A – SỐ CÔ ĐƠN

Time limit: 2s
Memory limit: 512MB

Đây là một bài rất dễ với giới hạn cho phép chúng ta làm bài này với rất nhiều cách khác nhau. Đơn giản nhất là các bạn hãy tạo một mảng $cnt[]$ với $cnt[x]$ là số lượng giá trị x có trong mảng:

```
for i := 1 to n do inc(cnt[a[i]]);
```

Như vậy giá trị số cô đơn là giá trị x có $cnt[x]$ là số lẻ, vì số lẻ thì khi chia cặp sẽ có một số x không có cặp.

Ngoài ra để giải bài các bạn cũng có thể sử dụng thuật toán sắp xếp để đưa các giá trị trong mảng về gần nhau hoặc dùng một cách rất thông minh là dùng toán tử bitwise xor. Với toán tử này ta có tính chất như sau: $\begin{cases} x \text{ xor } x = 0 \\ x \text{ xor } 0 = x \end{cases}$. Như vậy nếu ta thực hiện phép xor lên tất cả phần tử trong mảng (A_1 xor A_2 xor ... A_N) thì ta sẽ có được giá trị của phép tính chính là số cô đơn cần tìm vì tất cả những cặp số giống nhau sau khi thực hiện phép xor đều bằng 0.

Note: Phép xor trong Pascal được ký hiệu là "`xor`" còn trong C++ được ký hiệu là "^".

B – TRÒ CHƠI ĐIỆN TỬ

Time limit: 2s

Memory limit: 512MB

Đầu tiên để tránh bị lúng túng với dữ kiện "lập một chiến thuật ném bom sao cho tốn ít lần ném bom nhất nhưng vẫn đảm bảo phá hủy được hết tất cả xe tăng **trong bất kỳ tình huống nào**", ta cần xác định là chỉ cần lập được chiến thuật cho trường hợp xấu nhất là được. Như vậy ở trường hợp xấu nhất, ta chỉ cần cho rằng trong tất cả N ô đều có 2 xe tăng. 2 xe tăng này tượng trưng cho việc khi ném bom xuống nó sẽ di chuyển sang 1 trong 2 ô liền kề, bởi vì có nhiều xe tăng hơn thì cũng chỉ di chuyển về 1 trong 2 hướng đó.

Từ đây ta có thể giải quyết tình huống này như sau: Đầu tiên ném bom vào các ô **chẵn**, các xe tăng trong các ô này dù có di chuyển hướng này thì cũng chỉ di chuyển sang các ô **lẻ**. Tiếp theo ném bom vào các ô **lẻ**, những xe tăng trước đó di chuyển từ các ô **chẵn** sang sẽ bị phá hủy còn những xe tăng ban đầu trong các ô này sẽ di chuyển sang các ô **chẵn**. Cuối cùng là ném bom vào các ô **chẵn** để phá hủy các xe tăng vừa di chuyển từ các ô **lẻ** sang.

