A. Angariando Padrões

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

j3r3mias é um cara que adora observar padrões. No seu tempo livre ele gosta de catalogar sequências com propriedades excêntricas.

Ultimamente, j3r3mias vem estudando o operador de XOR bit a bit. Dadas quatro sequências numéricas de igual tamanho em hexadecimal, ele quer saber o número de combinações em que o XOR bit a bit de quatro números, um de cada sequência, equivale a 0. j3r3mias até poderia resolver este problema sozinho, mas ele está ocupado com outras sequências interessantes. Ajude o nosso amigo a expandir o seu catálogo.

Input

A primeira linha da entrada possui um inteiro N ($1 \leq N \leq 10^3$), indicando o tamanho das sequências numéricas.

As próximas 4 linhas, contém, cada uma, N números hexadecimais, com no máximo 16 algarismos cada, separados por um espaço.

Output

Forneça como resultado o número de combinações, em decimal, cuja operação de XOR bit a bit resulta em 0.

Examples

input
1 5 3 a c
output
1

input
2 35 ad f 82 61 47 b3 95
output
0

input
2

```
35 ec
83 90
58 ad
b3 1b
```

output

```
1
```

Note

No primeiro exemplo, a única combinação possível possui a propriedade esperada.

No segundo exemplo, não existe combinação cujo XOR bit a bit dos números envolvidos resulte em 0.

No terceiro exemplo, a única combinação cujo o o XOR desses números resulta em zero é: 35, 83, ad e 1b.

B. Buracos de Coruja

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

O professor Mandelli é apaixonado por corujas. No momento, ele possui N corujas e pretende fazer de sua propriedade no Lago Oeste o habitat delas. Como as corujas são ingênuas, Mandelli resolveu ajudar na adaptação delas ao cavar N buracos, de modo que cada buraco seja ocupado por exatamente uma coruja. Os buracos estão dispostos de forma linear, um do lado do outro, com distâncias variáveis entre si.

Agora, Mandelli liberou as corujas para voarem até os buracos. Entretanto, como elas ainda estão tentando entender o propósito desses buracos, as corujas pousaram em posições aleatórias na região, para a frustração do querido professor. Agora ele tentará pensar em uma maneira de ensinar as corujas a pousarem corretamente nas posições em que existem buracos.

Considerando que uma coruja i pode permanecer na mesma posição c_i em que se encontra, ou voar para uma posição a direita $(c_i + 1)$, ou a esquerda $(c_i - 1)$, sua tarefa consiste em determinar o tempo mínimo para que a última coruja ocupe um buraco. Considere que todas as corujas partem de suas posições no mesmo instante e que cada uma gasta 1 minuto para voar uma unidade à esquerda ou à direita em relação à posição atual.

Input

A primeira linha da entrada contém um número inteiro N $(1 \leq N \leq 10^5)$, indicando a quantidade de buracos, como também a quantidade de corujas.

A segunda linha da entrada contém N números inteiros c_i $(0 \leq c_i \leq 2 \cdot 10^9)$ separados por espaço, indicando a posição inicial da i-ésima coruja.

A terceira linha da entrada contém N números inteiros b_i $(0 \leq b_i \leq 2 \cdot 10^9)$ separados por espaço, representando a posição do i-ésimo buraco.

Output

Imprima um número inteiro indicando o tempo mínimo (em minutos) para que a última coruja ocupe um buraco.

Examples

input
4 1 2 3 4 2 4 6 3
output
2

input
5 8 5 2 9 12 4 3 1 7 20
output
8

input
7 13 6 3 8 19 10 21 2 1 5 4 7 8 9
output
12

Note

No primeiro exemplo de teste, o tempo mínimo gasto para que a última coruja entre em um buraco ocorre quando:

- A coruja 1 ($c_i = 1$) entra no buraco 1 ($c_i = 2$);
- A coruja 2 ($c_i = 2$) entra no buraco 4 ($c_i = 3$);
- A coruja 3 ($c_i = 3$) entra no buraco 2 ($c_i = 4$);
- A coruja 4 ($c_i = 4$) entra no buraco 3 ($c_i = 6$).

Assim, a última coruja, 4, gasta 2 minutos para se deslocar de sua posição inicial.

C. Carma, Polícia!

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Os maratonistas estavam num rolê animado, com uma conversa matemática que zumbia como uma geladeira aos ouvidos dos demais. Inconformados, os "outros" fizeram uma denúncia anônima e os representantes da lei foram despachados para intervir...

Apesar dos policiais chegarem com certa truculência, o jovem caipira que liderava o grupo

acalmou os ânimos e propôs que você fizesse um programa para agilizar o processo. A abordagem segue o formato padrão: apenas confere se os presentes estavam na lista dos procurados ou não, prendendo ou liberando cada um conforme a situação.

Input

A entrada consiste de uma linha com um inteiro I ($0 \leq I \leq 1000$) seguida de I linhas contendo, cada uma, as infrações descritas por um inteiro p ($0 < p \leq 10^{11}$) referente ao documento de identificação do procurado e um valor inteiro a ($0 < a \leq 300$) do artigo que tipifica o crime no código penal, separados por espaço.

A próxima linha descreve a ordem policial na abordagem, uma string com não mais de 100 caracteres, contendo um único inteiro M ($0 < M \leq 1000$), precedido e sucedido por um espaço em branco, que indica a quantidade de maratonistas no grupo. Esta é seguida da interlocução "*Carma, Policia!*", e então de M linhas apresentando, cada uma, o identificador m ($0 < m \leq 10^{11}$) do maratonista sendo entrevistado.

Output

Para cada maratonista idôneo m, agradeça sua colaboração com a mensagem `"Grato, cidadao m."`. Para cada meliante infiltrado, dê a voz de prisão com a mensagem `"Teje preso, m!"` e liste suas ofensas ordenadas pela quantidade (decrescente) e, em caso de empate, pelo número do artigo que a tipifica (crescente), conforme os exemplos.

Examples

input

```
5
23 5
1 101
23 2
23 5
23 1
Perdeu ceis 4 tudo ai!
Carma, Policia!
2
1
23
7
```

output

```
Grato, cidadao 2.
Teje preso, 1!
- Art. 101: 1 ocorrencia(s).
Teje preso, 23!
- Art. 5: 2 ocorrencia(s).
- Art. 1: 1 ocorrencia(s).
- Art. 2: 1 ocorrencia(s).
Grato, cidadao 7.
```

input

```
2
13 57
13 171
```

```
Mao na cabeca, ambos os 2 ai!
Carma, Policia!
1337
23
```

output

```
Grato, cidadao 1337.
Grato, cidadao 23.
```

D. Dick Vigarista e a Corrida Maluca

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Dick Vigarista e seu cachorro Mutley não desistem de vencer uma Corrida Maluca, e novamente tentarão sabotar os demais competidores. A próxima etapa da Corrida Maluca acontecerá no Deserto Mal-Assombrado, onde há N pontos de controle, numerados de 1 a N.

