Problema A
Alarme Despertador

Nome do arquivo fonte: **alarme.c, alarme.cpp ou alarme.java**

Daniela é enfermeira em um grande hospital, e tem os horários de trabalho muito variáveis.Para piorar, ela tem sono pesado, e uma grande dificuldade para acordar com relógios despertadores.

Recentemente ela ganhou de presente um relógio digital, com alarme com vários tons, e tem esperança que isso resolva o seu problema. No entanto, ela anda muito cansada e quer aproveitar cada momento de descanso. Por isso, carrega seu relógio digital despertador para todos os lugares, e sempre que tem um tempo de descanso procura dormir, programando o alarme despertador para a hora em que tem que acordar. No entanto, com tanta ansiedade para dormir, acaba tendo dificuldades para adormecer e aproveitar o descanso.

Um problema que a tem atormentado na hora de dormir é saber quantos minutos ela teria de sono se adormecesse imediatamente e acordasse somente quando o despertador tocasse. Mas ela realmente não é muito boa com números, e pediu sua ajuda para escrever um programa que, dada a hora corrente e a hora do alarme, determine o número de minutos que ela poderia dormir.

Entrada

A entrada contém vários casos de teste. Cada caso de teste é descrito em uma linha, contendo quatro números inteiros H1, M1, H2 e M2, com H1:M1 representando a hora e minuto atuais, e H2:M2 representando a hora e minuto para os quais o alarme despertador foi programado ($0 \leq H1 \leq 23$, $0 \leq M1 \leq 59$, $0 \leq H2 \leq 23$, $0 \leq M2 \leq 59$).

O final da entrada é indicado por uma linha que contém apenas quatro zeros, separados por espaços em branco.

Os dados devem ser lidos da entrada padrão.

Saída

Para cada caso de teste da entrada seu programa deve imprimir uma linha, cada uma contendo um número inteiro, indicando o número de minutos que Daniela tem para dormir.

O resultado de seu programa deve ser escrito na saída padrão.

Exemplo

Entrada	Saída
1 5 3 5 23 59 0 34 21 33 21 10 0 0 0 0	120 35 1417

Problema B
Sub-Prime

Nome do arquivo fonte: **subprime.c, subprime.cpp ou subprime.java**

A mais recente crise econômica foi em parte causada pela forma como os bancos faziam empréstimos para pessoas que não tinham capacidade de honrá-los e revendiam tais empréstimos para outros bancos (debêntures). Obviamente, quando as pessoas pararam de pagar os empréstimos, o sistema inteiro entrou em colapso.

A crise foi tão profunda que acabou atingindo países do mundo inteiro, inclusive a Nlogônia, onde o honrado primeiro ministro Man Dashuva ordenou que o presidente do Banco Central procurasse uma solução para o problema. Esse, por sua vez, teve uma idéia brilhante: se cada banco fosse capaz de liquidar seus empréstimos somente com suas reservas monetárias, todos os bancos sobreviveriam e a crise seria evitada. Entretanto, com o elevado número de debêntures e bancos envolvidos, essa tarefa é extremamente complicada, e portanto ele pediu a sua ajuda para escrever um programa que, dados os bancos e as debêntures emitidas, determine se é possível que todos os bancos paguem suas dívidas, utilizando suas reservas monetárias e seus créditos.

Entrada

A entrada contém vários casos de teste. A primeira linha de um caso de teste contém dois inteiros B e N, indicando respectivamente o número de bancos ($1 \leq B \leq 20$) e o número de debêntures emitidas pelos bancos ($1 \leq N \leq 20$). Os bancos são identificados por inteiros entre 1 e B. A segunda linha contém B inteiros Ri separados por espaços, indicando as reservas monetárias de cada um dos B bancos ($0 \leq Ri \leq 10^4$, para $1 \leq i \leq B$). As N linhas seguintes contêm cada uma três inteiros separados por espaços: um inteiro D, indicando o banco devedor ($1 \leq D \leq B$), um inteiro C, indicando o banco credor ($1 \leq C \leq B$ e $D \neq C$), e um inteiro V, indicando o valor da debênture ($1 \leq V \leq 10^4$).

O final da entrada é indicado por uma linha que contém apenas dois zeros, separados por um espaço em branco.

Saída

Para caso de teste, seu programa deve imprimir uma única linha, contendo um único caractere: 'S', se for possível liquidar todos as debêntures sem intervenção do Banco Central da Nlogônia, e 'N', se algum banco precisar de empréstimos do governo para liquidar suas debêntures.

