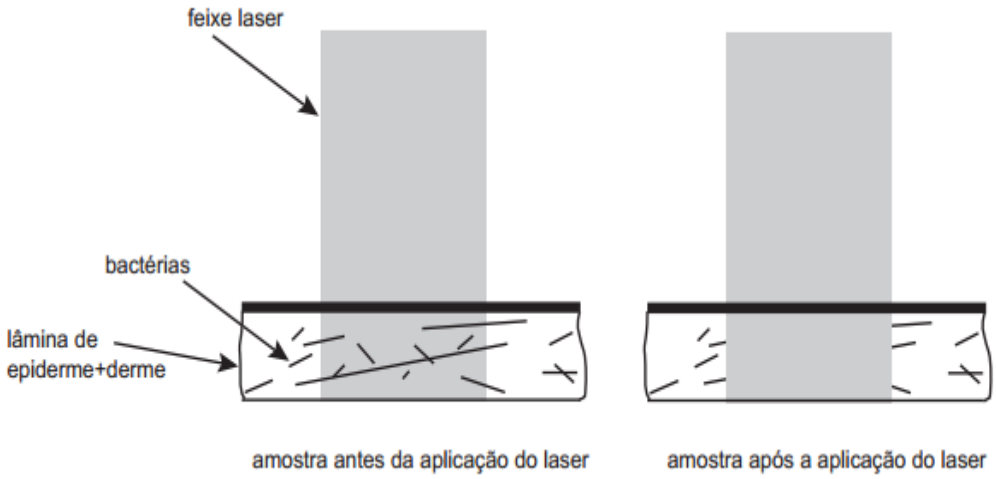
A. Tratamento a laser

Time limit: 0.565s Memory limit: 1536 MB

Um novo tratamento a laser está sendo pesquisado, para eliminar as bactérias causadoras de uma doença de pele que tem acometido muitas pessoas ultimamente. Essa doença, denominada Erictea Demasiana, é causada por uma bactéria de forma retilínea, e acomete pessoas que deixam de tomar sol por muito tempo. As bactérias instalam-se em colônias na derme, abaixo da epiderme, e causam uma coceira insuportável no local infectado.

O novo tratamento utiliza um laser de hélio-neônio que destrói completamente qualquer pedaço de bactéria que o laser atinja. No entanto, o laser pode danificar a derme, e por isso seu uso deve ser feito com parcimônia.

Para comprovar a eficácia do tratamento uma pesquisa foi encomendada. Para a pesquisa uma fina lâmina de pele contaminada é fotografada com um microscópio eletrônico de forma a retratar a infestação. Pela fotografia pode-se identificar perfeitamente as bactérias instaladas na lâmina, como ilustra a figura abaixo.



Uma peculiaridade interessante da Eritea Demasiana, que pode ser verificada na figura, é que ela nunca se posiciona verticalmente em relação à epiderme, de forma que o feixe de laser nunca é paralelo a uma bactéria.

Um feixe de laser é então aplicado perpendicularmente à lâmina, como ilustrado na figura. O feixe destrói todos os segmentos de bactéria que atinge. Por exemplo, na figura acima, após a aplicação do laser há 11 bactérias ou segmentos de bactéria remanescentes.

Sua tarefa é escrever um programa que responda a consultas para determinar, para diferentes feixes de laser, quantas bactérias ou segmentos de bactéria ainda permanecem na amostra.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A primeira linha do conjunto de testes contém dois números inteiros N e C que indicam respectivamente o número de bactérias presentes na amostra ($1 \leq N \leq 10^5$) e o número de consultas que serão realizadas ($1 \leq C \leq 10^5$). Cada uma das N linhas seguintes contém quatro números inteiros X_1, Y_1, X_2, Y_2 representando os dois extremos de uma bactéria, sendo (X_1, Y_1) as coordenadas de um extremo e (X_2, Y_2) as coordenadas do outro extremo ($-10^9 \leq X_1, Y_1, X_2, Y_2 \leq 10^9$ e $X_1 \neq X_2$).