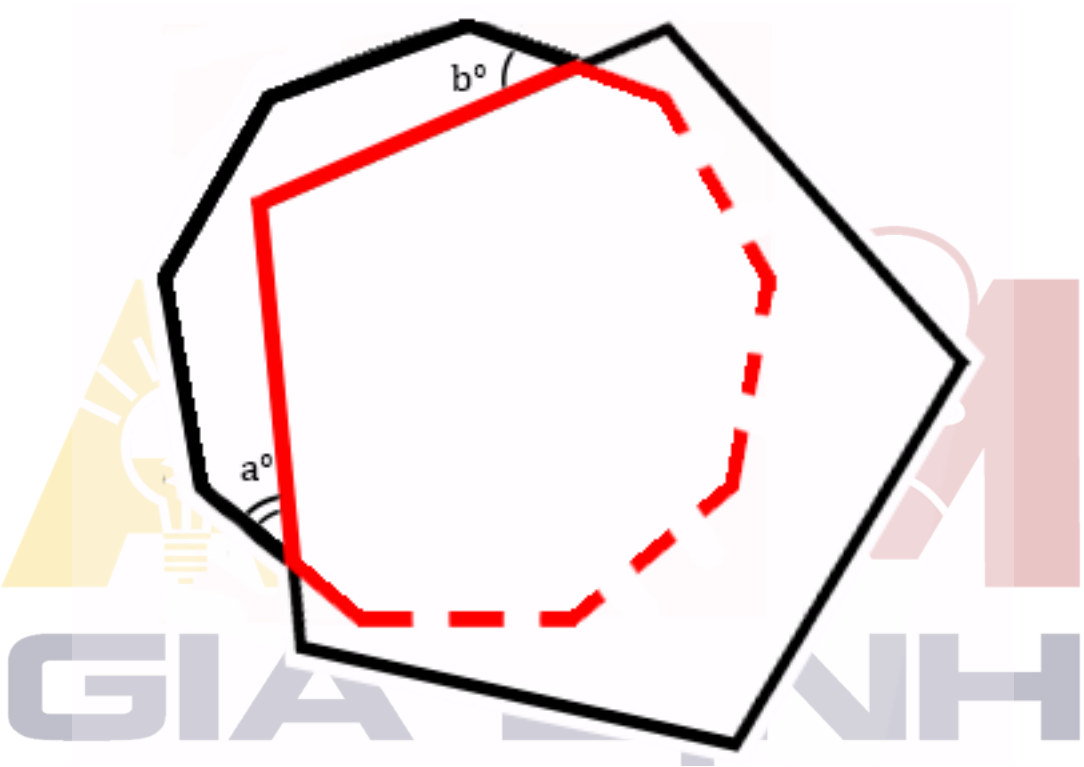
s

C – ĐA GIÁC "THỤ"

Time limit: 2s

Memory limit: 512MB

Để có thể khai thác tối đa mọi dữ kiện mà đề bài cho, ta cần biết tổng góc trong một đa giác x đỉnh là $180(x-2)$. Như vậy, số đo một góc trong một đa giác đều x đỉnh là $\frac{180(x-2)}{x}$. Từ đó ta sẽ lập một phương trình 1 ẩn dựa vào phần đa giác "bị giấu đi" (được tô màu đỏ) như hình:



Ta có đa giác màu đỏ gồm 1 đỉnh của đa giác công, 2 đỉnh được tạo thành do cạnh của đa giác công cắt cạnh của đa giác thụ. Vì đa giác thụ có K cạnh hiện ra, ta suy ra được đa giác thụ có $K-1$ đỉnh không bị che và do đó số đỉnh bị che (tức thuộc đa giác đỏ) là $M-K+1$.

Về số đo, ta có 1 đỉnh của đa giác công có số đo là $\frac{180(N-2)}{N}$, 2 đỉnh được tạo thành do cạnh của đa giác công cắt cạnh của đa giác thụ là $180-a$ và $180-b$ tương ứng tổng số đo 2 góc đó là $360-\frac{X}{Y}$, các đỉnh của đa giác thụ bị che có số đo là $\frac{180(M-2)}{M}$ nên ta có tổng số đo của các đỉnh này là $\frac{180(M-2)}{M} \times (M-K+1)$.

Kết hợp các dữ kiện trên, ta có phương trình tổng số đo trong đa giác đỏ phải bằng với tổng số đo các thành phần kể trên (tổng số đỉnh trong đa giác đỏ là $3+M-K+1=M-K+4$):

$$180(M-K+2)=\frac{180(N-2)}{N}+360-\frac{X}{Y}+\frac{180(M-2)}{M}\times(M-K+1)$$
$$\Leftrightarrow M-K+2=\frac{N-2}{N}+2-\frac{X}{180Y}+\frac{M-2}{M}\times(M-K+1)$$
$$\Leftrightarrow M-K+2=\frac{180Y(N-2)+360YN-XN}{180YN}+\frac{(M-2)(M-K+1)}{M}$$
$$\Leftrightarrow M-K+2=\frac{540YN-360Y-XN}{180YN}+\frac{M^2-M(K+1)+2K-2}{M}$$
$$\Leftrightarrow M^2-M(K-2)=M^2+M\left(\frac{540YN-360Y-XN}{180YN}-K-1\right)+2K-2$$
$$\Leftrightarrow M\left(K-2+\frac{540YN-360Y-XN}{180YN}-K-1\right)+2K-2=0$$
$$\Leftrightarrow M\left(\frac{540YN-360Y-XN}{180YN}-3\right)=2-2K$$
$$\Leftrightarrow M\times\frac{540YN-360Y-XN-540YN}{180YN}=2-2K$$
$$\Leftrightarrow M\times\frac{360Y+XN}{-180YN}=2-2K$$
$$\Leftrightarrow M=\frac{180YN(2K-2)}{360Y+XN}=\frac{360YN(K-1)}{360Y+XN}$$