Os competidores partirão todos do ponto 1 e seguirão uma rota composta por K pontos de controle $P_1, P_2, \ldots, P_K$, a qual inicia no ponto 1 e termina no ponto de chegada N. Dick pretende usar atalhos e rotas alternativas para chegar antes que os demais em algum ponto de controle $P_2, P_3, \ldots, P_{K-1}$ e preparar sua cilada. Ele pode até mesmo ativar o turbo de seu carro, o que reduz o tempo de trajeto entre dois pontos de controle pela metade. Estranhamente, até hoje não passou pela cabeça de Dick a ideia de usar os atalhos e o turbo para ir direto para a chegada...

Enquanto Dick combinava suas tramoias com Mutley, a largada foi dada, e quando os dois pilantras caíram em si, os competidores já estavam em P_2!

Auxilie o pobre trapaceiro e determine se ele é capaz de chegar em algum dos pontos de controle intermediários $P_2, P_3, \ldots, P_{K-1}$ antes dos demais competidores. Em caso afirmativo, descreva a rota que ele deve seguir e, se necessário, em que trecho o turbo deve ser ativado, lembrando que este turbo pode ser acionado uma única vez, em um ponto de controle, e que este turbo é desativado assim que ele atinge um novo ponto de controle.

Input

A primeira linha da entrada contém os inteiros N ($2 \leq N \leq 2 \times 10^5$) e M ($1 \leq M \leq 5 \times 10^5$), que representam o número de pontos de controle e o número de rotas que conectam dois pontos de controle.

As M linhas seguintes conterão, cada uma, três inteiros A, B ($1 \leq A, B \leq N, A \neq B$) e C ($1 \leq C \leq 10^9$), separados por um espaço em branco, os quais indicam que existe uma rota de mão dupla entre os pontos de controle A e B que pode ser percorrida em C minutos.

A linha seguinte contém o inteiro K ($2 \leq K \leq N$), que indica o número de pontos de controle da rota que será utilizada pelos competidores.

A última linha contém K inteiros distintos $P_1, P_2, \ldots, P_K$ ($1 \leq P_i \leq N$), separados por um espaço em branco, que indicam a rota seguida pelos competidores. É garantido que $P_1 = 1$, $P_K = N$ e que os competidores viajaram nesta rota, nesta ordem.

Output

Caso Dick Vigarista não consiga chegar a um dos pontos intermediários da rota dos competidores antes dos demais, imprima, em uma linha, a mensagem "`Nao`". Caso contrário, imprima, em uma linha, a mensagem "`Sim`".

Caso exista ao menos uma rota que permita a Dick chegar a ao ponto de controle P_r antes dos demais, imprima, em uma linha, o número de pontos de controle desta rota, incluindo o ponto de partida e o ponto de chegada. Por fim imprima, na linha seguinte, os pontos que compõe esta rota, na ordem que Dick deve visitá-los. Se Dick deve percorrer o trecho entre os pontos P e Q em velocidade normal, use a notação "`P -> Q`". Caso o trecho tenha que ser percorrido com o turbo ativado, use a notação "`P => Q`". Veja os exemplo para a formatação da rota.

Caso exista mais de uma rota que permita a Dick armar sua arapuca, imprima qualquer uma delas.

Examples

input
5 5 1 2 3 2 4 1 4 3 2 3 5 4 2 3 5 4 1 2 3 5
output
Sim 4 1 => 2 -> 4 -> 3

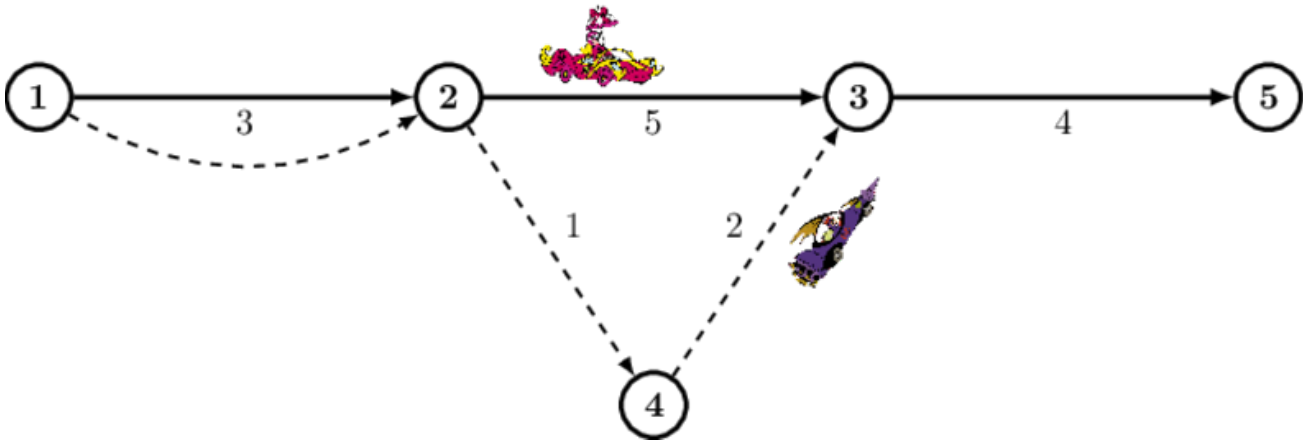
input
4 3 1 2 1 2 3 2 3 4 3 4 1 2 3 4
output
Nao

input
4 3 1 2 1 2 3 6 3 4 1 4 1 2 3 4
output

```
Sim
3
1 -> 2 => 3
```

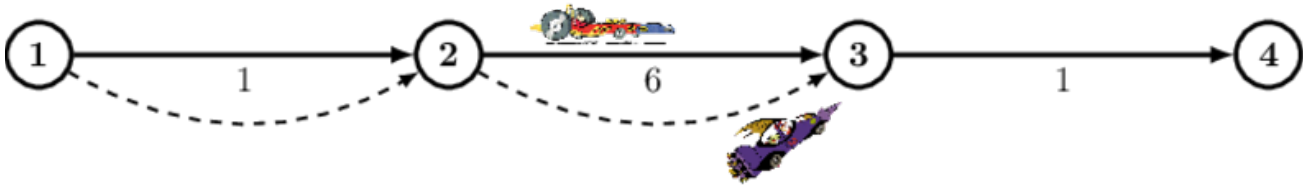
Note

No primeiro caso, os corredores seguiram pelas setas sólidas, chegando ao ponto de controle 3 em $3 + 5 = 8$ minutos. Já Dick partirá com 3 minutos de atraso, mas ao acionar o turbo, cruzará do ponto 1 ao ponto 2 em $1,5$ minutos. Segundo de 2 para 4 e daí para 3, ele chegará ao ponto 3 em $3 + 1,5 + 1 + 2 = 7,5$ minutos, tendo 30 segundo ainda para suas trapaças. Veja a figura abaixo:



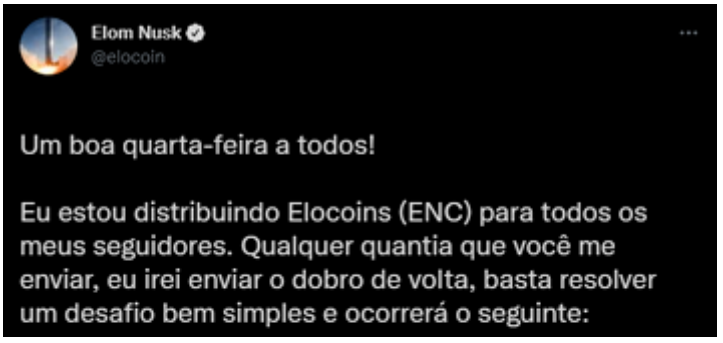
No segundo caso, mesmo com o auxílio do turbo, Dick não consegue chegar ao ponto 2 ou ao ponto 3 antes dos demais.

