Problema I – Incríveis Permutações

Limite de tempo: 1s

Limite de memória: 256MB

No reino da **Permutação**, o rei Ígor III adora enigmas matemáticos. Recentemente, ele propôs um novo desafio aos seus súditos: organizar os números de 1 a n em uma fila real de forma **bonita** — ou seja, de modo que **nenhum número esteja imediatamente ao lado de outro cuja diferença seja exatamente** 1.

Segundo o rei, números vizinhos que diferem em apenas 1 brigam como irmãos, e ele não quer confusão em seu palácio! Por isso, toda permutação enviada ao rei deve garantir que esses conflitos sejam evitados.

Como conselheiro real, você foi encarregado de resolver esse enigma. Dado o valor de n, construa uma permutação de números de 1 a n que satisfaça a exigência do rei, ou diga que isso é impossível.

Entrada

A entrada consiste em uma única linha contendo um número inteiro n ($1 \leq n \leq 10^6$), representando o tamanho da fila que o rei deseja organizar.

Saída

Imprima uma permutação dos números de 1 a n onde **nenhum par de elementos adjacentes tenha diferença igual a** 1. Se não for possível atender ao pedido do rei, imprima `NO SOLUTION`.

Qualquer permutação que atenda as restrições do enunciado será aceita pelo juiz.

Exemplo

Entrada	Saída
`5`	`4 2 5 3 1`
`3`	`NO SOLUTION`

Notas

No primeiro exemplo, a fila 4 2 5 3 1 satisfaz as regras do rei Ígor: nenhuma dupla vizinha briga, pois não há diferença de 1 entre vizinhos.

No segundo exemplo, com apenas três números, não é possível evitar os conflitos — e o rei, desapontado, mandará você direto para o calabouço (ou talvez só para a próxima tentativa).

Problema J – Jira Jira

Limite de tempo: 1s

Limite de memória: 256MB

Ana e Beto são um casal e estavam relembrando de como se conheceram. Os dois eram calouros e foram participar da **II Maratona de Programação do IFB**, onde em meio a desafios, coffee break e balões, também encontraram o amor! Desde então formaram uma equipe e competiram por muitos anos na **Maratona de Programação SBC**. Em um dos eventos da maratona, eles foram para um parque de diversões e foram em vários brinquedos, entre eles, a roda gigante! Foi nesse momento a dois, que o romance entrou no ar e eles tiveram um lindo momento (um dia antes da maratona!).

Porém, após todos esses anos e relembrando desse encontro nas alturas, eles se perguntam se conseguiriam fazer um problema relacionado a rodas gigantes. Eles se lembraram também que o Cláudio, ~~a vela~~ o terceiro membro da equipe, também entrou na roda gigante, porém em outra cabine. Mas eles não se lembravam se a cabine de Cláudio estava acima ou abaixo deles. Então, esse é o problema! Dada as posições da cabine do casal e de Cláudio, você deve dizer se a cabine de Cláudio está acima, abaixo ou se estão de frente para o outro!

Entrada

A entrada consiste de dois inteiros, AB e C ($0 \leq AB, C < 360$), separados por um espaço, que indicam, respectivamente, os ângulos em graus da cabine de Ana e Beto; e da cabine de Cláudio; em relação ao segmento de reta que liga o centro da roda-gigante até seu ponto mais a direita no sentido anti-horário.

Saída

A saída deve conter a string “Olha o Claudio ali em cima!"caso a cabine de Cláudio esteja acima do casal, “O Claudio ta ali embaixo!"caso a cabine do Cláudio esteja abaixo do casal ou “O Claudio ta do outro lado da roda!"caso a cabine do casal estejam de frente para o outro.

Exemplo

Entrada	Saída
`0 90`	`Olha o Claudio ali em cima!`
`258 91`	`Olha o Claudio ali em cima!`

Notas

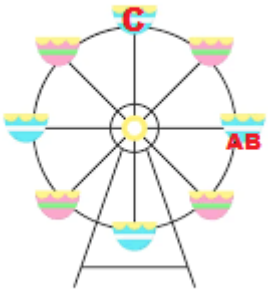


Imagem ilustrativa do primeiro exemplo.