Cada uma das C linhas seguintes representa uma consulta, e contém dois números inteiros X_i e X_f que indicam respectivamente a coordenada X inicial e final do feixe de laser ($-10^9 \leq X_i < X_f \leq 10^9$).

As consultas não são cumulativas: o resultado da consulta independe de consultas anteriores (em outras palavras, as consultas são sempre relativas às bactérias presentes inicialmente na amostra).

Saída

Seu programa deve imprimir, na saída padrão, uma linha para cada uma das C consultas da entrada. Cada linha deve conter um inteiro, o número total de bactérias e segmentos de bactérias remanescentes na amostra após a aplicação do laser.

Exemplo

Entrada:

```
1 4
1 0 4 0
0 2
3 5
2 3
0 5
```

Saída:

```
1
1
2
0
```

Entrada:

```
2 6
2 0 4 0
2 0 4 0
0 1
1 2
2 3
3 4
4 5
5 6
```

Saída:

```
2
2
2
2
2
2
```

Entrada:

```
2 3
0 0 3 5
3 5 0 2
0 3
1 2
2 7
```

Saída:

```
0
4
2
```

Seletiva IOI 2008

B. Poodle

Time limit: 1s Memory limit: 1536 MB

Maria vive perdendo coisas dentro de casa. Desde pequena, tudo em que ela põe a mão desaparece. Isso acontece porque ela é muito desorganizada, deixa tudo espalhado pela casa, tornando humanamente impossível localizar algum objeto no meio de tanta confusão. Ela sempre contou com a ajuda infalível de seu cãozinho Poodle, que consegue localizar seus objetos perdidos. Uma vez ela queria me mostrar a eficiência do seu Poodle. Escondeu propositalmente uma bola em um dos quartos, e gritou: “Poooooooodle!”. Então ela disse “Bola” e ele partiu para buscá-la. Ela ficou preocupada porque depois de 30 segundos ele ainda não tinha retornado com a bola. A surpresa foi que logo depois ele apareceu, triufante, carregando 4 bolas!!! A que Maria acabara de esconder e outras 3 de seus filhos, que ela nem se lembrava que existiam, e muito menos onde estavam!

Atualmente Maria faz pós-graduação em Computação. Seu projeto final é uma ferramenta de busca. Em homenagem a seu cãozinho que sempre buscou suas coisas, ela batizou seu projeto de Poodle. A ideia é simples: dada uma palavra, Poodle faz uma busca no disco e retorna todos os documentos que contém a dada palavra. Como no caso real da bola que comentei acima, na maioria das vezes a busca retorna bem mais resultados que o esperado. Os resultados são então exibidos agrupados em páginas. Por exemplo, se a ferramenta for configurada para exibir 10 resultados por página, e a busca retornar 143 resultados, eles serão exibidos em 15 páginas: 14 delas com 10 em cada uma, e a última com os 3 restantes.

A ferramenta já está pronta, e funciona muito bem. Mas Maria teve a feliz ideia de enfeitar o trabalho na tentativa de ganhar mais pontos... ao exibir o resultado de uma busca, Poodle mostra um logotipo com o nome da ferramenta, sendo que os o's de Poodle podem ser dois ou mais, dependendo da quantidade de páginas de resultado. A ideia é que “Poodle” seja escrito com tantas letras quantas forem as páginas de resultado, repetindo o's quando necessário. No exemplo acima, o logotipo seria “Pooooooooooodle”, que contém 15 letras.

Naturalmente, se a quantidade de páginas de resultados for inferior a 6, a palavra Poodle não será cortada, o logotipo será Poodle. E, para evitar que o logotipo fique tão grande que nem caiba na tela de resultados, ele será limitado a um máximo de 20 letras, mesmo que a quantidade de páginas de resultado seja superior a 20.