Vậy ta có phương trình với một ẩn M duy nhất, giải phương trình trên là ta sẽ có kết quả.

D – BẤT PHƯƠNG TRÌNH BẤT BÌNH THƯỜNG

Time limt: 2s

Memory limit: 512MB

Để giải được bài này thì hướng đi thật ra rất đơn giản, chúng ta sẽ không tìm một lời giải cụ thể cho một bộ dữ liệu riêng biệt nào mà chúng ta sẽ tìm lời giải cho mọi bộ dữ liệu!

Việc đầu tiên cần làm là xác định trường hợp vô nghiệm. Ta có $S(x)$ sẽ không thể nào bằng 0 được vì cặp số a, b được chọn phải là số nguyên dương. Như vậy trường hợp vô nghiệm duy nhất của chúng ta là $M = 0$. Để giải thích vì sao là trường hợp duy nhất, hãy tiếp tục với cách giải dưới đây.

Vì chúng ta đang hướng đến một nghiệm cho tất cả mọi trường hợp, vì vậy ta sẽ xét đến trường hợp khó khăn nhất: $N = 50000$ và $M = 1$. Để giải quyết trường hợp này ta sẽ cần đến 2 số rất lớn sao cho tổng chữ số của từng số lớn hơn N và tổng 2 số sẽ là một lũy thừa của 10.

Khi tổng của 2 số là một lũy thừa của 10, tổng kết quả sẽ có dạng số 1 ở đầu và theo sau là rất nhiều số 0, khi đó tổng chữ số của nó chỉ là 1.

Nếu lấy N chia cho giới hạn tối đa số lượng chữ số (lưu ý rằng nghiệm của chúng ta có dưới 15000 chữ số, vì vậy tối đa là 14999 chữ số, đây là một trick của đề bài mặc dù BTC đã cố tình in đậm để giúp các bạn một phần) ta sẽ thấy rằng nếu mỗi chữ số có giá trị từ 4 trở lên thì tổng chữ số của số đó chắc chắn sẽ lớn hơn N ($4 \times 14999 > N$). Như vậy thật ra lúc này bài toán còn lại rất đơn giản, ta sẽ chọn cặp số a, b có dạng $555 \ldots 5$ (14999 chữ số 5) và $444 \ldots 5$ (14998 chữ số 4 và cuối cùng là 1 chữ số 5).

E – TỔNG DIỆN TÍCH

Time limit: 2s
Memory limit: 512MB

Để giải bài này chúng ta chỉ cần chăm chỉ phát triển công thức. Mỗi cặp đường thẳng dọc kết hợp với mỗi cặp đường thẳng ngang sẽ tạo thành một hình chữ nhật, gọi x_a, x_b lần lượt là 2 đường thẳng dọc được chọn và y_c, y_d lần lượt là 2 đường thẳng ngang được chọn. Để tránh việc chọn một cặp đường thẳng nhiều lần, ta quy định $x_a < x_b$ và $y_c < y_d$. Như vậy với hình chữ nhật tạo thành từ các đường thẳng đã chọn ta có công thức tính diện tích là $(x_b - x_a)(y_d - y_c)$.

