**Bài 1. Tìm số** *(5 điểm)*

Cho 2 số nguyên A và B ($2 \le B \le A \le 10^9$). Tìm số nguyên X lớn nhất sao cho $X \ne 1$, $X \le A$ và X không chia hết cho mọi số nguyên từ 2 đến B.

**Dữ liệu:** Vào từ file văn bản FINDN.INP một dòng chứa hai số nguyên dương B và A ($2 \le B \le A \le 10^9$).

**Kết quả:** Ghi ra file FINDN.OUT số nguyên X tìm được. Nếu không tìm được ghi ra -1.

**Ví dụ:**

| FINDN.INP | FINDN.OUT |
|---|---|
| 3 6 | 5 |

- Phân tích X thành các thừa số nguyên tố, nếu ước nguyên tố nhỏ nhất của X lớn hơn B thì X sẽ không chia hết cho các số nguyên từ 2 -> B.
- Để tìm số X lớn nhất, ta bắt đầu từ X = A, xét tính thỏa điều kiện của X. Nếu không thỏa thì giảm X đi 1 đơn vị và tiếp tục kiểm tra điều kiện. Ta sẽ lặp các thao tác này đến khi X thỏa hoặc $X \le B$. Do khoảng cách giữa hai số nguyên tố nhỏ hơn $10^9$ không quá 300 nên số lần lặp sẽ không quá 300 lần.

**Bài 2. Số đặc biệt (***4 điểm***)**

Cho dãy số A gồm n phần tử $a_1, a_2, \ldots, a_n$. Số đặc biệt là số có các chữ số đều giống nhau. Ví dụ, số 22, 666 là số đặc biệt còn số 123, 335 không là số đặc biệt.

**Yêu cầu:** Hãy đếm số cặp chỉ số (i, j) sao cho:

- $1 \le i < j \le n$
- $a_i + a_j$ là một số đặc biệt

**Dữ liệu:** Vào từ file văn bản CSPEC.INP

- Dòng đầu tiên gồm số nguyên dương n ($1 \le n \le 2 \times 10^5$) là số phần tử của dãy A;
- Dòng thứ hai gồm n số nguyên $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$).

**Kết quả:** Ghi ra file CSPEC.OUT số cặp chỉ số tìm được.

**Ràng buộc:**

- Có 50% số test ứng với 50% số điểm của bài thỏa mãn: $n \le 2000$;
- 50% số test ứng với 50% số điểm của bài không có ràng buộc gì thêm.

**Ví dụ:**

| CSPEC.INP | CSPEC.OUT | Giải thích |
|---|---|---|
| 3<br>13 9 2 | 2 | - Với i = 1, j = 2, ta có $a_1 + a_2 = 13 + 9 = 22$ là một số đặc biệt.<br>- Với i = 1, j = 3, ta có $a_1 + a_3 = 13 + 2 = 15$ không là một số đặc biệt.<br>- Với i = 2, j = 3, ta có $a_2 + a_3 = 9 + 2 = 11$ là một số đặc biệt. |

- Ta thấy số lượng số đặc biệt không lớn (có tổng cộng 55 số đặc biệt có giá trị không vượt quá $2 \times 10^6$).
- Khởi tạo mảng cnt với toàn bộ phần tử bằng 0.
  cnt[x] = số lượng phần tử có giá trị bằng x. Duyệt qua các phần tử của mảng a. Với mỗi phần tử $a_i$:
  - Duyệt qua từng số đặc biệt x, cộng thêm giá trị cnt[$x - a_i$] vào đáp án.
  - Tăng giá trị cnt[$a_i$] thêm 1.

Độ phức tạp: O(N) với N là số phần tử của dãy.