No terceiro caso, embora Dick parta com 1 minuto de atraso, o turbo permite vencer o trecho de 2 a 3 em apenas 3 minutos, de modo que ele chega ao ponto 3 em $1 + 1 + 3 = 5$ minutos, isto é, com 2 minutos de vantagem para os seus concorrentes. Veja a figura:



E. Elocoin

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output



Criptomoedas são o futuro, e mais uma vez o investimento de Elom Nusk lhe rendeu muito dinheiro! E dessa vez a *Elocoin* (moeda criada por Elom, cujo a sigla é ENC) rendeu tanto que ele resolveu distribuir entre todos os seus "seguidores". Para isso, basta a pessoa transferir uma quantia qualquer de ENC e resolver um desafio corretamente. Se der um pouco de sorte, ela receberá uma transferência com o dobro do valor de volta (em ENC, claro!).

O desafio a ser resolvido é o seguinte: a pessoa que fez a transferência receberá dois números, um em base decimal e outro em binária, e deve informar se o primeiro divide o segundo. Caso a divisão seja exata, a pessoa sortuda receberá a transferência com o dobro do valor.

Dessa forma, pediram a sua ajuda para escrever um programa que resolva um desafio enviado por Elom Nusk.

Input

A primeira linha contém um inteiro T ($1 \leq T \leq 1000$), que indica o número de casos de testes.

Cada caso de teste é composto por duas linhas, uma contendo 2 inteiros N ($1 \leq N \leq 10^5$) e S ($1 \leq S \leq 10^5$), separados por espaço em branco, em que N indica o divisor e S indica o número de dígitos do número binário. A outra linha contém um número binário B com S dígitos.

Output

Para cada caso de teste, seu programa deverá imprimir em uma linha a string `"To the moon!"`, se N dividir B, ou `"Phishing de criptomoeda."` caso contrário.

Examples

input
1 7 8 10101111
output
To the moon!

input
2 6 5 11000 6 5 11001

output

```
To the moon!
Phishing de criptomoeda.
```

Note

No primeiro teste, o único caso de teste contém o número 10101111_2 que representa o número 175, e este pode ser dividido exatamente por 7.

F. Fermentação de Pães

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Seu Heládio possui uma pequena fábrica de pães *gourmet*, em que cada pão possui um tempo ideal (em minutos) de fermentação. Na fábrica, existe uma "esteira de descanso" para fermentação de pães, que possui largura de exatamente um pão e uma profundidade que comporta 10^5 pães. A colocação e a retirada de pães é feita exclusivamente pelo acesso frontal da esteira.

Atualmente, a fábrica é capaz de produzir um pão por minuto, em que um pão recém-produzido é imediatamente colocado para fermentação nessa esteira, de forma que os demais pães (já em fermentação) são automaticamente empurrados para seu interior. Entretanto, devido aos tempos distintos de fermentação, pode ser difícil retirar os pães da esteira no momento ideal, uma vez que os pães ficam enfileirados, um atrás do outro, em seu interior.

O seguinte procedimento é adotado pelos funcionários da fábrica quando um pão é recém-produzido e necessita ser colocado em fermentação:

- primeiramente, os pães que atingirem o tempo ideal de fermentação, ou que passaram desse tempo, são retirados da esteira pelo acesso frontal, um a um, até que se encontre um pão ainda em fermentação;
- um pão recém-produzido é colocado normalmente na esteira caso sua fermentação seja concluída futuramente no mesmo momento ou antes do pão que estiver no acesso frontal da esteira;
- caso contrário, o funcionário retira M pães da esteira (ou menos, caso existam menos do que M pães na esteira), identifica e descarta os pães que atingiram a fermentação ideal ou que passaram esse prazo. Em seguida, os pães em fermentação e o pão recém-produzido são inseridos na esteira, de forma ordenada decrescente em relação ao momento em que a fermentação ideal de cada um seja concluída.

Depois que todos os pães foram colocados, os funcionários retiram os pães da esteira sempre aguardando o tempo ideal de fermentação do pão que está próximo ao acesso frontal.

Seu Heládio está preocupado com a produtividade da sua fábrica de pães e pede sua ajuda. Sabendo-se que em um dia qualquer N pães foram produzidos, determine a quantidade de pães que foram retirados da esteira de descanso no momento exato em que atingiram o tempo ideal de fermentação.

Input

A primeira linha da entrada contém dois números inteiros N e M $(1 \leq N \leq 2 \cdot 10^5, 1 \leq M \leq \min(50, N))$, separados por um espaço em branco, indicando o número de pães e o número máximo de pães a serem retirados, segundo o processo descrito no texto.

A segunda linha da entrada contém N números inteiros $a_1, \ldots, a_N$ $(1 \leq a_i \leq 5 \cdot 10^5)$ indicando o tempo ideal de fermentação do pão a_i. Vale ressaltar que o pão a_i fica pronto para ser colocado na esteira de descanso exatamente 1 minuto antes do pão a_{i+1}.

Output

Imprima um número inteiro indicando a quantidade de pães que foram retirados da esteira de descanso no momento exato em que atingiram o tempo ideal de fermentação.

Examples

input
5 1 1 1 1 1 1
output
5

input
5 1 1 2 3 4 5
output
4

input
6 2 14 6 3 13 11 5
output
5

input
8 2 10 20 30 80 70 40 60 50
output
6

Note

1. No primeiro exemplo de teste, todos os $N = 5$ pães possuem 1 minuto para uma fermentação ideal. Assim, a cronologia do processo consiste em:

1. O pão 1 é colocado na esteira;

2. O pão 1 está pronto, é retirado da esteira. Depois disso, o pão 2 é colocado;
3. O pão 2 foi fermentado e é retirado da esteira. O pão 3 é colocado na esteira;
4. O pão 3 está pronto e é retirado da esteira. Em seguida, coloca-se o pão 4 na esteira;
5. A fermentação do pão 4 foi concluída, é feita sua remoção, para depois colocar o pão 5 na esteira.

