Problema A. AAAAAAdenilação

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

A poliadenilação é a adição de uma cauda poli(A) ao RNA, geralmente o RNA mensageiro (mRNA). A cauda poli(A) é composta de uma sequência de bases adeninas e fica na extremidade que é transcrita por último, a extremidade 3'. Ela é importante para a estabilidade do mRNA pois permite que o RNA mensageiro sobreviva mais tempo sem ser digerido acidentalmente pela própria célula.

O aperfeiçoamento no entendimento desses e outros processos biológicos nos permitiu hoje a criação de vacinas que utilizam impressoras de DNA que criam fitas de DNA que são transcritas para fitas mRNA que ultimamente são encapsuladas em um frasco. Assim, uma das otimizações que se foi buscada é o tamanho ideal da cauda poli(A) para permitir que a vacina fosse mais eficaz.

Ariel teve interesse em descobrir qual o tamanho do cauda poli(A) de alguns códigos RNA, e acabou descobrindo que é possível colocar códigos ligadores no meio do código do poli(A), e que podem existir outras caudas, como o poli(C) após a cauda de poli(A). Então para contabilizar o tamanho da cauda corretamente, Ariel decidiu que a cauda poli(A) é a maior subsequência que começa com 7 `'A'`s, e termina com 7 `'A'`s, independente de onde esteja.

No código da vacina para SARS-CoV-2 da BioNTech/Pfizer por exemplo, é possível encontrar um ligador de tamanho 10 AGCAUAUGACU dentro do poli(A), que deixa a junção dos pedaços de DNA sintético impressos seja mais fácil e também faz com que o mRNA resultante tenha um comprimento mais uniforme.

Ajude Ariel então a encontrar o tamanho da cauda poli(A) dado o código RNA.

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^6$), o tamanho do código de RNA.

Segue uma linha com N caracteres `'G'`, `'A'`, `'U'` ou `'C'`, representando o código RNA.

Saída

Imprima o tamanho da cauda poli(A), ou 0 se não há cauda poli(A).

Exemplos

entrada padrão
75 GAGAUUGCUAGCAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGCAUAUGACUAAAAAAAAAAAAAAAAAAAAAA
saída padrão
63

entrada padrão
75 GAGCAGAUUAAUAAAAAAAAAAAAAAAAAAAAAAAAAAUGCAUCCCCCCCCCCCCCCCCCCCAAAGGCUGCACCAGAAUU
saída padrão
23

entrada padrão
60 AUGGAUUCUAACACUGUGUCAAGUUUUCAGGUAGAUUGCUUCCUUUGGCAUGUCCGAAAA
saída padrão
0

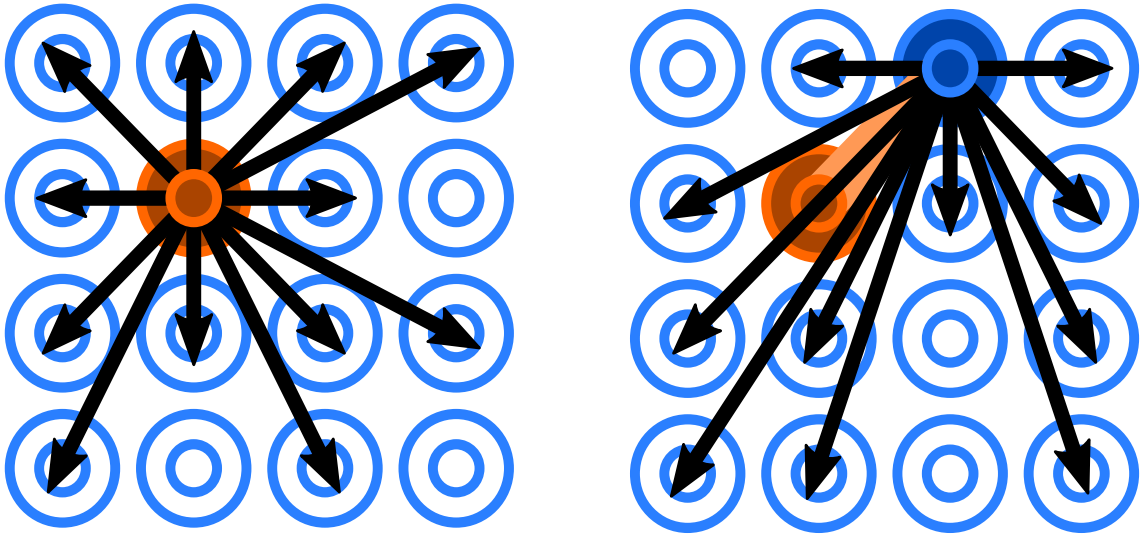
Problema B. Bloqueio de Tela

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	10 segundos
Limite de memória:	256 megabytes