Bây giờ giả sử ta cố định cặp đường thẳng dọc x_a, x_b, ta sẽ có tổng các hình chữ nhật được tạo thành từ cặp đường thẳng dọc x_a, x_b và tất cả các cặp đường thẳng ngang khác là:

$$(x_b - x_a)(y_2 - y_1) + \cdots + (x_b - x_a)(y_M - y_1) + \cdots + (x_b - x_a)(y_M - y_{M-1})$$

$$\Leftrightarrow \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} (x_b - x_a)(y_j - y_i)$$

$$\Leftrightarrow (x_b - x_a) \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} y_j - y_i$$

$$\Leftrightarrow (x_b - x_a) \sum_{i=1}^{M} (i-1) \times y_i - (M-i) \times y_i$$

$$\Leftrightarrow (x_b - x_a) \sum_{i=1}^{M} (2i - M - 1) \times y_i.$$

Và vì với mỗi cặp x_a, x_b ta đều kết hợp được với tất cả cặp đường thẳng ngang bất kỳ, vì vậy đặt $\sum_{i=1}^{M}(2i - M - 1) \times y_i$ làm phần tử chung và giải tương tự ta sẽ có công thức cuối cùng là

$$\sum_{i=1}^{N}\left((2i - N - 1) \times x_i\right) \times \sum_{j=1}^{M}\left((2j - M - 1) \times y_j\right).$$

Tới đây, để chạy được trong thời gian cho phép, ta chỉ cần dùng 1 vòng lặp for để tính $S_x = \sum_{i=1}^{N}\left((2i - N - 1) \times x_i\right)$ và dùng 1 vòng lặp for khác để tính $S_y = \sum_{j=1}^{M}\left((2j - M - 1) \times y_j\right)$ sau đó tính kết quả cuối cùng là $S_x \times S_y$.

F – PHÁC HỌA

Time limt: 2s

Memory limit: 512MB

Trong trường hợp tốt nhất, tức có nhiều người đồng thời có cả 3 đặc điểm nhất, thì ta dễ dàng tính được số lượng đó là $\min(A, B, C)$, vậy số lượng người cần phải gặp trước khi chắc chắn tìm được người phù hợp là $N - \min(A, B, C)$.

Trong trường hợp xấu nhất, tức có ít người đồng thời có cả 3 đặc điểm nhất, ta sẽ xét như sau:

Đầu tiên xét ta chắc chắn sẽ có A người là nam trong tổng số N người:

Tiếp theo vì là trường hợp xấu nhất, B người đẹp trai ta sẽ ưu tiên những người không phải là nam trước (ờm thì… nữ cũng đẹp trai được mà :v), sau đó nếu còn dư thì bắt buộc những người là nam trước đó phải đẹp trai rồi:

Cuối cùng, tương tự ở trên, C người cao to ta sẽ ưu tiên những người chỉ là nam hoặc chỉ đẹp trai hoặc không có cả 2 đặc điểm kia, những người còn lại thì bắt buộc phải có những người có luôn cả 3 đặc điểm:

Nếu các đặc điểm được phân bố như trên hình, ta có đáp án chính là số lượng ô ở dòng trên cùng. Như vậy số lượng cần gặp trước khi chắc chắn tìm được người phù hợp trong trường hợp này là $\min(3N - A - B - C, N)$, lưu ý là nếu không có ai có cả 3 đặc điểm thì lẽ dĩ nhiên là Phúc không biết điều đó và phải gặp cả N người thì mới phát hiện ra.

G – BỎ PHIẾU

Time limit: 2s

Memory limit: 512MB

Để số phiếu của cả N ứng cử viên bằng nhau, ta có tổng số lượng lá phiếu trong lớp phải chia hết cho N. Tổng số lượng lá phiếu chính là số lượng lá phiếu đã bầu, $A_1 + A_2 + \cdots + A_N$, và K lá phiếu chưa bầu. Tức là, để số phiếu của cả N ứng cử viên bằng nhau, điều đầu tiên là $(A_1 + A_2 + \cdots + A_N + K) \bmod N = 0$.

Khi đó số phiếu của một ứng cử viên bất kỳ sẽ là $X = \frac{A_1+A_2+\cdots+A_N+K}{N}$. Rõ ràng nếu hiện tại đã có một ứng cử viên có số phiếu vượt quá X thì trường hợp N ứng cử viên có số phiếu bằng nhau sẽ không thể xảy ra.

Vậy chỉ cần xét 2 điều kiện ở trên là ta đã có thể xác định trường hợp N ứng cử viên có số phiếu bằng nhau có xảy ra hay không rồi.