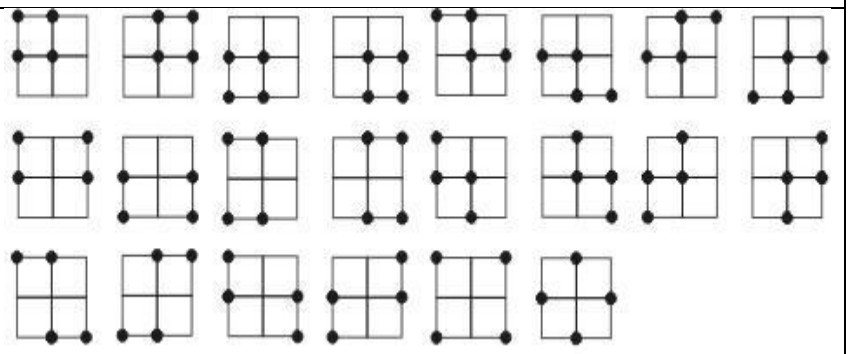
**Bài 3. Hình bình hành (*4 điểm*)**

Cho 2 số nguyên dương $m$ và $n$. Đếm số hình bình hành khác nhau mà tọa độ các đỉnh của nó có tung độ là số nguyên nằm trong đoạn $[0, m]$ và hoành độ là số nguyên nằm trong đoạn $[0, n]$.

**Dữ liệu:** Vào từ file văn bản CPARA.INP gồm một dòng duy nhất chứa 2 số nguyên dương $n, m$ ($0 < n, m \leq 2000$).

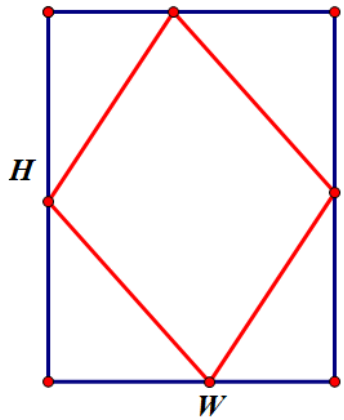
**Kết quả:** Ghi ra file văn bản CPARA.OUT một số nguyên duy nhất là đáp án bài toán.

**Ví dụ:**

| CPARA.INP | CPARA.OUT | Giải thích |
|---|---|---|
| 2 2 | 22 |  |

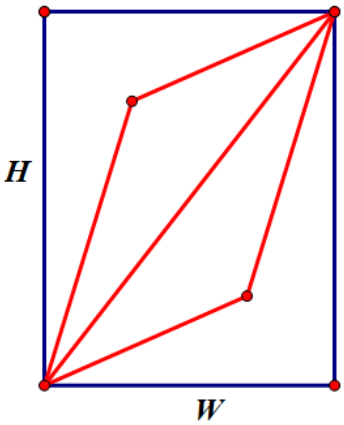
Ta đếm số hình bình hành trong hình chữ nhật có $H$ điểm dọc và $W$ điểm ngang. Có hai trường hợp:

- Trường hợp 1: Hình bình hành có 4 đỉnh nằm trên 4 cạnh của hình chữ nhật và không có đỉnh nào của hình bình hành là đỉnh của hình chữ nhật.




Dễ thấy vì 2 đỉnh liên tiếp của hình bình hành đặt trên hai cạnh bên của hình chữ nhật (hai điểm còn lại lấy đối xứng qua tâm hình chữ nhật, từ đây chúng ta có $(H-2)*(W-2)$ hình bình hành.

- Trường hợp 2: Hình bình hành có hai đỉnh là hai đỉnh của một đường chéo hình chữ nhật.




Nếu chúng ta đặt 2 đỉnh đối của hình bình hành là hai đỉnh của một đường chéo hình chữ nhật thì hai đỉnh khác là hai điểm nguyên bất kỳ đối xứng với tâm của hình chữ nhật, từ đó ta có $H*W - GCD(H-1, W-1) - 2$ hình bình hành.

Do đó, chúng ta đến với các thuật toán sau đây:

```
long long res = 0, tg = 0 ;
for (int h = 2; h <= N + 1; h++) {
    for (int w = 2; w <= M + 1; w++) {
        tg = (h - 2) * (w - 2) + h * w - GCD(h - 1, w - 1) - 2;
        res += (N - h + 2) * (M - w + 2) * tg;
    }
}
```

Thuật toán này có độ phức tạp của $O(N * M\log N)$.

Ta có thể tính toán trước một ma trận GCD [X] [Y], $1 \le x \le N$ và $1 \le y \le M$, trong $O(N * M)$. Như vậy, độ phức tạp đã được giảm xuống còn $O(N * M)$.