Sua tarefa é ajudar Maria a montar o logotipo.

Entrada

Há vários casos de teste.

Cada caso de teste é uma linha contendo dois números inteiros, N e P, sendo N o número de documentos encontrados pelo Poodle, e P o número de resultados exibidos por página ($1 \leq N \leq 1.000.000$, $1 \leq P \leq 100$). A entrada termina quando $N = P = 0$.

Saída

Para cada caso de teste da entrada seu programa deve produzir uma linha na saída contendo a palavra Poodle, ajustando a quantidade de o's de acordo com as regras descritas no enunciado.

Exemplos

Entrada:

```
20 4
143 10
42 5
80 3
0 0
```

Saída:

```
Poodle
Pooooooooooodle
Poooooodle
Poooooooooooooooodle
```

Maratona Mineira 2012

C. Arranging Heaps

Time limit: 2s Memory limit: 128 MB

A mining company extracts terbium, a rare metal used for constructing lightweight magnets, from riv sand. They mine the Long River at N mining points, each of them identified by its distance from t river source. At each mining point, a relatively small but highly valued heap of mineral ore is extract from the river.

To collect the mineral ore, the company regroups the N produced heaps into a smaller number K heaps, each located at one of the initial mining points. The newly formed heaps are then collect by trucks.

To regroup the N heaps, they use a barge, which in practice can carry any amount of mineral o because it is very large. The barge starts at the river source and can only travel downriver, so the he produced at a mining point X can be taken to a mining point Y only if $Y > X$. Each heap is mov completely to another mining point, or not moved at all. The cost of moving a heap of weight W fr a mining point X to a mining point Y is $W \times (Y - X)$. The total cost of the regrouping is the su of the costs for each heap movement. Notice that a heap which is not moved has no influence on t total cost.

Given the values for N and K, the N mining points, and the weight of the heap each mining poi produced, write a program that calculates the minimum total cost to regroup the N initial heaps in K heaps.

Input

Each test case is described using several lines. The first line contains two integers N and K denoti respectively the number of initial heaps and the desired number of heaps after regrouping ($1 \leq K$ $N \leq 1000$). Each of the next N lines describes one of the initial heaps with two integers X and indicating that the mining point X produced a heap of weight W ($1 \leq X, W \leq 10^6$). Within each te case the heaps are given in strictly ascending order considering their mining points.

Sample Output

For each test case output a line with an integer representing the minimum total cost to regroup the initial heaps into K heaps.

Sample Input

```
3 1
20 1
30 1
40 1
3 1
11 3
12 2
13 1
6 2
10 15
12 17
16 18
18 13
30 10
32 1
6 3
```

10 15
12 17
16 18
18 13
30 10
32 1

Sample Output

30
8
278
86

D. Stone Game

Time limit: 3s

Jack and Jim are playing an interesting stone game. At the beginning of the game there are N pile(of stones. Each pile has P_i $(i = 1..N, 1 \leq P_i \leq 2 * 10^9)$ stones. They take turns to take away sor of the stones. There are some rules: they must choose one pile at a time. They can take away a number of stones except 0, of course, not bigger than the number of stones in the pile. One who tak away the last stone will win the game. Jack is the first to fetch the match, and Jim is the second. N Jack asks you for help, to decide while facing some initializations whether he is sure to win or not.

Input

The input file contains several scenarios. Each of them consists of 2 lines:

The first line consists of a number N. The second line consists of N numbers, meaning P_i $(i = 1..N$ There is only one space between two border numbers.

The input file is ended with $N = 0$.

Output

For each scenario, print a line containing '`Yes`' if he is sure to win, or '`No`' otherwise.