Problema K – KMP

Limite de tempo: 1.5s

Limite de memória: 256MB

A KMP (Kodando na Maratona de Programação) é uma *start-up* formada por ex-maratonistas que apoia eventos de programação em todo o Brasil. Sua ação de apoio mais recente foi um sorteio de prêmios, cujo regulamento foi bastante peculiar: cada participante recebeu um número positivo x_i e foi sorteado um primo p. Todos os participantes cujo número x_i era divisível por p foram contemplados.

Dado sucesso do evento, a empresa quer repetir o sorteio, só que agora os participantes devem ser inscritos em dupla! Cada dupla vai concorrer com o inteiro $y = x_i x_j$, onde x_i e x_j são os inteiros recebidos por cada um dos membros da dupla. Dado sua popularidade, há N participantes querendo formar dupla com o professor Saad, onde cada participante é identificado por um positivo distinto entre 1 e N. Ele recebeu o inteiro x_S e pretende formar dupla com o participante que maximizar a chance de serem contemplados, ou seja, ele quer escolher o k-ésimo participante tal que $\rho(x_S x_k)$ seja o maior possível, onde $\rho(n)$ corresponde ao número de primos distintos que dividem n.

Como o professor Saad está muito ocupado com a organização do evento, ajude-o escrevendo um programa que, dados os valores de N, x_S e os inteiros atribuídos aos participantes, determine o identificador k do participante que Saad deve fazer dupla no sorteio.

Entrada

A primeira linha da entrada contém os inteiros N ($1 \leq N \leq 5 \times 10^5$) e x_S ($1 \leq x_S \leq 2 \times 10^7$), separados por um espaço em branco.

A segunda linha da entrada contém N inteiros x_i ($1 \leq x_i \leq 10^7$), separados por um espaço em branco, indicando o inteiro recebido pelo i-ésimo participante.

Saída

Imprima, em uma linha, o identificador k do participante que deve formar dupla com o professor Saad. Se há mais um participante que maximize as chances do professor, imprima o identificador de qualquer um deles.

Exemplo

Entrada	Saída
3 10 6 15 21	3
7 8 1 2 3 4 5 6 7	3
3 5 2 2 2	1

Notas

No primeiro caso, temos que $\rho_1(10 \times 6) = \rho_1(60) = \rho_1(2^2 \times 3 \times 5) = 3$, $\rho_2(10 \times 15) = \rho_2(150) = \rho_2(2 \times 3 \times 5^2) = 3$ e $\rho_3(10 \times 21) = \rho_3(210) = \rho_3(2 \times 3 \times 5 \times 7) = 4$. Portanto o professor Saad deve fazer dupla com o participante 3.

No segundo caso, Saad poderia fazer dupla com os participantes 3, 5 ou 7.

No terceiro caso, observe que os números recebidos pelos participantes não são, necessariamente, distintos.

Problema L – Leila, a CabeLeila

Limite de tempo: 1s

Limite de memória: 256MB

Em uma bela tarde, Leila, a Cabeleila, decidiu se aposentar de seus saques intergaláticos e abrir um salão de beleza! O Salão da CabeLeila fez tanto sucesso que, se ela conseguisse, poderia trabalhar 24 horas e sempre teria um cliente a espera.

Porém, com todo o sucesso, também vem os problemas... Leila gosta de testar novos produtos para melhorar a saúde capilar de seus clientes. Com a alta demanda de clientes, ela precisa então que você, aprendiz de cortes de cabelo intergaláticos, faça os testes e apresente os resultados para ela!