Exemplo:

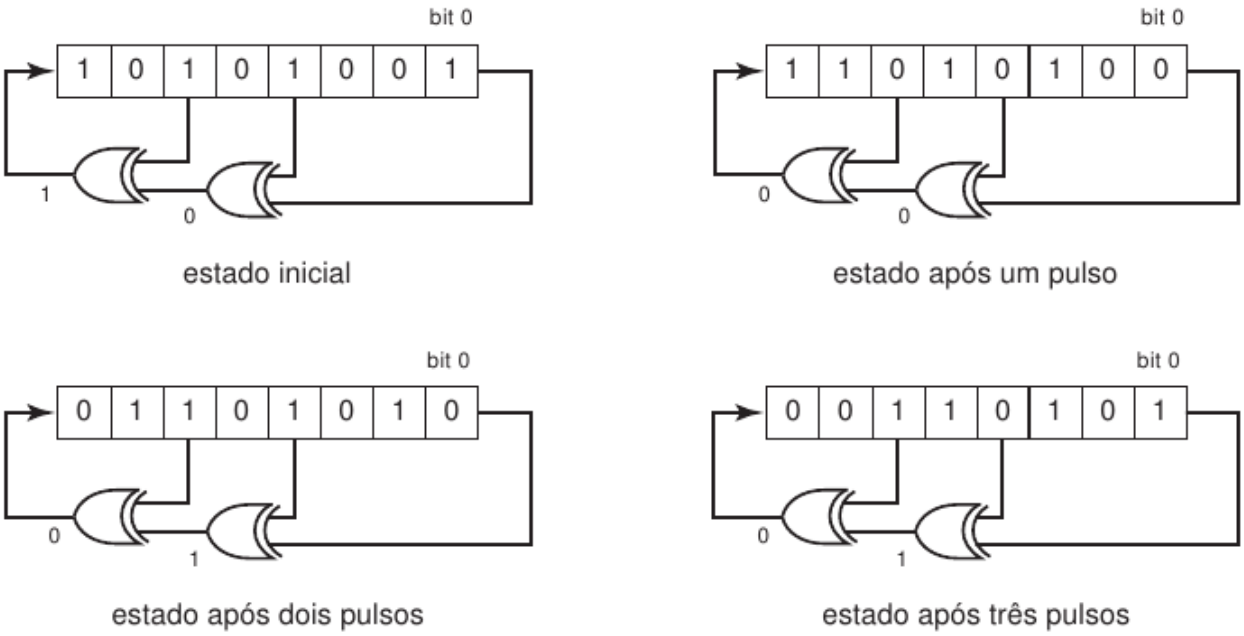
Entrada	Saída
3 3 1 1 1 1 2 1 2 3 2 3 1 3 3 3 1 1 1 1 2 1 2 3 2 3 1 4 3 3 1 1 1 1 2 2 2 3 2 3 1 2 0 0	S N S

Problema C
Registrador de Deslocamento

Nome do arquivo fonte: **registrador.c, registrador.cpp ou registrador.java**

Um Registrador de Deslocamento é um circuito que desloca de uma posição os elementos de um vetor de bits. O registrador de deslocamento tem uma entrada (um bit) e uma saída (também um bit), e é comandado por um pulso de relógio. Quando o pulso ocorre, o bit de entrada se transforma no bit menos significativo do vetor, o bit mais significativo é jogado na saída do registrador, e todos os outros bits são deslocados de uma posção em direção ao bit mais significativo do vetor (em direção à saída).

Um Registrador de Deslocamento com Retroalimentação Linear (em inglês, lfsr) é um registrador de deslocamento no qual o bit de entrada é determinado pelo valor do ou-exclusivo de alguns dos bits do registrador antes do pulso de relógio. Os bits que são utilizados na retroalimentação do registrador são chamados de torneiras. A figura abaixo mostra um lfsr de 8 bits, com três torneiras (bits 0, 3 e 5).



Neste problema, você deve escrever um programa que, dados o número de bits de um lfsr, quais bits são utilizados na retroalimentação, um estado inicial e um estado final do lfsr, determine quantos pulsos de relógio serão necessários para que, partindo do estado inicial, o lfsr chegue ao estado final (ou determinar que isso é impossível).

Entrada

A entrada contém vários casos de teste. Cada caso de teste é composto por três linhas. A primeira linha contém dois números inteiros N, T , indicando respectivamente o número de bits ($2 \leq N \leq 32$) e o número de torneiras ($2 \leq T \leq N$). Os bits são identificados por inteiros de 0 (bit menos significativo) a N - 1 (bit mais significativo). A segunda linha contém T inteiros, separados por

espaços, apresentando os identificadores dos bits que são torneiras, em ordem crescente. O bit 0 sempre é uma torneira. A terceira linha contém dois números em notação hexadecimal I e F , separados por um espaço em branco, representando respectivamente o estado inicial e o estado final do lfsr.

O final da entrada é indicado por uma linha que contém dois zeros separados por espaços em branco.

Saída

Para cada caso de teste da entrada seu programa deve imprimir uma única linha. Se for possível chegar ao estado final a partir do estado inicial dado, a linha da saída deve conter apenas um inteiro, o menor número de pulsos de relógio necessários para o lfsr atingir o estado final.

Caso não seja possível, a linha deve conter apenas o caractere '*'.

O resultado de seu programa deve ser escrito na saída padrão.

Exemplo:

Entrada	Saída
8 3 0 3 5 a9 35 5 2 0 4 1b 2 7 3 0 2 3 4d 1a 0 0	3 * 61