TAA - LEE 02

A. Distância Binária

2 seconds, 256 megabytes

Na Teoria da Informação, a distância de hamming entre duas strings de mesmo comprimento é o número de posições nas quais elas diferem entre si. De uma maneira simplificada, essa distância corresponde ao menor número de substituições necessárias para transformar uma string na outra. Por exemplo, considerando os caracteres de uma string, a distância de hamming para as palavras "cebola" e "cevada", pode ser calculada da seguinte forma:

```
c e b o l a
c e v a d a
0 1 2 3 4 5
    | | |
```

Ou seja, para transformar cebola em cevada, basta alterar, na palavra "cevada", as posições 2, 3 e 4. Assim, a distância de hamming para essas duas palavras é 3.

Essa distância é amplamente utilizada em computação, em áreas como teoria da informação, teoria da codificação, telecomunicações e criptografia. Em telecomunicações ela é utilizada para contar o número de bits corrompidos na transmissão de uma mensagem de um determinado comprimento. Neste caso, a distância é calculada a partir do alfabeto binário $\{0, 1\}$ (palavras compostas por zeros e uns). Por exemplo, os números binários 101 e 011 tem distância igual a dois porque é necessário alterar dois bits para transformar um no outro.

Assim, sua tarefa é dados dois números inteiros positivos, calcular a distância Hamming entre eles.

Input

A primeira linha contém um inteiro T ($1 \leq T \leq 100000$), que representa o número de casos de teste.

Cada uma das N linhas seguintes contém dois inteiros positivos, separados por espaço, na base decimal X e Y ($0 \leq X, Y \leq 2^{63}$), que representam os números cujo a distância de hamming deve ser calculada.

Output

A saída deve possuir N linhas, sendo uma para cada caso de teste contendo a distância hamming das representações binárias de X e Y.

input
1 6 7
output
1

input
2 8 23 15 8
output
5 3

No primeiro caso de testes, os números 6 e 7 na base binária são 110 e 111, e apenas o bit menos sigficativo precisa ser modificado para transformar um em outro.

B. Número Abandonado

1 second, 256 megabytes

Você foi convidado para uma festa vip de números inteiros.

Nessa festa, todo o número tem o seu respectivo par, que é o seu igual, exceto por um desafortunado e abandonado número inteiro. Avisado da solidão desse número inteiro, você resolveu encontrar esse número solitário e garantir que ele também se divirta na festa.

Input

A entrada contém múltiplos casos de testes. Cada caso de teste contém duas linhas. A primeira linha contém um inteiro N ($1 \leq N \leq 9999$), onde N é ímpar e representa o número de inteiros na festa.

A segunda linha contém N inteiros V ($0 \leq V \leq 10^9$), separados por espaço, que representam os números presentes na festa.

A entrada termina por $N = -1$, a qual não deve ser processada.

Output

A saída deve conter um único inteiro para cada caso de teste, que é o número abandonado.

input
7 1 1 8 5 3 3 8 -1
output
5

input
7 2 8 2 9 3 8 3 5 4 1 4 4 1 -1
output
9 4

C. Soma Igual

1 second, 256 megabytes

Nesta tarefa, você irá receber um conjunto de elementos C, em ordem não-decrescente, e um valor S. Seu objetivo é determinar se existe um **par de elementos** distintos a_i e b_j em C, com $i \neq j$, com cuja soma seja igual a S.

Input

A primeira linha da entrada contém um inteiro T ($1 \leq T \leq 100$), que indica a quantidade de casos de teste. Cada caso de testes é composto por uma linha que contém um inteiro N ($2 \leq N \leq 10^4$) que indica a quantidade de elementos, uma linha com N inteiros V_i ($1 \leq V_i \leq 10^6$), separados por um espaço, que representam os elementos e, por último uma linha com um inteiro S ($1 \leq S \leq 10^6$) que é o valor a ser verificado.

Output

Para cada caso de teste, imprima uma linha de saída com a mensagem `"SIM"`, se existir um par de elementos distintos cuja soma seja igual a S, ou `"NAO"`, caso contrário, conforme os exemplos.

input
1 7 1 5 12 15 16 17 22 29
output
SIM

input
2 5 1 2 3 4 5 10 4 3 7 7 12 14

output
NAO SIM

A solução esperada não exige o uso de nenhuma estrutura de dados complexa, apenas manipulação simples de vetores.

D. Super Bit, Ativar!

1 second, 256 megabytes

A cada segundo, milhares partículas chamadas de raios cósmicos, viajando perto da velocidade da luz, atingem cada metro quadrado da atmosfera da Terra, lançando uma cascata de partículas carregadas que caem na superfície do planeta. Quando essas partículas atingem os transistores de microchips, elas podem causar falhas e até mesmo provocar travamentos de computadores. Esse é um fenômeno raro, mas bits podem ser invertidos devido a uma dessas interferências.