Chúng ta cũng có thể lưu ý rằng trong một hình chữ nhật H * W có thể là cùng một số hình chữ nhật như một hình chữ nhật W * H . Thủ thuật này gần như tăng gấp đôi tốc độ của thuật toán.

**Bài 4. Trò chơi Robot** *(4 điểm)*

Trong một trò chơi, có n ngọn núi được đánh số từ 1 đến $n$. Các ngọn núi được nối với nhau bởi $m$ con đường hai chiều, mỗi con đường nối trực tiếp hai ngọn núi, đảm bảo luôn có đường đi lại giữa hai ngọn núi bất kì (trực tiếp hoặc đi qua một số ngọn núi khác). Giữa hai ngọn núi bất kì không có quá một con đường nối trực tiếp. Có $b$ kho báu, mỗi kho báu nằm ở một ngọn núi khác nhau. Có $r$ Robot được đặt ở $r$ ngọn núi khác nhau.

**Yêu cầu:** Hãy tính số con đường ít nhất cần đi của mỗi Robot để đến được một kho báu bất kì.

**Dữ liệu:** Vào từ file văn bản ROBOT.INP

- Dòng thứ nhất gồm bốn số nguyên: $n, m, b, r$ ($2 \le n \le 5 \times 10^5$; $1 \le m \le 5 \times 10^5$ ; $1 \le b, r \le n$).
- Dòng thứ hai gồm $b$ số nguyên là chỉ số của các ngọn núi có kho báu.
- Dòng thứ ba gồm $r$ số nguyên là chỉ số của các ngọn núi có Robot.
- m dòng tiếp theo, mỗi dòng gồm hai số nguyên $u$ và $v$ thể hiện có một con đường hai chiều nối trực tiếp hai ngọn núi $u$ và $v$.

**Kết quả:** Ghi ra file văn bản ROBOT.OUT $r$ số nguyên trên cùng một dòng là kết quả tính được của các Robot đặt ở các ngọn núi theo thứ tự của dữ liệu.

**Ràng buộc:**

- Có 40% số test ứng với 40% số điểm của bài thỏa mãn: $2 \le n \le 2000$, $1 \le m \le 2000$;
- 60% số test ứng với 60% số điểm của bài không có ràng buộc gì thêm.

**Ví dụ:**

| ROBOT.INP | ROBOT.OUT | Giải thích |
|---|---|---|
| 6 6 2 3<br>3 2<br>1 5 4<br>1 2<br>1 6<br>3 6<br>2 3<br>4 5<br>3 4 | 1 2 1 | - Robot đặt ở ngọn núi 1: 1 → 2<br>- Robot đặt ở ngọn núi 4: 4 → 3<br>- Robot đặt ở ngọn núi 5: 5 → 4 → 3 |

Coi mỗi ngọn núi có robot là đỉnh đỏ, ngọn núi có kho báu là đỉnh xanh.

Sub 1: $2 \le n \le 2000$, $1 \le m \le 2000$ Với mỗi đỉnh đỏ, ta thực hiện loang (BFS) để tìm đỉnh xanh gần nhất. Độ phức tạp: $O(n * (n + m))$.

Sub 2: $2 \le n \le 5.10^5$ , $1 \le m \le 5.10^5$. Ta có thể gộp các đỉnh xanh thành một tập hợp, và xem đó như một đỉnh. Khoảng cách ngắn nhất từ một đỉnh đỏ tới đỉnh xanh bất kì cũng có thể hiểu là khoảng cách ngắn nhất từ tập hợp các đỉnh xanh đến đỉnh đỏ đó. Vì vậy, để giảm độ phức tạp, ta sẽ tính khoảng cách ngắn

nhất từ tập hợp các đỉnh xanh đến các đỉnh đỏ. Điều này có thể thực hiện bằng cách đưa tất cả các đỉnh xanh vào queue khi thực hiện thuật toán loang (BFS). Độ phức tạp: O(n + m)

**Bài 5. Lật bánh** *(3 điểm)*

An rất thích ăn bánh rán. Anh ấy đến cửa hàng mua bánh làm sẵn rồi về rán chín lên. Để bánh chín, bánh cần được rán trong chảo ở lửa vừa phải trong chính xác 2×N giây. N giây rán một mặt và N giây rán mặt còn lại. An thấy rằng mình có thể không lật được chiếc bánh chính xác N giây sau khi bắt đầu rán.