Leila tem N produtos capilares, entre tônicos, xampus, condicionadores, máscaras. Cada produto p_i tem um nível de acidez a_i e um nível de qualidade q_i. Como é de conhecimento geral, o produto final não pode passar de uma acidez X, então ela pediu para você listar a melhor mistura utilizando j produtos que não ultrapasse esse limite. Como ela está sem tempo para decidir qual a melhor quantidade de produtos, liste a melhor mistura para todo $1 \leq j \leq N$.

Entrada

A primeira linha da entrada consiste em dois inteiros N, X ($1 \leq N \leq 10$, $1 \leq X \leq 10^6$), que correspondem, respectivamente, a quantidade de produtos e o limite de acidez máximo.

As próximas N linhas, possuem dois inteiros a_i, q_i ($1 \leq a_i, q_i \leq 10^6$) representando as propriedades de p_i.

Saída

A saída consiste em N inteiros, seja r_i o i-ésimo inteiro da resposta, r_i deve ser a melhor mistura usando i produtos tal que a soma de acidez não ultrapasse X. Caso não exista resposta, imprima 0.

Exemplo

Entrada	Saída
3 10 9 11 3 6 4 7	11 13 0

Notas

No exemplo, Leila:

- Com 1 produto, consegue uma qualidade de 11, usando apenas o primeiro produto.
- Com 2 produtos, consegue uma qualidade de 13, usando o segundo e terceiro produtos.
- Com 3 produtos, não consegue fazer a mistura, pois a acidez ultrapassa o limiar X.

Problema M – Music Tour

Limite de tempo: 1s

Limite de memória: 256MB

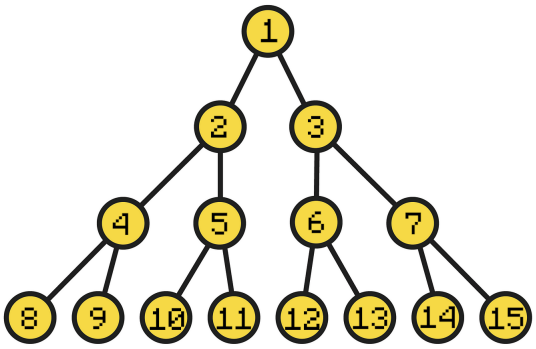
Depois do lançamento do seu novo álbum, Chuu irá fazer um tour em Binarisília!

Binarisília é uma cidade que pode ser representada por uma *árvore binária perfeita* de profundidade N, onde cada região é representada por um vértice da árvore e as arestas representam rodovias que interligam essas regiões.

Aqui, uma árvore binária perfeita de profundidade N, com $N \in \mathbb{N}$, é definida como uma árvore que atende as seguintes propriedades:

- o nível 1 da árvore contém apenas o vértice 1;
- todo vértice que não está no nível N possui exatamente dois filhos;
- todo vértice no nível N possui nenhum filho.

Por exemplo, a figura a seguir ilustra uma árvore binária perfeita de profundidade 4:



O Aeroporto Internacional de Binarisília está na região representada pelo vértice 1. Chuu chegará no aeroporto e irá apresentar um show em todas as regiões da cidade. Depois, ela voltará para o aeroporto e irá embora. Chuu irá usar as rodovias da cidade para viajar de uma região para outra; por exemplo, na árvore ilustrada acima, para Chuu ir da região 8 para a 9, ela precisa fazer o caminho $8 \to 4 \to 9$.

Dado o inteiro N que especifica a estrutura de Binarisília, o agente de viagem de Chuu quer saber o seguinte, qual é o número mínimo de visitações diferentes que Chuu precisa fazer para apresentar um show em toda região de Binarisília e ir embora? Ajude-o neste problema! Como o número pode ser muito grande, apresente a resposta módulo $10^9 + 7$.

Entrada

A única linha da entrada contém um único número inteiro N ($1 \leq N \leq 3 \cdot 10^5$) — a profundidade da árvore que representa Binarisília.