Sample Input

```
1
100
3
1 5 1
4
1 1 1 1
0
```

Sample Output

```
Yes
Yes
No
```

E. Rational Approximation

Time limit: 3s

A polynomial $p(x)$ of degree n can be used to approximate a function $f(x)$ by setting the coeficients $p(x)$ to match the first n coeficients of the power series of $f(x)$ (expanded about $x = 0$). For examp

$$\frac{1}{(1-x)} \approx 1 + x + x^2 + \ldots + x^n$$

Unfortunately, polynomials are "nice" and they do not work well when they are used to approx mate functions that behave poorly (e.g. those with singularities). To overcome this problem, we c instead approximate functions by rational functions of the form $p(x)/q(x)$, where $p(x)$ and $q(x)$ a polynomials. You have been asked by Approximate Calculation Machinery to solve this problem, they can incorporate your solution into their approximate calculation software.

Given m, n, and the first $m+n$ coeficients of the power series of $f(x)$, we wish to compute t polynomials $p(x)$ and $q(x)$ of degrees at most $m-1$ and $n-1$, respectively, such that the power seri expansion of $q(x) \cdot f(x) - p(x)$ has 0 as its first $m+n-1$ coefficients, and 1 as its coefficient correspondi to the x^{m+n-1} term. In other words, we want to find $p(x)$ and $q(x)$ such that

$$q(x) \cdot f(x) - p(x) = x^{m+n-1} + \ldots$$

where $\ldots$ contains terms with powers of x higher than $m+n-1$. From this, $f(x)$ can be approximat by $p(x)/q(x)$.

Background Definitions

A polynomial $p(x)$ of degree n can be written as $p_0 + p_1 x + p_2 x^2 + \ldots + p_n x^n$, where p_i's are intege in this problem.

A power series expansion of $f(x)$ about 0 can be written as $f_0 + f_1 x + f_2 x^2 + \ldots$, where f_i's a integers in this problem.

Input

The input will consist of multiple cases. Each case will be specified on one line, in the form $m\ n\ f_0\ f_1 \ldots$ where f_i is the coefficient of x^i in the power series expansion of f. You may assume that $1 \leq$ $1 \leq n \leq 4$, $2 \leq m+n \leq 10$, and f_i are integers such that $|f_i| \leq 5$. The end of input will be indicated a line containing $m = n = 0$, and no coefficients for f. You may assume that there is a unique soluti for the given input.

Output

For each test case, print two lines of output. Print the polynomial $p(x)$ on the first line, and then $q($ on the second line. The polynomial $p(x)$ should be printed as a list of pairs (p_i, i) arranged in ascendi order in i, such that p_i is a non-zero coefficient for the term x^i. Each non-zero coefficient p_i should printed as 'a/b', where $b > 0$ and a/b is the coefficient expressed in lowest terms. In addition, if $b =$ then print only a (and omit b). If $p(x) = 0$, print a line containing only '`(0,0)`'. Separate the pairs the list by one space. The polynomial $q(x)$ should be printed in the same manner. Insert a blank li between cases.

Sample Input

```
2 2 0 0 1 1
4 2 1 2 3 4 5 -2
1 1 2 3
```

```
1 4 -5 0 -2 1-2
0 0
```

Sample Output

```
(0,0)
(1,1)

(-4/33,0) (-1/11,1) (-2/33,2) (-1/33,3)
(-4/33,0) (5/33,1)

(2/3,0)
(1/3,0)

(25/6,0)
(-5/6,0) (1/3,2) (-1/6,3)
```

F. Assalto ao banco central

Time limit: 1.423s Memory limit: 1536 MB

Armando, um funcionário do Banco Central da Sildávia, descobriu que há nos cofres do banco cerca de U$ 164,5 milhões. Como Armando não é a pessoa mais honesta do mundo, ele decidiu que irá tentar roubar esse dinheiro.

As paredes do cofre, porém, são revestidas por uma grossa camada de titânio, de forma que é inviável entrar nele exceto pela porta. Essa porta usa uma trava eletrônica extremamente sofisticada. Existem 32 chaves eletrônicas, e a porta só se abre se pelo menos **M** delas estiverem presentes. Cada um dos diretores do banco possui um microchip contendo zero ou mais dessas chaves implantado em algum lugar de seu corpo.