Petra decidiu instalar uma ROM customizada no seu celular. Fã de bloqueios de tela de deslizar, Petra ficou surpreso com as novas funcionalidades que a ROM trouxe para o seu aparelho, permitindo que matrizes de até 5x4 pontos sejam criadas! Petra então se perguntou o quão seguro seria utilizar um bloqueio de tela desse tamanho ao invés de um PIN tradicional, e decidiu saber mais sobre esse sistema incrementado de bloqueio de tela.

Depois de alguma experimentação, Petra descobriu que além da extensão de tamanho, cada ponto de uma padrão pode ser ou segurado (laranja) ou deslizado (azul), resultado de um ponto da matriz ficar pressionado por um intervalo maior de tempo. Isso significa que por exemplo um padrão que é 100% composto de pontos deslizados é diferente de uma padrão onde o primeiro ponto é segurado, e os outros deslizados.

Já a extensão da matriz de pontos é como no Android original, isto é, para que um movimento seja válido é necessário que o segmento de reta formado pelo ponto atual e o que se quer conectar não passe por um ponto que não faz parte do padrão. Um movimento em L por exemplo é aceito, assim como movimentos mais complicados.



Petra então quer saber quantas combinações diferentes são possíveis nessa ROM customizada para alguns tamanhos diferentes de padrão. Para fazer comparações, Petra determinou que apenas alguns pontos do padrão podem ter duas cores, e outros não, e como a quantidade de combinações pode ser muito grande, Petra também pediu para você dar as combinações mod 998244353.

Entrada

A primeira linha contém o tamanho do padrão dado por dois inteiros N e M ($1 \leq N, M \leq 5$) com $\min(N, M) \leq 4$.

Em seguida, seguem N linhas, com M inteiros P ($1 \leq P \leq 2$), sendo que P indica a quantidade de cores diferentes possíveis para aquele ponto. Um ponto $P = 1$ não registra diferença entre ser acionado segurado ou não segurado, e já um ponto $P = 2$ pode ser acionado segurado ou não segurado, formando padrões diferentes.

No final, seguem A e B ($1 \leq A \leq B \leq NM$), o tamanho mínimo e máximo do tamanho dos padrões.

Saída

Imprima o número de combinações diferentes possíveis mod 998244353.

Exemplos

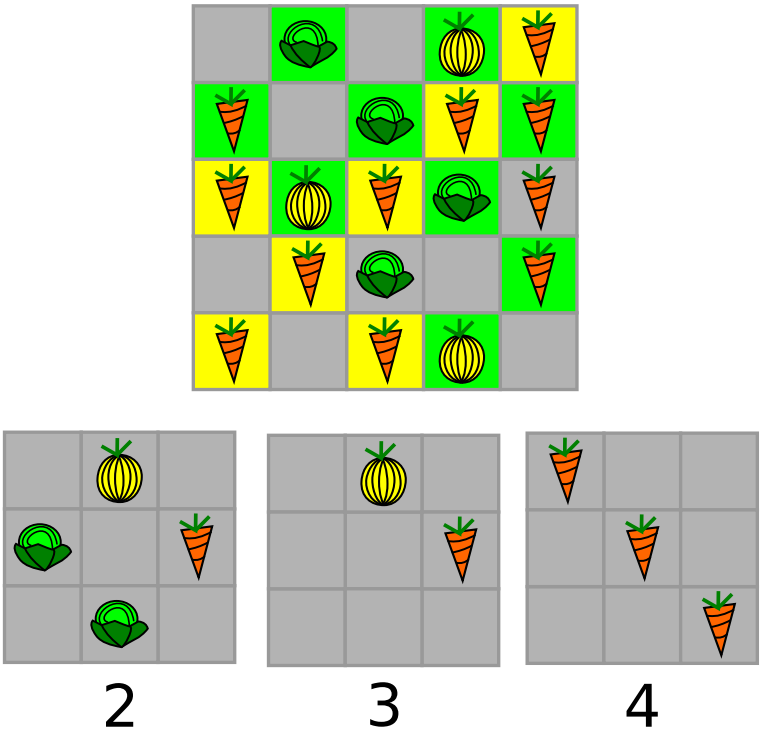
entrada padrão	saída padrão
2 2 1 1 1 1 2 2	12
2 2 2 1 1 1 2 2	18
3 3 1 1 1 1 1 1 1 1 1 4 9	389112
4 5 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 20	9192018658