Por fim, o pão 5 está pronto e é removido. Assim, 5 pães puderam ser retirados no momento ideal da sua fermentação.

2. No segundo exemplo de teste, considerando que $M = 1$ (pode-se retirar apenas um pão durante a colocação de um novo pão na esteira), a cronologia do processo consiste em:

1. O pão 1 é colocado na esteira (ficará pronto aos 2 minutos);
2. O pão 1 está pronto e é retirado da esteira. O pão 2 (ficará pronto daqui a 4 min) é colocado na esteira;
3. O pão 3 será colocado na esteira, mas como ficará pronto daqui a 6 min, o pão 2 é retirado da esteira, coloca-se o pão 3 e retorna-se o pão 2 novamente;
4. O pão 2 está pronto e é retirado da esteira. Em seguida, antes de colocar o pão 4 (ficará pronto aos 8 min) na esteira , deve-se retirar o pão 3, colocar o pão 4 e depois retornar o pão 3 para a esteira;
5. O pão 5 (ficará pronto aos 10 min) será colocado na esteira. Retira-se o pão 3, coloca-se o pão 5 e, depois, o pão 3 é retornado para a esteira.

Aos 6 minutos, o pão 3 estará pronto e é retirado. Como o próximo pão a ser retirado é o pão 5, deve-se aguardar sua retirada do acesso frontal aos 10 minutos. Restou o pão 4 ficou pronto aos 8 minutos, mas como ele não pôde ser retirado da esteira, é descartado. A resposta é 4.

G. Grupos de controle

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Um instituto de pesquisa está conduzindo uma série de experimentos sociais, para os quais se voluntariaram N indivíduos. Na próxima fase dos experimentos os voluntários serão divididos em grupos de controle, e em cada grupo será aplicada uma questão onde cada membro deve optar por uma das duas respostas disponíveis. Para que cada pergunta tenha uma resposta majoritária, o número M de indivíduos em cada grupo deve ser ímpar.

Você foi designado para montar os grupos de controle. A primeira tarefa é determinar o número M de membros em cada grupo de controle, de tal forma que o número de grupos distintos seja mínimo. Cada voluntário deve ser alocado a um único grupo, todos os voluntários devem participar de algum grupo de controle e todos os grupos devem ter o mesmo número de membros.

Input

A entrada é composta por uma única linha, que contém o inteiro N ($1 \leq N \leq 10^{18}$).

Output

Imprima, em uma linha, o número M de membros de cada grupo de controle que minimiza a quantidade de grupos.

Examples

input
2
output
1

input
3
output
3

input
10
output
5

Note

No primeiro caso, serão dois grupos de controle, cada um com um único membro.

No segundo caso haverá um único grupo de controle, com todos os três voluntários.

No terceiro caso teremos dois grupos de controle, com 5 membros cada. Em relação à restrição quanto ao número de participantes, seria possível formar 10 grupos com um indivíduo cada, mas este arranjo não minimizaria o número de grupos de controle.

H. Hagar.io

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

`Hagar.io` é um jogo online que fez muito sucesso por meados de 2015. Nesse jogo, você controla uma célula do tipo *hagar* e o objetivo é fazê-la crescer, absorvendo células de outros

jogadores e destruindo grupos de células adversárias.

Em uma partida, cada jogador pertence a um único grupo e inicia com 3 pontos de vida. Se um grupo chega a 0 pontos de vida, ele é eliminado. Durante a partida, existem dois tipos de ações: **encostar** e **atacar**. Quando ocorre uma ação de encostar entre dois grupos distintos, se ambos possuem a mesma quantidade de vida, eles se repelem e cada grupo perde 1 ponto de vida; caso contrário, os grupos se fundem e esta junção resultante possui a soma dos pontos de vida de ambos. Note que, apesar do grupo com maior número de pontos de vida absorver o grupo menor, este não deixa de existir e sim faz parte de um grupo maior composto pela junção dos dois grupos! Já a ação de atacar ocorre a distância: um grupo ataca e o outro contra-ataca como mecanismo de defesa. O grupo com o maior número de pontos de vida destrói o grupo menor e seus pontos de vida vão a zero. E, por último, em casos de empate, elementos do mesmo grupo se atacando ou um ataque a um grupo já removido do jogo, nada acontece (feijoada).

Assim, seu objetivo é, dadas as ações de uma partida de Hagar.io, informar o total de ações de ataque que um determinado grupo venceu durante a partida.

Input

A primeira linha da entrada contém 3 inteiros N ($1 \leq N \leq 5 \cdot 10^5$), A ($0 \leq A \leq 5 \cdot 10^5$) e J ($1 \leq J \leq N$), separados por espaço em branco, onde N indica o número de jogadores, A indica o número de ações da partida e J é a identificação do jogador cujo o número de batalhas vencidas deverá ser contabilizado.

Cada uma das A linhas seguintes contém três inteiros T ($T \in \{1, 2\}$), X ($1 \leq X \leq N$) e Y ($1 \leq Y \leq N$), separados por espaço. T indica o tipo de ação podendo ser encostar (1) ou atacar (2), X indica o grupo que está encostando/atacando e Y o grupo que está sendo encostado/atacado.

Output

A saída deve conter um inteiro com o número de vitórias que o jogador J obteve durante os ataques envolvendo o seu grupo.

Examples

input
6 6 1 1 1 4 1 1 3 2 3 5 2 4 6 2 1 6 2 3 2
output
3

input
10 10 10 1 1 10

```
1 1 2
1 2 3
1 3 4
1 4 5
2 1 6
2 1 7
2 1 8
1 1 10
2 1 9
```

output

```
1
```

Note

No primeiro exemplo, o jogador pertence ao grupo 1 e os grupos podem ser vistos da seguinte forma:

$\{(3)\ \mathbf{1}\}, \{(3)\ 2\}, \{(3)\ 3\}, \{(3)\ 4\}, \{(3)\ 5\}, \{(3)\ 6\}$

Na primeira ação, os grupos 1 e 4 encostam e se repelem (perdendo um ponto de vida cada):

$\{(2)\ \mathbf{1}\}, \{(3)\ 2\}, \{(3)\ 3\}, \{(2)\ 4\}, \{(3)\ 5\}, \{(3)\ 6\}$

Na segunda ação, os grupos 1 e 3 encostam e o grupo 3 absorve o grupo 1:

$\{(3)\ 2\}, \{(5)\ 3, \mathbf{1}\}, \{(2)\ 4\}, \{(3)\ 5\}, \{(3)\ 6\}$

Na terceira ação, os grupos 3 e 5 batalham entre si e o grupo 3 vence (+1 ponto para o jogador):

$\{(3)\ 2\}, \{(5)\ 3, \mathbf{1}\}, \{(2)\ 4\}, \{(3)\ 6\}$

Na quarta ação, os grupos 4 e 6 batalham entre si e o grupo 6 vence:

$\{(3)\ 2\}, \{(5)\ 3, \mathbf{1}\}, \{(3)\ 6\}$

Na quinta ação, os grupos 1 e 6 batalham entre si e o grupo 1 vence (+1 ponto para o jogador):

$\{(3)\ 2\}, \{(5)\ 3, \mathbf{1}\}$

Na sexta ação, os grupos 3 e 2 batalham entre si e o grupo 3 vence (+1 ponto para o jogador):

$\{(5)\ 3, \mathbf{1}\}$

O jogador 1 pertencia ao grupo de três batalhas a qual saiu vencedor e a resposta final é 3.