Assim, Armando concluiu, a única maneira de realizar o roubo é sequestrar alguns dos diretores do banco e trazê-los até a porta do cofre. Porém, sequestrar vários diretores ao mesmo tempo sem chamar a atenção da polícia é uma tarefa difícil. Assim, Armando quer minimizar o número de diretores a serem sequestrados. Ele, porém, está tendo dificuldade em determinar exatamente qual é o número mínimo e repassou essa tarefa a você.

Ele criou uma lista de diretores que ele crê que consegue sequestrar, e das chaves carregadas por cada um desses diretores. Dado o número mínimo de chaves **M** necessário para abrir a porta do cofre e uma lista das chaves que cada diretor possui, determine o número mínimo de pessoas que Armando deve sequestrar de forma a conseguir abrir o cofre.

Observações

- As chaves são identificadas por inteiros entre 0 e 31, inclusive;
- Quaisquer **M** chaves distintas abrem o cofre.

Entrada

A entrada começa com uma linha contendo um inteiro **T**, o número de casos de teste ($1 \leq \mathbf{T} \leq 20$). Em seguida, há **T** casos de teste.

Cada caso de teste começa com uma linha contendo dois inteiros **M** ($0 \leq \mathbf{M} \leq 32$) e **D** ($1 \leq \mathbf{D} \leq 20$), respectivamente o número mínimo de chaves necessário para abrir o cofre e o número de diretores que podem ser sequestrados. Em seguida, há **D** linhas, uma para cada diretor.

Cada uma dessas linhas começa com um inteiro **C** ($0 \leq \mathbf{C} \leq 32$), o número de chaves distintas presentes no microchip daquele diretor. Em seguida, há **C** inteiros identificando as chaves. Cada chave é identificada por um inteiro entre 0 e 31, inclusive.

Saída

Para cada caso de teste, imprima uma linha na saída contendo um inteiro que representa o número mínimo de diretores a serem sequestrados para que o assalto seja realizado com sucesso. Se for impossível conseguir abrir o cofre usando apenas os diretores da lista, imprima "`Desastre!`".

Exemplos

Entrada:

```
3
5 2
6 1 13 23 24 26 29
0
5 3
2 10 15
3 0 10 15
3 0 2 10
4 4
3 9 19 20
2 19 20
2 9 19
3 17 19 20
```

Saída:

```
1
Desastre!
2
```

No primeiro caso de teste, o primeiro diretor possui mais chaves do que o necessário, então basta sequestrá-lo.

No segundo caso, os três diretores juntos possuem apenas 4 chaves distintas (0, 2, 10, 15), de forma que não é possível obter as 5 chaves necessárias para se abrir o cofre.

No terceiro caso, nenhum diretor isoladamente possui 4 chaves. Porém, se sequestrarmos o quarto diretor (com as chaves 17, 19, e 20), podemos sequestrar também ou o primeiro ou o terceiro diretor (que possuem a chave 9) e abrir o cofre. Assim, 2 diretores são o suficiente.

Seletiva UFMG 2011

G. Jogo da velha

Time limit: 0.648s Memory limit: 1536 MB

O jogo da velha é um dos jogos mais antigos da humanidade; os primeiros registros dele são do século I antes de Cristo, no Império Romano. João e Maria jogam bastante jogo da velha, mas depois de algum tempo eles decidiram jogar uma variante do jogo da velha tradicional, o jogo da velha 1-D.

O jogo da velha 1-D é um jogo disputado por dois jogadores em um tabuleiro 1 × **N**; inicialmente, todas as casas do tabuleiro estão vazias. Os jogadores alternam-se desenhando uma cruz sobre uma casa vazia. O primeiro jogador a completar uma sequência de três ou mais cruzes em casas consecutivas ganha o jogo.