Saída

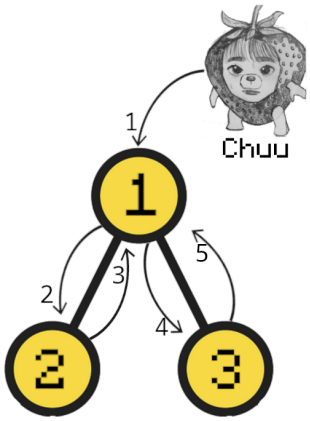
Imprima uma linha com um único número inteiro — a quantidade mínima de visitações que Chuu precisa fazer no seu tour para ela apresentar um show em todas as regiões da cidade e ir embora, módulo $10^9 + 7$.

Exemplo

Entrada	Saída
2	5
4	29
1	1
39	511620080
300000	360325491

Notas

No primeiro caso de teste, para $N = 2$, a resposta é 5, Chuu pode passar por Binarisília da seguinte forma:



A ordem de visitação do tour é $1 \to 2 \to 1 \to 3 \to 1$, totalizando 5 visitações diferentes. Note que as regiões visitadas para voltar ao aeroporto são contadas.

No quarto caso de teste, para $N = 39$, Chuu faz 1099511627773 visitações, então, a resposta é 511620080 pois este é o resto da divisão de 1099511627773 por $10^9 + 7$.

Problema A – Academia

Limite de tempo: 1s

Limite de memória: 256MB

Powerlifting é um esporte que está crescendo cada vez mais. A competição se baseia em três movimentos, o agachamento, o supino e o levantamento terra. Cada competidor tem o direito de tentar 3 vezes cada movimento e vence aquele que tiver a maior soma do maior peso levantado de cada movimento entre todos os atletas da mesma categoria de peso.

Ozne está colhendo vários dados estatísticos e observando como os números podem influenciar nesse esporte, ele está meio ocupado, então pediu sua ajuda para calcular a pontuação $Total$, que consiste na soma dos 3 movimentos somados, ou seja $Total = S + B + D$, onde S é o agachamento, B é o supino e D é o levantamento terra. Dado os valores S, B e D, diga para Ozne qual é o $Total$ acumulado!

Entrada

A entrada consiste em três inteiros S, B, D ($1 \leq S,\ B,\ D \leq 500$), representando os pesos que um levantador fez no agachamento, supino e levantamento terra, respectivamente.

Saída

A saída consiste em um único inteiro $Total$, representando quanto o atleta levantou nessa competição.

Exemplo

Entrada	Saída
`154 108 175`	`437`
`134 94 166`	`394`

Problema B – Bola Quadrada

Limite de tempo: 4s

Limite de memória: 256MB

O Kiko finalmente teve o seu sonho realizado: ganhou uma bola quadrada de tamanho $n \times n$ da Dona Florinda.

Essa bola quadrada possui números inteiros de 1 a n, mas possui um pequeno problema, ela não está totalmente preenchida. Vaidoso como é, Kiko quer preencher a bola da seguinte forma:

1. Os números devem ser inteiros no intervalo $[1, n]$.
2. As linhas não devem ter números repetidos.
3. As colunas não devem ter números repetidos.
4. Todos os espaços devem ser preenchidos.

Ajude Kiko a preencher a sua bola quadrada para que possa contar vantagem sobre o Chavinho.

Entrada

A primeira linha da entrada possui um inteiro n. As próximas n linhas descrevem cada linha da bola de Kiko. Cada linha possui n inteiros, separados por um espaço. Valores 0 indicam que aqueles espaços da bola não foram preenchidos.

Restrições:

- $1 \leq n \leq 7$
- Não haverá linha ou coluna com números repetidos.

Saída

Se for possível preencher a bola de Kiko de acordo com as restrições do problema, imprima n linhas, cada uma com n inteiros, separados por um espaço, descrevendo a bola. O juiz aceitará qualquer descrição de bola, desde que seja válida.

Caso não seja possível, imprima uma linha com "`-1`".