I. Internacional

time limit per test: 1.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Você foi contratado por um serviço de inteligência (cujo nome não podemos citar aqui, por razões de confidencialidade) para rastrear a movimentação de N pessoas ao redor do mundo.

As posições iniciais delas são conhecidas: a i-ésima pessoa está no país a_i.

Como todos sabem, existem K companhias aéreas no mundo, e cada uma oferece diversos voos internacionais. Cada companhia pode oferecer zero ou mais voos saindo de um determinado país, mas **não** é garantido que se o voo de A a B é ofertado, o de B para A também será. Além disso, no máximo um voo de A para B é ofertado **por companhia**.

Um ponto interessante do seu trabalho é o seguinte: imagine que uma pessoa esteja no país A e pegue um voo da companhia 1. Sabendo que essa companhia possui os voos de A para B e de A para C, é impossível determinar exatamente em qual país a pessoa está após o voo, mas é possível deduzir que ela pode estar ou em B ou em C, não há nenhuma outra possibilidade.

Sua missão consistirá em rastrear as N pessoas, fazendo esse tipo de dedução. Mais especificamente, serão dados Q eventos dos tipos

- 1 l r k: você recebe uma informação dizendo que as pessoas $l, l+1, \ldots, r$ pegaram um voo da companhia k. Caso um indivíduo esteja em um país que não possui voo de saída dessa companhia, **ele permanece onde estava**, sem pegar avião algum.
- 2 i: você deve reportar para seus superiores uma lista de países em que a i-ésima pessoa pode estar.

Input

Na primeira linha da entrada são dados dois inteiros N e M: a quantidade de pessoas rastreadas e o número de países existentes no mundo ($1 \leq N \leq 10^5, 2 \leq M \leq 10$).

A segunda linha possui N inteiros $a_1, a_2, \ldots, a_n$, onde a_i representa o país em que a i-ésima pessoa está localizada inicialmente ($1 \leq a_i \leq M$)

Na terceira linha são dados E e K: o número de voos e o número de companhias aéreas ($1 \leq E \leq K \cdot M \cdot (M-1), 1 \leq K \leq 100$).

Cada uma das próximas E linhas possui 3 inteiros a b c, que indicam que há um voo saindo de a e chegando em b, ofertado pela companhia aérea c.

Depois disso, há uma linha com um inteiro Q, a quantidade de eventos que você deve processar ($1 \leq Q \leq 5000$).

Por fim, são dadas Q linhas, que contém os eventos em ordem cronológica. Um evento do tipo 1 é dado no formato 1 l r c ($1 \leq l, r \leq N, 1 \leq c \leq K$), enquanto eventos do tipo 2 têm o formato 2 i ($1 \leq i \leq N$).

Output

Para cada query do tipo 2, imprima um inteiro p (o número de países em que a pessoa i pode estar) e depois p inteiros (os países). Os países podem ser impressos em qualquer ordem, mas nenhum país deve aparecer mais de uma vez por linha.

Example

input
2 4

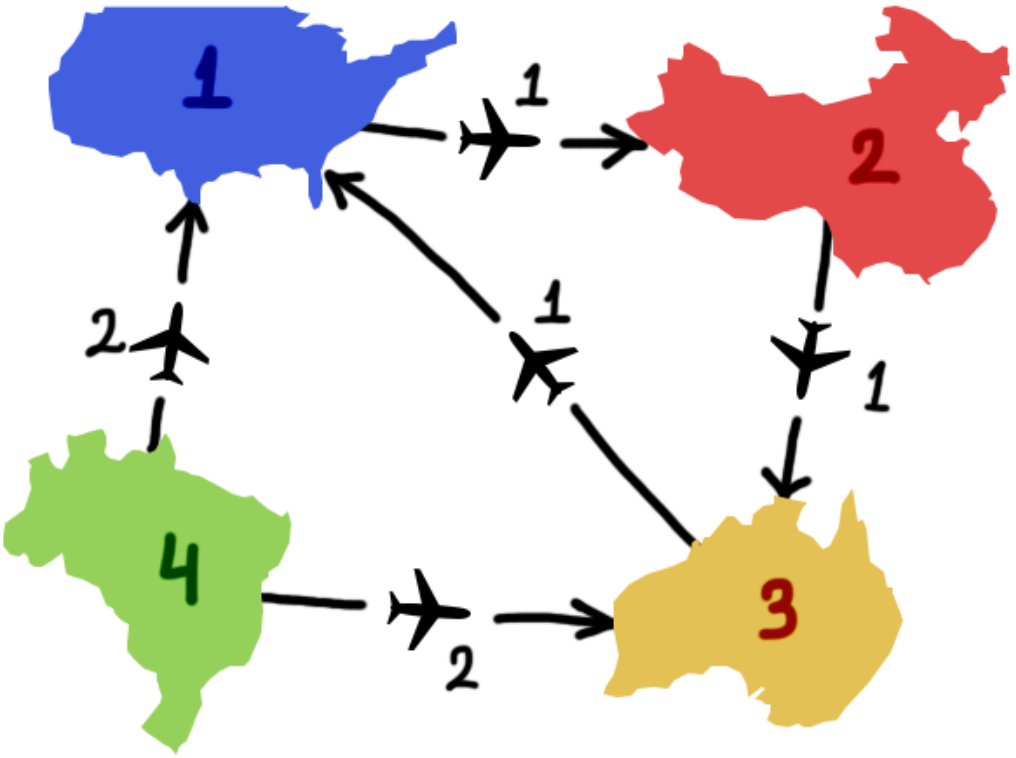
```
4 2
5 2
1 2 1
2 3 1
4 1 2
3 1 1
4 3 2
7
2 1
1 1 2 2
2 1
2 2
1 1 2 1
2 1
2 2
```

output

```
1 4
2 1 3
1 2
2 1 2
1 3
```

Note

No primeiro exemplo, temos 4 países e 5 voos, mostrados na imagem abaixo, onde os números ao lado dos aviões representam a companhia que oferece aquele voo.