Có K khoảng thời gian mà An có thể lật bánh. Khoảng thời gian thứ $i$, bắt đầu từ thời điểm $l_i$ giây kể từ khi bắt đầu rán đến $r_i$ giây. An thấy không cần thiết phải lật chính xác một lần chiếc bánh khi đang rán. Thay vào đó, anh sẽ lật nó nhiều lần để chiếc bánh được rán đúng N giây ở một mặt và N giây ở mặt còn lại.

**Yêu cầu:** Tính xem An có thể hoàn thành kế hoạch của mình không? Nếu An chỉ có thể lật chiếc bánh rán trong khoảng thời gian đã định, thì số lần tối thiểu An sẽ phải lật chiếc bánh là bao nhiêu?

**Dữ liệu:** Vào từ file văn bản LATBANH.INP

- Dòng đầu tiên chứa hai số nguyên N và K ($1 \le N \le 5 \times 10^5$ , $1 \le K \le 100$)
- K dòng tiếp theo chứa mỗi dòng chứa hai số nguyên $l_i$ và $r_i$ ($0 \le l_i \le r_i \le 2 \times N$), điều đó có nghĩa là An có thể lật bất cứ lúc nào trong khoảng thời gian [$l_i$ , $r_i$ ]. Dữ liệu đảm bảo rằng $l_i > r_i - 1$ với $2 \le i \le k$.

**Kết quả:** Ghi ra file văn bản LATBANH.OUT số lần lật ít nhất cần thực hiện. Nếu không thể thì in ra -1

**Ràng buộc:**

- Có 30% số test ứng với 30% số điểm của bài thỏa mãn: $N \le 10$;
- 20% số test ứng với 20% số điểm của bài thỏa mãn: có kết quả không quá 2 lần lật;
- 20% số test ứng với 20% số điểm của bài thỏa mãn: $N \le 1000$;
- 20% số test ứng với 20% số điểm của bài thỏa mãn: $N \le 10^5$;
- 20% số test ứng với 20% số điểm của bài không có ràng buộc gì thêm.

**Ví dụ:**

| LATBANH.INP | LATBANH.OUT | Giải thích |
|---|---|---|
| 10 2<br>3 5<br>11 13 | 2 | - Lật lần 1 ở giây thứ 3<br>- Lật lần 2 ở giây thứ 13 |
| 10 3<br>3 5<br>9 10<br>11 13 | 1 | - Lật 1 lần ở giây thứ 10 |

- Sub 1: Duyệt $2^{2N}$ .
- Sub 2: Kiểm tra trường hợp kết quả = 1 nếu thời điểm N có khoảng lật, với kết quả = 2 thì tìm 2 đoạn mà có chênh lệch trong khoảng N bằng cách duyệt $K^2$.
- Sub3: Quy hoạch động dp[i][j] = số lần lật ít nhất đến thời điểm i và mặt trên đã được rán j giây:
  - Đến thời điểm i nếu không lật bánh thì dp[i][j] = dp[i − 1][j]
  - Nếu có thể lật bánh thì dp[i][j] = dp[i − 1][i − j] + 1
- Sub 4, 5: ta có nhận xét rằng trong một khoảng thời gian có thể lật từ [$l_i$, $r_i$] thì tối đa ta chỉ lật 2 lần, do đó ta chỉ cần quan tâm đến các thời điểm cuối cùng của mỗi khoảng.

  Gọi dp[i][j] = số lần lật ít nhất đến thời điểm $r_i$ và mặt trên đã được rán j giây:
  - dp[i][j] = dp[i − 1][j] nếu không lật
  - dp[i][j] = $\min_{l_i \le t \le r_i}$ (dp[i − 1][t − j]) + 1 khi dùng 1 lật
  - dp[i][j] = $min_{j-(r_i-l_i) \le j' < j}$(dp[i − 1][$j'$]) + 2 khi dùng 2 lật

Ta sẽ dùng deque để tính nhanh được dp[i][j] với độ phức tạp O(N.K). Một số cách dp khác có thể sẽ cần dùng cấu trúc cây để tính nhanh thì thuật toán là O(N.K. log N) thì chỉ qua được subtask 4.