq r \leq n$.

Với mỗi dãy con, tính độ cao trung bình và nếu dãy con là hài hòa (độ cao trung bình bằng k) thì cập nhật lại dãy con hài hòa dài nhất và vị trí đầu tiên của dãy con.

```
int pos, mx = 0;
for (int l = 1; l <= n; l++)
    for (int r = l; r <= n; r++) {
        // xét dãy con a[l], ..., a[r]
        long long sum = 0;
        for (int i = l; i <= r; i++)
            sum += a[i];
        if (sum == 1LL * k * (r - l + 1) &&
            (r - l + 1 > mx || (r - l + 1 == mx && l < pos))) {
            pos = l;
            mx = r - l + 1;
        }
    }
if (mx > 0)
    cout << mx << " " << pos;
else
    cout << 0;
```

Độ phức tạp thuật toán là $O(n^3)$.

Subtask 2 (30%): $1 \leq n \leq 10^3$

Ta cải tiến thuật toán trong subtask 1.

Với mỗi dãy con, tính độ cao trung bình bằng kĩ thuật tiền xử lý tổng tiền tố. Gọi $s_i = a_1 + a_2 + \dots + a_i$, ta tính s_i bằng công thức truy hồi như sau:

- $s_0 = 0$;
- $s_i = s_{i-1} + a_i$ với $i = 1, 2, \dots, n$;

Khi đó tổng các số của dãy con $a_l, a_{l+1}, \dots, a_r$ là $s_r - s_{l-1}$. Dãy con này là hài hòa khi và chỉ khi:

$$\frac{s_r - s_{l-1}}{r - l + 1} = k$$

Một chú ý khi viết chương trình là phép chia trên có thể dẫn đến sai số, vì vậy ta viết điều kiện trên trở thành: $s_r - s_{l-1} = k(r-l+1)$.

```
long long s[100001];
s[0] = 0;
for (int i = 1; i <= n; i++)
    s[i] = s[i - 1] + a[i];
int pos, mx = 0;
for (int l = 1; l <= n; l++)
    for (int r = l; r <= n; r++) {
        long long sum = s[r] - s[l - 1];
        if (sum == 1LL * k * (r - l + 1) &&
            (r - l + 1 > mx || (r - l + 1 == mx && l < pos))) {
            pos = l;
            mx = r - l + 1;
        }
    }
if (mx > 0) cout << mx << " " << pos;
else cout << 0;
```

Độ phức tạp thuật toán là $O(n^2)$.

Subtask 3 (40%): Không có thêm ràng buộc nào

Giảm các phần tử a_i đi k đơn vị. Khi đó một dãy con là hài hòa khi và chỉ khi tổng tất cả các phần tử của dãy con bằng 0.

Tiếp theo, thay vì xét mọi dãy con, ta xét phần tử cuối cùng của dãy con. Giả sử chỉ số phần tử cuối cùng của dãy con là $(r=1, 2, \dots, n)$. Theo như nhận xét ở trên, các dãy con hài hòa có chỉ số phần tử cuối cùng là r , sẽ có chỉ số phần tử đầu tiên l ($1 \leq l \leq r$) thỏa mãn:

$$a_l + a_{l+1} + \dots + a_r = 0 \Leftrightarrow s_r - s_{l-1} = 0 \Leftrightarrow s_r = s_{l-1}$$

Hơn nữa ta cần tìm dãy hài hòa dài nhất, nên ta cần tìm chỉ số l nhỏ nhất ($1 \leq l \leq r$) mà $s_{l-1} = s_r$, tức là tìm tổng tiền tố đầu tiên có giá trị bằng s_r .

Để làm được việc này, ta sẽ xét r lần lượt từ 1 đến n , với mỗi r ta tính tổng $\sum_{i=1}^r a_i + a_2 + \dots + a_r$. Khi đó xảy ra hai trường hợp:

- Nếu trước đó chưa xuất hiện tổng tiền tố nào bằng $\sum_{i=1}^r a_i$ thì sẽ không có dãy hài hòa nào kết thúc tại r cả và ta ghi nhận tổng tiền tố $\sum_{i=1}^r a_i$ xuất hiện đầu tiên tại chỉ số r : $id[r]$;
- Nếu trước đó đã xuất hiện tổng tiền tố bằng $\sum_{i=1}^r a_i$ thì tồn tại dãy hài hòa dài nhất kết thúc tại r với chỉ số phần tử đầu tiên là $l = id[r]$ và có độ dài là $r - l + 1$.

Mặt khác chú ý rằng tổng $\sum_{i=1}^r a_i$ có thể rất lớn (có thể đạt tới 10^{14}) nên ta không thể cài đặt id bằng mảng được, mà cần cài đặt bằng cấu trúc dữ liệu map.

```
for (int i = 1; i <= n; i++)
    a[i] -= k;
int pos, mx = 0;
map<long long, int> id;
id[0] = 0;
long long sum = 0;
for (int r = 1; r <= n; r++) {
    sum += a[r];
    if (id.find(sum) != id.end()) {
        int l = id[sum] + 1, len = r - l + 1;
        if (len > mx || (len == mx && l < pos)) {
            pos = l;
            mx = len;
        }
    }
    else id[sum] = r;
}
```

```
if (mx > 0) cout << mx << " " << pos;  
else cout << 0;
```

Độ phức tạp thuật toán là $O(n \cdot \log_2 n)$.

----- HẾT -----