Analisando os 7 eventos dados, temos:

- 2 1: A primeira pessoa só pode estar no país 4, porque não pegou nenhum voo
- 1 1 2 2: Ambos os indivíduos tentam pegar um voo da companhia 2.
- 2 1: A pessoa 1 pode ter pego o voo $4 \rightarrow 1$ ou $4 \rightarrow 3$ no último evento, portanto ela

está em 1 ou em 3.
- 2 2: No último evento do tipo 1, a pessoa número 2 tenta pegar um voo da companhia 2, porém o país que ela está não possui nenhum voo de saída dessa companhia, portanto ela permanece no país 2.
- 1 1 2 1: Ambas pessoas pegam um voo da companhia 1.
- 2 1: Note que o indivíduo número 1 podia estar no país 1 ou no 3 anteriormente, mas não sabemos qual. Caso ele esteja em 1, pegará o voo $1 \rightarrow 2$, mas caso esteja em 3, pegará $3 \rightarrow 1$. Sendo assim, esse indivíduo pode estar em 1 ou 2 agora.
- 2 2: Como a pessoa número 2 estava no país 2, ela pega o voo $2 \rightarrow 3$ e agora, com certeza, está no país 3.

J. Já Apaguei

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

João Lonèn é curador de uma galeria de artes, que é composta de vários núcleos conectados por corredores. A partir de qualquer núcleo é possível chegar a outro por uma sequência de corredores e, para melhorar a experiência dos transeuntes, é garantido que eles não podem andar em círculos, isto é, para chegarem a um núcleo previamente visitado, eles devem percorrer de volta todo o caminho traçado até então.

Preocupado com o consumo de eletricidade de sua galeria, João decidiu bolar uma estratégia que não prejudicasse a exposição: não permitir que a iluminação de núcleos adjacentes ficassem inativas simultaneamente. Como cada núcleo possui uma necessidade específica de consumo elétrico, e João não é muito bom em contas, ele pediu a sua ajuda para calcular quais os núcleos cuja energia elétrica deverá ser desativada com um grito de "já apaguei!", visando maximizar a economia de energia, a qual, por sua vez, corresponde à soma das necessidades energéticas dos núcleos desligados.

Input

A primeira linha da entrada possui um inteiro N ($1 \leq N \leq 10^5$), indicando o número de núcleos.

A próxima linha possui N números inteiros positivos, separados por espaço, indicando o consumo C_i de cada núcleo ($1 \leq C_i \leq 10^9$).

As próximas $N-1$ linhas, possuem um par de inteiros (u, v) indicando que existe um corredor conectando os núcleos u e v ($1 \leq u, v \leq N, u \neq v$).

Output

Seu programa deverá imprimir duas linhas: a primeira linha informará o máximo de energia que pode ser economizada por João, enquanto a segunda disponibilizará uma lista de núcleos, separados por espaço, que deverão ser desativados.

Em caso de múltiplas respostas possíveis, o juiz aceitará qualquer uma.

Examples

input

```
4
2 5 3 2
1 2
2 3
3 4
```

output

```
7
2 4
```

input

```
5
10 2 3 4 5
1 2
1 3
1 4
1 5
```

output

```
14
2 3 4 5
```

input

```
6
2 3 1 3 1 2
1 2
3 2
2 4
4 5
4 6
```

output

```
6
2 5 6
```

Note

No primeiro exemplo, João deve gritar "já apaguei" para os núcleos 2 e 4, cujo consumo total é 7.

No segundo exemplo, João prefere deixar o núcleo central ligado e desligar todos os seus adjacentes, cujo consumo total é 14.

Considerando o terceiro exemplo, uma possível solução é deixar os núcleos 2, 5 e 6 desligados. Outra estratégia válida é desativar os núcleos 1, 3 e 4.

K. Karen e os dados

time limit per test: 1.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Karen ganhou de presente alguns dados peculiares idênticos. Ao contrário de dados comuns, sempre que alguém os joga qualquer combinação única de valores é equiprovável. Karen passou a jogá-los como passatempo. Certo dia, todos os dados caíram em números diferentes.

Logo pensou "Uau! Qual a chance disso acontecer?", mas, como este problema era muito simples para Karen, ela logo dificultou. Agora ela quer saber a chance de que o resultado tenha exatamente K valores distintos, para todo inteiro K entre 1 e N.

Como a resposta pode ser muito grande, imprima-a módulo $10^9 + 7$. Se a resposta têm formato $\frac{p}{q}$ para p e q inteiros, imprima um inteiro x $(0 \leq x < 10^9 + 7)$ tal que $x \times q \equiv p \mod 1000000007$. É garantido, para todo conjunto de testes do problema, que $q \not\equiv 0 \mod 1000000007$.

Input

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

As próximas T linhas contém, cada uma, dois inteiros N $(1 \leq N \leq 1000)$ e M $(\max(N, 4) \leq M \leq 10^5)$, que representam o número de dados e a quantidade de faces em cada um, respectivamente.

É garantido que a soma de N para todos os casos de uma entrada é, no máximo, igual a 10^4.

Output

Para cada caso de teste imprima N inteiros. O k-ésimo inteiro deve ser a probabilidade de que um lançamento resulte em exatamente k valores distintos.

Example

input
2 2 4 3 5
output
800000006 200000002 142857144 571428576

```
285714288
```

Note

No primeiro caso de teste temos 2 dados com 4 faces cada um. As possíveis combinações são: {1, 1}, {1, 2}, {1, 3}, {1, 4}, {2, 2}, {2, 3}, {2, 4}, {3, 3}, {3, 4} e {4, 4}. Portanto a chance de ter exatamente um valor único é $\frac{4}{10}$ e a chance de ter dois exatemente dois valores distintos é $\frac{6}{10}$. Veja que

$800000006 \times 10 \equiv 4 \mod 1000000007$

e

$200000002 \times 10 \equiv 6 \mod 1000000007$

No segundo caso de teste, as chances são $\frac{5}{35}$, $\frac{20}{35}$ e $\frac{10}{35}$, respectivamente. Observe que