Exemplo

Entrada	Saída
3 1 0 0 0 2 0 0 0 3	1 3 2 3 2 1 2 1 3
3 0 0 0 0 0 0 0 0 0	1 2 3 2 3 1 3 1 2
2 0 1 0 2	2 1 1 2
3 0 1 2 0 2 3 3 0 1	-1

Notas

Os 3 primeiros exemplos ilustram cenários em que é possível preencher a bola. Nos primeiros dois, inclusive, é possível preencher a bola de várias formas diferentes.

No último exemplo, não é possível preencher a bola de Kiko conforme sua vontade.

Problema C – Caleb, Chefe Competente

Limite de tempo: 1s

Limite de memória: 256MB

Caleb criou uma empresa com n funcionários e estabeleceu uma hierarquia clara de chefias. Ele, sendo o fundador, não tem chefe, mas todos os outros funcionários possuem um chefe direto. Dizemos que a é chefe de b se, ao subir na hierarquia de b, encontramos a. Formalmente, a é **chefe** de b se satisfaz a definição recursiva a seguir:

- a é chefe direto de b, ou
- a é chefe do chefe direto de b.

Recentemente, Caleb implementou uma nova regra para manter as reuniões mais eficientes e garantir que todos os níveis da hierarquia sejam devidamente representados. Essa regra funciona da seguinte forma. Inicialmente, m funcionários são convidados para uma reunião. Realizamos o seguinte procedimento enquanto possível:

- Processamos todo par de funcionáros a, b convidados para a reunião. Consideremos a lista $p_1, \ldots, p_k$ de funcionários da empresa que são chefes simultaneamente de a e b. Pode ser provado que existe um funcionário dessa lista mais baixo na hierarquia, isto é, existe i tal que p_j é chefe de p_i para $j \neq i$. O funcionário p_i é, então, convidado para a reunião.

Note que, ao convidar um novo funcionário para a reunião, o procedimento acima terá de ser realizado com a nova lista de funcionários convidados, até que não haja mais funcionários convidados.

Os funcionários serão indexados de 1 até n e será informado o chefe direto de cada funcionário, com exceção do fundador Caleb, que terá índice 1. Dada a lista de funcionários inicialmente convidados, ajude Caleb e encontre todos os funcionários que participarão da reunião.

Entrada

A primeira linha de entrada contém dois inteiros n, m ($2 \leq m \leq n \leq 10^5$) — a quantidade de funcionários na empresa e a quantidade chamada para reunião, respectivamente.

A segunda linha de entrada contém $n-1$ inteiros $p_2, p_3, \ldots, p_n$ ($1 \leq p_i \leq n$) — p_i representando o chefe direto do funcionário i.

A última linha de entrada comtém m inteiros $c_1, \ldots, c_m$ ($1 \leq c_i \leq n$; $c_i \neq c_j$ se $i \neq j$) — os funcionários chamados para reunião.

Saída

Imprima uma linha contendo um inteiro k — a quantidade total de funcionários que terão que participar da reunião.