Maria logo percebeu que, dependendo da situação do jogo, sendo sua vez de jogar, ela pode sempre garantir a vitória, independente das jogadas de João. Isto é relativamente fácil para tabuleiros menores, mas para tabuleiros maiores, mesmo após várias jogadas, esta tarefa é mais difícil; por isso, ela pediu que você escrevesse um programa que, dada a situação do tabuleiro, decide se ela tem uma estratégia vencedora.

Entrada

A entrada contém vários casos de teste. A primeira linha de caso de teste contém um inteiro **N**, indicando o tamanho do tabuleiro ($3 \leq \mathbf{N} \leq 10^4$). A linha seguinte contém uma sequência de **N** caracteres indicando quais casas do tabuleiro já foram ocupadas: um '.' indica que a casa correspondente está vazia, enquanto um 'X' indica que a casa já teve uma cruz desenhada sobre ela. A entrada nunca contém três 'X' consecutivos.

O ultimo caso de teste é seguido por uma linha que contém um único número zero.

Saída

Para cada caso de teste da entrada seu programa deve imprimir uma única linha contendo um único caractere: 'S' caso Maria possua uma estratégia vencedora e 'N' caso contrário.

Exemplo

Entrada:

```
5
.....
5
..X..
6
X.X.X.
12
............
0
```

Saída:

```
S
N
S
N
```

Primeira fase da Maratona de Programação - 2010

H. Macaco rural

Time limit: 0.837s Memory limit: 1536 MB

Você foi contratado como programador para um novo website de compras coletivas chamado *Macaco Rural*. Para se diferenciar dos seus vários concorrentes, esse site planeja oferecer ofertas para *pares* de produtos. Por exemplo, “compre uma bola de futebol e uma camisa oficial da seleção brasileira com 75% de desconto”.

O site quer planejar as ofertas do dia para os próximos n dias. Para tanto, há uma lista de 2n produtos, com seus respectivos preços (já com os descontos aplicados). Como ofertas muito caras vendem menos, seu chefe quer minimizar o preço da oferta mais cara. Cabe a você agrupar os 2n produtos em n pares de forma tal que o custo do par mais caro seja minimizado. O custo de um par é a soma dos custos dos produtos que o compõem.

Entrada

Há vários casos de teste.

Cada caso de teste começa com uma linha que contém um único inteiro N, o número de produtos que serão usados para criar as ofertas ($1 \leq N \leq 2.000.000$, e N é sempre um número par). Em seguida, há uma linha contendo N inteiros P_1, P_2, ..., P_N, que representam os preços dos N produtos que serão pareados em N/2 ofertas ($0 \leq P_i \leq 1.000.000.000$, para todo i).

A entrada termina com N = 0, que não deve ser processado.

Saída

Para cada caso de teste, imprima uma linha contendo um único inteiro, que é o maior preço de uma oferta, quando os N produtos são pareados de forma tal a minimizar esse maior preço.

Exemplos

```
Entrada:
4
1 19 26 17
8
3 9 6 18 14 1 7 8
0

Saída:
36
19
```

No primeiro caso de teste, há 3 possibilidades:

- (1, 19), (17, 26), com custos 1+19=20 e 17+26=43. O maior custo é 43.
- (1, 26), (17, 19), com custos 1+26=27 e 17+19=36. O maior custo é 36.
- (1, 17), (19, 26), com custos 1+17=18 e 19+26=45. O maior custo é 45.

Dessas três opções, a que minimiza o maior custo é a segunda, que leva a um custo máximo de 36.

I. Luzes de palco

Time limit: 0.109s Memory limit: 1536 MB

A cerimônia de premiação da OBI deste ano será espetacular. A atração principal será um show exclusivo dos Rolling Stones. Em preparação para o show, os técnicos de som e luzes da famosa banda estão visitando o Brasil. Um dos técnicos, ao saber que o evento tem tantos programadores altamente treinados, solicitou a ajuda para resolver um problema da banda.