Pensando no pior, a empresa Inovação em Dispositivos Periféricos (IDP) pediu a sua ajuda para consertar possíveis problemas que possam ocorrer devido a essa interferência cósmica. Sua tarefa é a seguinte, após detectar que em um número um bit está com problemas, você deve reativar esse bit, tranformando-o em 1, caso não o seja. Por exemplo, se na memória encontra-se o número 32, que está com problema no bit 3 (começando a contagem a partir de zero e do bit menos significativo), a ativação desse bit fica o seguinte:

```
7 6 5 4 3 2 1 0    <- Posições
0 0 1 0 0 0 0 0    <- 32 em binário
_______-_____    <- Ativando-se o bit na posição 3
0 0 1 0 1 0 0 0    <- 40 em binário
```

Input

A primeira linha da entrada possui dois inteiros N ($0 < N \leq 100000$) e B ($0 \leq B \leq 31$), separados por espaço. As N linhas seguintes possuem N inteiros V ($0 \leq V \leq 10^9$) cada, que são os números cujo o bit na posição B deve ser ativado.

Output

A saída deve possuir N linhas, onde cada linha é o inteiro que V_i com o respectivo bit ativado, conforme os exemplos.

input
1 3 32

output
40

input
3 2 8 32 20

output
12 36 20

Curiosidade aleatória: https://www.youtube.com/watch?v=AaZ_RSt0KP8

E. Pribit

1 second, 256 megabytes

Euclides é um menino que adora números primos, porém ele já mexeu tanto com esses números que agora ficou quase sem ideias de o que mais há para descobrir com eles. Tentando diferentes abordagens, ele decidiu misturar números primos com sua respectiva representação binária, e acabou chamando esses números de **pribits**!

Um pribit é um número que, dada a sua representação binária, possui um número primo de bits com valor igual a 1 (um).

Agora, Euclides quer a sua ajuda para, dado um número, responder se esse inteiro é um pribit ou não.

Input

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 10000$), que indica a quantidade de números a serem verificados. Em seguida haverá N linhas onde cada linha contém um inteiro V ($1 \leq V \leq 10^{15}$) que são os valores a serem verificados.

Output

Para cada inteiro V, imprima uma linha de saída com a mensagem "X eh um pribit" se o número for um pribit, onde X é o número verificado, ou "X nao eh um pribit" caso contrário.

input
3 6 4 12

output
6 eh um pribit 4 nao eh um pribit 12 eh um pribit

input
4 7 11 17 19

output
7 eh um pribit 11 eh um pribit 17 eh um pribit 19 eh um pribit

No primeiro exemplo do primeiro caso de teste, o valor 6 em binário possui a representação 110, que possui dois bits com o valor 1. Sendo 2 primo, logo este é um pribit.

F. TriOU

1 second, 256 megabytes

Neste exercício, três inteiros A, B e C foram utilizados para gerar um quarto inteiro Y, que é a aplicação da operação OR bit-a-bit desses três números, da seguinte forma:

$$Y = (A \vee B \vee C)$$

Sua tarefa é, dados apenas os dois dos inteiros (A e B) e o resultado da operação (Y), descobrir qual o inteiro faltante (C), se for possível. Por exemplo, dados os valores $A = 8$, $B = 2$ e $Y = 11$, pode-se verificar que $C = 1$ é uma resposta válida para o inteiro faltante, pois:

```
1000  ->  8 (A)
0010  ->  2 (B)
0001  ->  1 (?)

____
1011  -> 11 (Y)
```

Ademais, **além de encontrar o inteiro faltante, este deve ser a menor resposta válida possível**, se esta existir.

Input

A primeira linha da entrada contém um inteiro T ($1 \leq T \leq 1000$) que representa o número de casos de teste.

Em seguida, há T pares de linhas onde na primeira dessas linhas há dois inteiros A e B ($0 \leq A, B \leq 10^7$), que são os números dos quais se tem acesso. A segunda linha do caso de teste contém um inteiro Y ($0 \leq Y \leq 10^7$), que é o resultado da operação.

Output

A saída deve conter T linhas, onde cada deve conter um inteiro C ($0 \leq C \leq 10^7$) que é o mínimo valor possível para a solução da operação, ou -1 se tal número não existir.

input
1 8 2 11
output
1

input
2 4 8 15 13 5 22
output
3 -1

A operação OU bit-a-bit, representada pelo símbolo $\vee$, é uma operação lógica que compara cada par de bits em duas sequências binárias e retorna um resultado onde um bit no resultado é definido como 1 se pelo menos um dos bits correspondentes nas duas sequências for 1, caso contrário, seu valor é zero. Por exemplo, dados $A = 1010$ e $B = 1101$, a operação OU bit-a-bit ($A \mid B$) resulta em 1111.