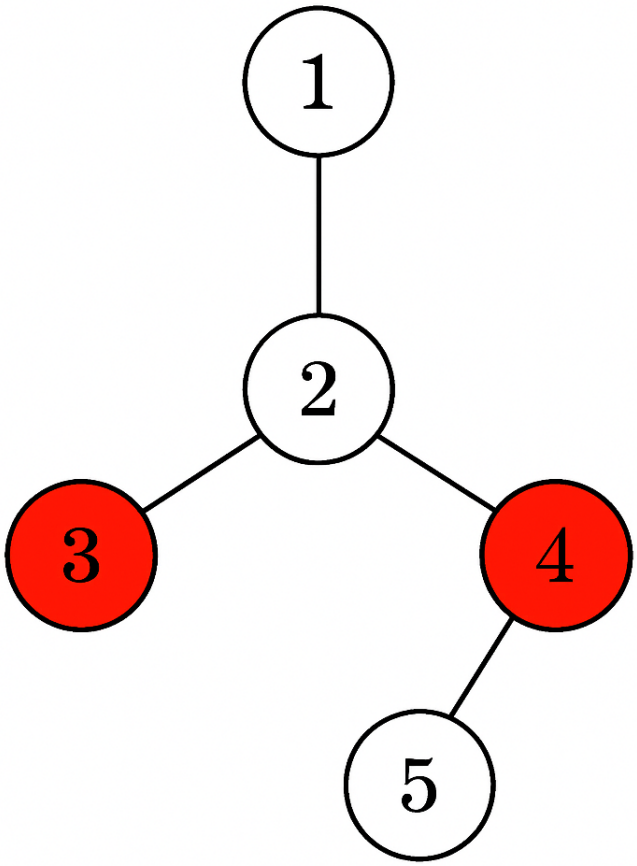
Imprima outra linha contendo k inteiros $r_1, \ldots, r_k$ — os índices **ordenados de forma crescente** dos funcionários convidados para a reunião.

Exemplo

Entrada	Saída
5 2 1 2 2 4 3 4	3 2 3 4
9 4 1 2 2 1 5 4 2 4 7 3 6 5	6 1 2 3 5 6 7

Notas

A árvore abaixo é a árvore de hierarquias do caso de teste 1, com os funcionários em vermelho sendo os inicialmente chamados para reunião. Considerando o par 3, 4, são chefes simultaneamente deles os vértices 1 e 2. O mais baixo na hierarquia é 2, então este é chamado para a reunião. Realizando o procedimento com qualquer outro par nos da funcionários já convidados para a reunião, então paramos.



Problema D – Daniel Triste

Limite de tempo: 1s

Limite de memória: 256MB

Daniel é um adorável tigre que mora com os pais. Ele é um filhote felino feliz, mas fica frustrado se fanfarrões firulam o forte nome da família. Os **Saad** se orgulham de serem sinônimos de boa fortuna, boa sorte, sucesso e felicidade, e insinuar que sejam tristes é uma ofensa. Para minimizar os efeitos desta *bullynagem*, Daniel pediu apoio aos maratonistas.

Crie um plugin para o zap que corrige o nome de Daniel caso o escrevam errado, afinal, não é triste prevenir tretas para três tigres com prática de programação.

Entrada

A entrada consiste de uma mensagem com pelo menos um e não mais que 100 caracteres. É garantido que o nome de Daniel, quando apresentado, é separado das demais palavras da mensagem.

Saída

Apresente a mensagem como enviada, mas corrija cada ocorrência "errada" do nome de Daniel. Cuidado com a caixa da letra!

Exemplo

Entrada	Saída
`Daniel Sad`	`Daniel Saad`
`so sad`	`so sad`
`Sorria, Daniel Saad!`	`Sorria, Daniel Saad!`
`daniel sad, but true...`	`daniel saad, but true...`
`Daniel Sadico reprovou geral.`	`Daniel Sadico reprovou geral.`
`3 PRATOS DE TRIGO PARA 3 DANIEL SAD.`	`3 PRATOS DE TRIGO PARA 3 DANIEL SAAD.`
`Sad baixo!`	`Sad baixo!`

Notas

Os fanfarrões firulam apenas de uma forma: retirando um dos 'a' do sobrenome de Daniel de modo que seja visto como "triste" (em inglês: *sad*). Qualquer outra variação é considerada um erro normal de digitação e deve ser ignorada.

Problema E – Evidências de um Merge

Limite de tempo: 0.5s

Limite de memória: 256MB

Eles tentaram esconder os seus sentimentos, mas acabaram deixando... evidências.

Dois arrays — `A` e `B` — tentam negar que combinam perfeitamente, mas todos os algoritmos apontam que eles foram feitos um para o outro.