Para controlar as luzes do palco durante o show, a banda tem um sofisticado equipamento com um painel coberto de botões. Cada botão controla duas luzes distintas: cada vez que o botão é pressionado o estado de ambas as luzes é invertido. Ou seja, se uma luz está apagada, após o botão ser pressionado ela se acende. Da mesma forma, se uma luz está acesa, após o botão ser pressionado ela se apaga. Dois botões distintos não controlam o mesmo par de lâmpadas.

O que o técnico quer saber é, dada uma certa configuração inicial do estado do conjunto de luzes, se existe uma sequência de acionamento dos botões para levar o estado das luzes para uma outra dada configuração.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado). A primeira linha do conjunto de testes contém dois números inteiros L e B que indicam respectivamente o número de luzes ($2 \leq L \leq 10^4$) e o número de botões ($1 \leq B \leq 10^6$). As luzes são identificadas por números de 1 a L.

A segunda linha da entrada contém L inteiros I_i representando a configuração inicial das luzes ($0 \leq I_i \leq 1$ para $1 \leq i \leq L$). I_i representa o estado da luz de número i: o valor ‘0’ indica que a luz está apagada e o valor ‘1’ indica que a luz está acesa. A terceira linha da entrada contém L inteiros F_i representando a configuração final das luzes ($0 \leq F_i \leq 1$ para $1 \leq i \leq L$), de maneira similar à configuração inicial.

Cada uma das B linhas seguintes contém um par de inteiros X e Y que representam as luzes controladas por um dos botões ($1 \leq X, Y \leq L$).

Saída

Seu programa deve imprimir, na saída padrão, uma linha contendo a letra ‘S’ caso exista uma sequência de acionamento de botões que coloca as luzes na configuração final. Caso contrário, seu programa deve imprimir a letra ‘N’.

Exemplo

Entrada:

```
4 3
0 0 0 0
1 1 1 1
1 2
1 3
1 4
```

Saída:

```
S
```

Entrada:

```
4 2
1 0 1 0
1 1 0 0
1 2
3 4
```

Saída:

```
N
```

Seletiva IOI 2008

J. Mixtures

Time limit: 3s Memory limit: 1536 MB

Harry Potter has n mixtures in front of him, arranged in a row. Each mixture has one of 100 different colors (colors have numbers from 0 to 99).

He wants to mix all these mixtures together. At each step, he is going to take two mixtures that stand next to each other and mix them together, and put the resulting mixture in their place.

When mixing two mixtures of colors a and b, the resulting mixture will have the color (a+b) mod 100.

Also, there will be some smoke in the process. The amount of smoke generated when mixing two mixtures of colors a and b is a*b.

Find out what is the minimum amount of smoke that Harry can get when mixing all the mixtures together.

Input

There will be a number of test cases in the input.

The first line of each test case will contain n, the number of mixtures, 1 <= n <= 100.

The second line will contain n integers between 0 and 99 - the initial colors of the mixtures.

Output

For each test case, output the minimum amount of smoke.

Example

```
Input:
2
18 19
3
40 60 20

Output:
342
2400
```

In the second test case, there are two possibilities:

- first mix 40 and 60 (smoke: 2400), getting 0, then mix 0 and 20 (smoke: 0); total amount of smoke is 2400
- first mix 60 and 20 (smoke: 1200), getting 80, then mix 40 and 80 (smoke: 3200); total amount of smoke is 4400

The first scenario is a much better way to proceed.

K. Cinefinlândia

Time limit: 0.145s Memory limit: 1536 MB

Na pacata cidade interiorana de Cinefilândia, o assunto predileto dos moradores é, obviamente, filmes. Para facilitar a conversação, os moradores referem-se aos filmes somente por suas siglas. O filme *The Lord of the Rings*, por exemplo, é tratado simplesmente por LotR.