$142857144 \times 35 \equiv 5 \mod 1000000007$,

$571428576 \times 35 \equiv 20 \mod 1000000007$

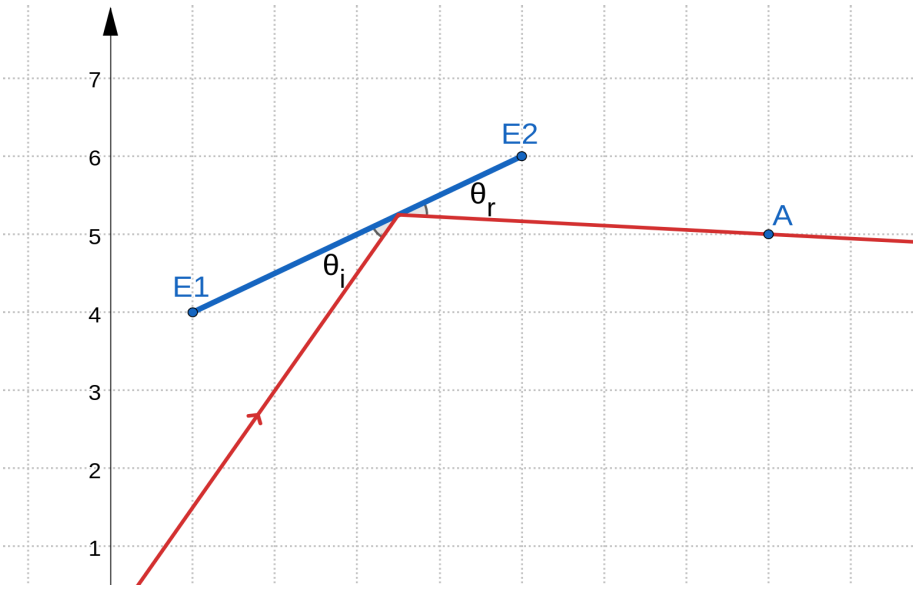
e

$285714288 \times 35 \equiv 10 \mod 1000000007$

L. Laser do Léo

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Leonardo acaba de comprar um laser e decidiu trollar Alberto, um de seus amigos, ao apontar seu laser para ele. Ambos estão na mesma sala, então para que não fique óbvio quem está com o laser, ele decidiu não apontar diretamente, mas através de um espelho. A sala pode ser considerada como um plano cartesiano 2D infinito, em que Leonardo e Alberto são os pontos com coordenadas (x_L, y_L) e (x_A, y_A), respectivamente, e o espelho, que reflete para ambos os lados, é representado por um segmento de reta unindo os pontos (x_{E1}, y_{E1}) e (x_{E2}, y_{E2}).



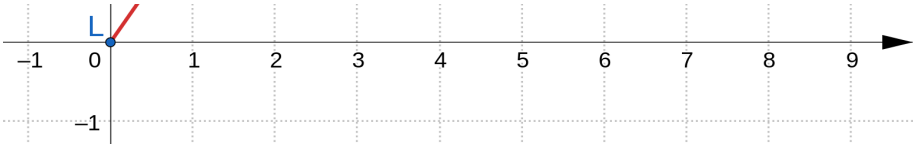


Imagem do primeiro caso de teste.

Leonardo é estudioso e sabe que o espelho em questão respeita as leis da física, ou seja, o ângulo de incidência é sempre igual ao ângulo de reflexão ($\theta_i = \theta_r$).

O espelho divide o plano em dois semi-planos. É garantido que Leonardo e Alberto estão localizados no mesmo semi-plano e que nenhum dos dois está em posição colinear com o espelho, e é possível desconsiderar Alberto e Leonardo como obstáculos, ou seja, o laser simplesmente os atravessa.

Ajude Leonardo a saber se é possível atingir Alberto com o laser através do espelho.

Input

A primeira linha da entrada contém um inteiro T ($1 \leq T \leq 10^4$) indicando o número de casos de testes.

A primeira linha de cada caso de teste contém quatro inteiros x_L, y_L, x_A, y_A ($-500 \leq x_L, y_L, x_A, y_A \leq 500$), representando as coordenadas de Leonardo e Alberto, respectivamente.

A segunda linha contém outros quatro inteiros x_{E1}, y_{E1}, x_{E2}, y_{E2} ($-500 \leq x_{E1}, y_{E1}, x_{E2}, y_{E2} \leq 500$), representando as coordenadas dos extremos do espelho.

É garantido que nenhum dos quatro pontos L, A, $E1$ e $E2$ ocupam a mesma posição no espaço.

Output

Para cada caso de teste, imprima "De onde veio isso?" caso Leonardo consiga acertar Alberto com o laser através do espelho, e "Leo, eu estou te vendo..." caso contrário.

Examples

input
2 0 0 8 5 1 4 5 6 0 0 9 7 1 4 5 6
output
De onde veio isso? Leo, eu estou te vendo...

input
1 3 3 1 3 1 1 3 1
output

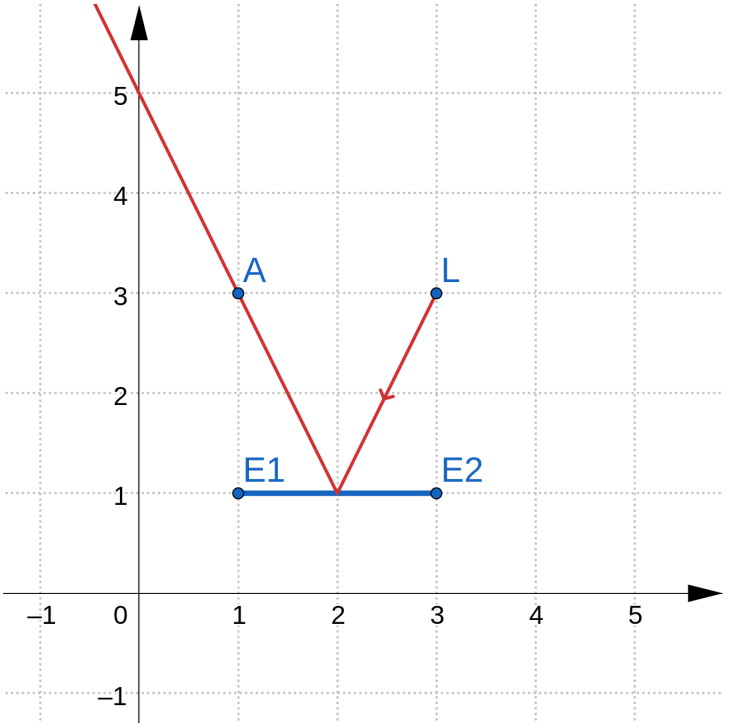
```
De onde veio isso?
```

input
2 5 5 5 10 0 0 10 0 5 10 5 5 0 0 10 0
output
De onde veio isso? De onde veio isso?

input
1 1 1 1 -1 0 0 0 10
output
De onde veio isso?

Note

No segundo caso de teste, é possível acertar Alberto com o laser, como pode ser visto na imagem abaixo:



No terceiro caso de teste, também é possível acertar Alberto com o laser, pois de acordo com o enunciado, Leonardo e Alberto não são considerados obstáculos pelo laser.

M. Mariana e os cursos

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input
output: standard output

Um portal de cursos preparatórios para concursos públicos online está oferecendo descontos em todos os seus N cursos. Mariana deseja aproveitar esta oferta, pois está se preparando para um certame que acontecerá dentro de T dias. Cada curso tem duração de t dias e está sendo ofertado com um desconto de d reais. Cada curso só pode ser adquirido uma única vez.

Mariana pretende assistir todos os cursos que comprar, integralmente, um de cada vez, antes do certame. Cumprida esta condição, ela pretende acumular o maior total de descontos possível. Auxilie Mariana elencando os M cursos que ela deve adquirir, de modo que ela consiga assistir todos antes do concurso e que a dê o maior total de descontos possível.

Input

A primeira linha da entrada contém os valores dos inteiros N e T ($1 \leq N, T \leq 10^3$), separados por um espaço em branco.

Cada uma das N linhas seguintes contém as informações do i-ésimo curso ($1 \leq i \leq N$): sua duração t_i ($1 \leq t_i \leq 10^3$) e o seu desconto d_i ($1 \leq d_i \leq 10^5$), em reais, separados por um espaço em branco.