Dados dois arrays, `A` e `B`, contendo inteiros ordenados de forma crescente, você deve uni-los em um único array, também ordenado — um verdadeiro merge de corações. Porém, cada número pode aparecer qualquer quantidade de vezes nos arrays de entrada, mas deve aparecer apenas uma vez na saída.

Sim, neste romance algorítmico, não há espaço para figurinhas repetidas.

Entrada

A primeira linha contém dois inteiros n e m ($1 \leq n, m \leq 10^5$), separados com um espaço, representando os tamanhos dos arrays `A` e `B`, respectivamente.

A segunda linha contém n inteiros $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^5$), separados com um espaço, ordenados de forma crescente.

A terceira linha contém m inteiros $b_1, b_2, \ldots, b_m$ ($1 \leq b_i \leq 10^5$), separados com um espaço, também ordenados de forma crescente.

Saída

Imprima uma única linha com os elementos da fusão entre `A` e `B`, ordenados de forma crescente, sem repetições.

Exemplo

Entrada	Saída
3 3 1 3 5 2 4 6	1 2 3 4 5 6
4 4 1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8
2 3 1 2 2 3 4	1 2 3 4

Notas

Os exemplos mostram a fusão de `A` e `B`, de acordo com as regras do enunciado. Especialmente, no terceiro exemplo, é possível ver que o número 2 não foi duplicado na saída.

"Há evidências demais no amor... mas neste algoritmo, basta uma para provar o sentimento."

Problema F – Fernando, Francisco e Frações

Limite de tempo: 1s

Limite de memória: 256MB

Fernando e Francisco são grandes amigos e colegas de classe. Ambos acabaram de entrar no ensino médio e estavam animados para estudar juntos. Porém, logo nas primeiras semanas, o professor de matemática decidiu aplicar uma lista de exercícios para avaliar o conhecimento prévio da turma — e, para a surpresa dos dois, a lista era cheia de expressões envolvendo frações e ordem das operações!

Sem saber muito bem por onde começar, Fernando e Francisco tentaram resolver os exercícios juntos, mas rapidamente perceberam que as expressões pareciam complicadas demais. Vendo a dificuldade da dupla, o professor sugeriu que eles pedissem uma ajudinha extra... para você!

Sua missão é simples: dado o número de operações e a expressão fornecida, ajude Fernando e Francisco a resolver corretamente as contas, respeitando a ordem das operações.

Entrada

A primeira linha contém um inteiro N e a segunda contém uma string S, de tamanho N, que representa a expressão matemática. É garantido que existem, no máximo, 30 frações e a expressão é formada apenas por sinais (+, -, * e /) e frações no formato a/b onde $-10 <= a <= 10$ e $1 <= b <= 10$. É garantido que existem, no máximo, 5 operações do tipo * e /.

Saída

A saída do problema deverá conter uma única fração, no formato a/b, onde $mdc(a, b) = 1$ e $b > 0$, que representa o resultado da avaliação da expressão numérica

Exemplo

Entrada	Saída
15 1/2 + 2/5 * 3/7	47/70
9 1/2 / 3/4	2/3
11 -1/2 + -2/4	-1/1
3 1/9	1/9

Problema G – Galinhas Globais

Limite de tempo: 1s

Limite de memória: 256MB

A música "Galinha Globais" faz parte do seriado "Cororicó" e ensina a pronúncia da palavra galinha em várias partes do mundo:

```
Nós somos galinhas globais
Estamos em todo o planeta
Estamos em todo o planeta

No Brasil a gente diz: galinha
Na Argentina a gente diz: gallina
No Brasil a gente diz: galinha
Na Argentina a gente diz: gallina

Na Alemanha somos huhn
Nos Estados Unidos somos hen ou chicken
Estamos em todo planeta
Estamos em todo planeta

Na França somos poule
Israel, tarnegolét
Egito somos farkha
Itália, gallina

Polônia, kura
Finlândia, kana
E no Japão é niwatori
Na Russia, kuritsa, kuritsa

Em Angola, em Angola
Tem a galinha da Angola que é cinza e branca e vive cantando assim:
Tô fraca, tô fraca, tô fraca
Tô fraca, tô fraca, tô fraca
```

Crie um programa que imprima a pronúncia da palavra galinha de acordo com um país informado.