Como Cinefilândia está entrando para rotas turísticas por causa de suas famosas poças de lama anti-reumáticas, a prefeitura está montando um guia para turistas e contratou você para gerar as siglas dos filmes de forma computadorizada. A regra é simples: o primeiro caractere de cada palavra do nome do filme é usado na sigla na forma maiúscula, exceto para palavras da lista especial. Para estas palavras, se forem a primeira do nome, a palavra é siplesmente ignorada; e, caso não sejam a primeira, tem seu caractere correspondente na forma minúscula.

Entrada

A entrada dos dados inicia-se com uma linha com um inteiro N correspondente ao número de casos de teste. Para cada caso de testes, segue uma linha com os inteiros E e F ($1 \leq E,F \leq 100$), seguida por uma linha com E palavras especiais, com no máximo 10 caracteres cada, e por F linhas com os nomes dos filmes, com no máximo 100 caracteres cada. Palavras, especiais ou não, são formadas por caracteres minúsculos e maiúsculos e dígitos, e podem ser separadas por hífens (-) e (:), além de espaços.

Na entrada, caracteres maiúsculos ou minúsculos são considerados iguais. Isto é, as palavras “harry” e “hArrY” são consideradas iguais.

Saída

Para cada nome de filme, seu programa deve gerar a sigla correspondente, com as primeiras letras de cada palavra. Palavras especiais tem sua letra correspondente na sigla na forma minúscula e as demais palavras tem sua letra correspondente na sigla na forma maiúscula. Além disso, se a primeira palavra do nome for uma palavra especial, então ela deve ser ignorada. Atenção: se as duas primeiras palavras forem especiais, então só a primeira será ignorada.

Todos os casos de teste geram alguma sigla.

Após cada caso de teste (inclusive o último), imprima uma linha em branco.

Exemplos

Entrada:

```
2
11 3
The and an E ou of De O da dos Das
The Lord of The Rings
O Senhor Dos Aneis
E o Vento Levou
2 3
no in
Xisto no Espaco 3
Rambo 2: A missao
O imperio contra-ataca
```

Saída:

```
LotR
SdA
oVL

XnE3
R2AM
OICA
```

Maratona Mineira 2012

L. Competição de chocolate

Time limit: 0.188s Memory limit: 1536 MB

Carlos e Paula acabaram de ganhar um saco com bolinhas de chocolate. Como sabem que vão comer tudo muito rápido inventaram uma brincadeira:

- Eles vão comer de forma alternada, um depois o outro, sendo que sempre a Paula começa.
- Quem comer a última bolinha ganha a bricadeira.
- A cada vez, só se pode comer de 1 a M bolinhas, sendo o M decidido pela mãe de Paula, de forma que não engasguem com o chocolate.

Um exemplo de partida para M = 5, onde Paula ganhou:

Quem joga	Quantas comeu	Número de bolinhas restantes
	-	17
Paula	5	12
Carlos	4	8
Paula	2	6
Carlos	5	1
Paula	1	0

Ambos são muito espertos e jogam de maneira ótima, de forma que se existe para um deles uma sequência de jogadas que garante a vitória independente da jogada do outro, essa pessoa jogará dessa forma.

Tarefa

Sua tarefa é determinar quem vai ganhar a brincadeira, se ambos jogam de forma ótima.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (normalmente o teclado).

A entrada consiste de uma linha contendo dois inteiros N ($1 \leq N \leq 10^6$) e M ($1 \leq M \leq 10^3$), sendo N o número de bolinhas de chocolate e M o número de bolinhas permitidas por vez.

Saída

Seu programa deve imprimir, na saída padrão, uma linha, contendo o nome do vencedor, como exemplificado abaixo.

Exemplos

Entrada

```
5 3
```

Saída

```
Paula
```

Entrada

```
30 5
```

Saída

```
Carlos
```

OBI 2009 - fase 2 nível júnior