Output

Imprima, em uma linha, o número M de cursos que Mariana deve adquirir. Em seguida imprima, em uma linha, os identificadores dos cursos a serem adquiridos, em qualquer ordem, separados por um único espaço em branco. Caso exista mais de um conjunto de cursos que atenda as condições impostas por Mariana, imprima qualquer um deles.

Examples

input
3 10 8 100 3 50 4 60
output
2 3 2

input
3 10 8 120 3 50 4 60
output
1 1

input
5 20

```
7 50
5 80
8 90
10 120
9 100
```

output

```
3
3 2 1
```

Note

No primeiro caso, Mariana deve comprar os cursos 2 e 3. Eles levarão 7 dias para serem assistidos, o que é possível dentro das 10 dias que Mariana tem, e ela terá um desconto total de 110 reais. Se ela adquirisse apenas o primeiro curso, seria possível assisti-lo (8 dias), mas o desconto total seria de apenas 100 reais.

No segundo caso, o curso 1 sozinho gera o maior desconto possível (120 reais). Veja que ela não pode adquirir, por exemplo, os cursos 1 e 2, uma vez que eles tem duração total de 11 dias, e Mariana dispõe de apenas 10 dias para assisti-los.

No terceiro caso, os três primeiros cursos totalizam 20 dias, com um desconto de 220 reais. Veja que os cursos 4 e 5 também formam uma solução válida: eles totalizam 19 dias, e geram os mesmo 220 reais de desconto.

N. Nario Party

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Enzo, Quirino e Faustino, após um produtivo dia de treino, decidiram jogar o famoso jogo da Mintendo chamado Nario Party. O jogo funciona por turnos, e cada jogador controla um personagem no mapa, que é representado por um grid quadrado de lado $2n + 1$, cujas coordenadas inteiras x, y tais que $-n \leq x, y \leq n$.

Os jogadores estão na última rodada e é a vez de Enzo jogar. A posição do personagem de Enzo é exatamente $(0, 0)$, e ele vai lançar um dado de d lados para determinar a quantidade de posições que ele poderá andar (os valores dos dados são $1, 2, \ldots, d$). Existem m moedas localizadas no mapa, e para coletar uma moeda o personagem deve terminar seu turno exatamente na posição da moeda.

Enzo então se faz a seguinte pergunta: "se eu lançar o dado e obter o valor x, quantas moedas são alcançáveis?". Uma moeda é dita alcançável se é possível coletá-la movendo exatamente x vezes no mapa. Motivado por essa pergunta, ele pede para você que faça um programa que calcula o **valor esperado** da quantidade de moedas alcançáveis ao lançar um dado de d lados.

Formalmente, se a posição do personagem é (x, y) então em um movimento o personagem pode ir para $(x + 1, y)$, $(x - 1, y)$, $(x, y + 1)$, $(x, y - 1)$. Considere que cada valor no dado de d faces tem a mesma probabilidade de ser obtida. Note também que é possível ter **mais de uma moeda na mesma posição**.

Input

A primeira linha contém 3 inteiros n, m, d ($1 \leq n \leq 10^9; 1 \leq m \leq 10^5; 1 \leq d \leq 2 \cdot 10^9$), separados por um espaço em branco – a dimensão do mapa, o número de moedas restantes e o número de faces do dado, respectivamente.

As próximas m linhas contém dois inteiros x_i e y_i cada ($-n \leq x_i, y_i \leq n; (x_i, y_i) \neq (0, 0)$), separados por um espaço em branco – a posição da i-ésima moeda.

Output

Imprima um único inteiro — o valor esperado da quantidade de moedas alcançáveis a Enzo ao lançar um dado de d lados. Sua resposta será considerada correta se o erro absoluto ou relativo não exceder 10^{-6}. Formalmente, se sua resposta é a e a do juiz é b, sua resposta será aceita se $\frac{|a-b|}{max(1,b)} \leq 10^{-6}$.

Example

input
4 4 8 1 1 1 2 -3 2 4 4
output
1.250000000

Note

Para o caso de teste 1, vamos analisar todas as possibilidades de valores x ao rolar o dado de $d = 8$ faces e calcular quais posições podemos alcançar.

- $x = 1$: nenhum;
- $x = 2$: posição $(1, 1)$;
- $x = 3$: posição $(1, 2)$;
- $x = 4$: posição $(1, 1)$;
- $x = 5$: posições $(1, 2)$ e $(-3, 2)$;
- $x = 6$: posições $(1, 1)$;
- $x = 7$: posições $(1, 2)$ e $(-3, 2)$;
- $x = 8$: posições $(1, 1)$ e $(4, 4)$.

Então o valor esperado é $10/8 = 1.25$.

O. Obtendo Informações

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Os estagiários sofrem bastante para desempenhar suas tarefas quando estão lidando com uma

tecnologia que ainda não dominam. Frequentemente podemos vê-los andando de um lado para outro solicitando apoio dos seus supervisores.

Alguns supervisores são interessados e prestativos e conseguem atender o estagiário prontamente, mas, infelizmente, nem todos são solícitos para ajudar os estagiários, muitos preferem "empurrar o problema com a barriga" e encaminhar o estagiário para outro supervisor. Neste processo, o estagiário segue as recomendações, saltando de supervisor a supervisor, até voltar ao original, que lhe fez o primeiro encaminhamento. Para que este processo seja disfarçado pelos supervisores "picaretas", eles combinaram de sempre indicar supervisores os quais ainda não foram indicados.

Calcule o número máximo de supervisores que podem ser consultados por um estagiário quando este inicia o processo de obtenção de informações por um supervisor arbitrário.

Input

A primeira linha da entrada possui um inteiro N ($1 \leq N \leq 10^5$), indicando o número de supervisores.

A próxima linha possui N inteiros $X_1, \ldots, X_N$ ($1 \leq X_i \leq N$), separados por um espaço, em que cada X_i contém a identificação do próximo supervisor indicado pelo i-ésimo supervisor. Caso $X_i = i$, então o i-ésimo supervisor atende o estagiário solicitamente, caso contrário, o i-ésimo supervisor indica o supervisor X_i ao estagiário.

Output

Seu programa deverá imprimir em uma única linha um inteiro com o número máximo de supervisores que podem ser consultados por um estagiário.

Examples

input
5 2 3 4 5 1
output
5

input
5 1 2 3 4 5
output
1

input
5 3 1 2 5 4
output
3

Note

No primeiro exemplo, o estagiário precisa consultar 5 supervisores no máximo até retornar ao original, independentemente de qual supervisor foi acionado primeiro.

No segundo exemplo, todos os supervisores são solícitos e atendem o estagiário, portanto, o número máximo de supervisores consultados neste cenário é 1.

No terceiro exemplo, ao consultar os supervisores de 1 a 3, o estagiário passa por 3 supervisores até retornar ao original.