Entrada

A entrada possui uma linha contendo um dos países citados na música, sem acentos ou cedilhas, mas repare que o nome de um país pode ser composto.

Saída

Imprima a pronúncia correta da palavra galinha conforme o país lido e a música, **sem acentos**. No caso de Angola, seu programa deverá imprimir "`to fraca`", e no caso dos Estados Unidos, "`hen ou chicken`".

Exemplo

Entrada	Saída
`Alemanha`	`huhn`
`Estados Unidos`	`hen ou chicken`
`Angola`	`to fraca`

Problema H – Hanimeitor

Limite de tempo: 1s

Limite de memória: 256MB

Toda maratona costuma disponibilizar um placar para que os participantes e o restante das pessoas possam acompanhar em tempo real a pontuação de cada equipe. Essa página é chamada de hanimeitor e ela mantém algumas informações relavantes. Dentre essas informações, há o tempo restante para o fim da maratona, a qual exibe um relógio que conta o tempo em segundos e fica piscando e atualizando a informação sobre o tempo restante.

0:13:35	0:13:35	0:13:35	0:13:35	0:13:35	0:13:35
0:13:36	0:13:36	0:13:36	0:13:36	0:13:36	0:13:36
0:13:36	0:13:37	0:13:37	0:13:37	0:13:37	0:13:37
0:13:38	0:13:38	0:13:38	0:13:38	0:13:38	0:13:38
0:13:39	0:13:39	0:13:39	0:13:39	0:13:39	0:13:39
0:13:39	0:13:39	0:13:40	0:13:40	0:13:40	0:13:40
0:13:41	0:13:41	0:13:41	0:13:41	0:13:41	0:13:41
0:13:42	0:13:42	0:13:42	0:13:42	0:13:42	0:13:42

Observando o comportamento do relógio, Monk ficou um pouco irritado com o seguinte fato: o relógio muda a cada segundo, mas a parte da animação que pisca a tela e exibe essa mudança não ocorre no mesmo ritmo. Isso faz com que em alguns momentos o relógio pisque de maneira desarmonizada, o que pode causar um certo desconforto visual.

Ao inspecionar o código fonte da página, Monk percebeu que, apesar de os efeitos começarem ao mesmo tempo, os valores de atualização do relógio e do efeito da animação são diferentes, e agora ele ficou curioso para saber quantas vezes, durante a competição, o efeito e o relógio piscam ao mesmo tempo.

Assim, Monk pediu a sua ajuda para que você escreva um programa que, dado o tempo de atualização do relógio e o tempo de atualização do efeito da animação, ele consiga saber quantas vezes o relógio e o efeito piscaram ao mesmo tempo durante a competição.

Entrada

A primeira linha da entrada contém um inteiro T ($1 \leq T \leq 100$) que representa o número de casos de teste. Cada caso de teste contém duas linhas. A primeira linha contém duas strings I e T, no formado $hh : mm$, que representam o horário inicial e o horário final da competição, respectivamente, onde hh é a hora (de 00 a 23) e mm é o minuto (de 00 a 59). Também é garantido que a competição acontece em um mesmo dia e $I <= T$. A segunda linha contém dois números inteiros S ($1 \leq S \leq 800$) e P ($1 \leq P \leq 800000$) que indicam o tempo de atualização do relógio em segundos e o tempo de atualização do efeito em milissegundos, respectivamente.

Saída

Para cada caso de teste, imprima uma linha contendo um inteiro que representa o número de vezes que o relógio e o efeito piscaram ao mesmo tempo durante a competição.

Exemplo

Entrada	Saída
1 13:00 17:00 1 2500	2880
1 14:00 14:01 6 1000	10