Problema C. Choque de Cultura

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	2 segundos
Limite de memória:	256 megabytes

Choque de Cultura é um jogo muito popular na Nlogônia. Nele, dois jogadores se alternam colocando plantações de cebola, cenoura e couve em um tabuleiro bidimensional. Os jogadores possuem em suas mãos cartas com um tabuleiro 3x3 com um certo padrão desses três itens, e a qualquer momento, qualquer um deles pode iniciar um *choque de cultura*.

Quando isso ocorre, ambos os jogadores precisam pegar suas cartas e contabilizar a quantidade de combinações que são possíveis com as cartas que estão na sua mão com o que está no tabuleiro. Os padrões podem se sobrepor, sendo esta uma boa estratégia, porque assim que um padrão é usado ao menos uma vez, ele some da mão do jogador e vai para a pilha de descarte. As cartas também não tem direção específica, qualquer uma das 4 rotações da carta (0°, 90°, 180°, 270°) são válidas para se fazer uma combinação. Porém, lembre-se que espelhamento vertical ou espelhamento horizontal não são equivalentes a operações de rotação.



Petya está jogando com Petra e precisa de sua ajuda para contar a quantidade de combinações que podem ser encontrados no tabuleiro dadas as cartas em sua mão.

Entrada

A primeira linha contém dois inteiros N e M ($1 \leq N, M \leq 1000$), o tamanho do tabuleiro.

Segue então N linhas com M caracteres cada indicando o que está plantado em cada posição. `'0'` significa que não tem nada plantado, `'1'` que uma cebola está plantada, `'2'` que uma cenoura está plantada em `'3'` que uma couve está plantada.

A próxima linha contém um inteiro Q ($1 \leq Q \leq 1000$), o número de cartas na mão de Petya.

Seguem Q conjuntos de 3 linhas, sendo que cada linha contém 3 caracteres, no mesmo formato do tabuleiro, sendo este o padrão de uma carta na mão de Petya. É garantido que as cartas tem ao menos uma posição não-zero.

Saída

Para cada carta, imprima a quantidade de combinações encontradas no tabuleiro na mesma ordem da entrada.

Exemplos

entrada padrão	saída padrão
5 5 03012 20322 21232 02302 20210 3 010 302 030 010 002 000 200 020 002	2 3 4
4 4 0010 0111 1000 0001 3 001 000 000 000 010 000 100 010 000	6 6 3

Problema D. Dados Vazados

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

Uma nova vulnerabilidade encontrada no sistema de diversas empresas da Nlogônia expôs os dados de muitos usuários. Com a nova Lei Grande Disseminação é Punição (LGDP), o governo agora precisa investigar e punir as empresas adequadamente. Além do crime de permitir o vazamento dos dados, é certo que algumas dessas empresas venderam os dados de seus usuários para outras empresas sem notificá-los, e podemos descobrir exatamente quem elas são olhando para os dados vazados.

Para fazer essa investigação, foi primeiro achado alguns usuários cujos cadastro foi restrito a apenas uma empresa. Se o cadastro desse usuário vazou para outra empresa, é certo que houve venda de dados indevida. Para anonimizá-los, foi criado um identificador sequencial para cada um, de 1 a M. Sabemos que uma empresa x vendeu todos os seus dados a outra empresa y se o conjunto de usuários da empresa x for subconjunto da empresa y, ou seja $U_x \subseteq U_y$.

Sua tarefa é identificar todas as empresas que venderam as suas informações para outras empresas, mas que não compraram informações de nenhuma outra empresa, isto é, empresas *fonte* de informações. É possível que existam empresas *fonte* que tem exatamente a mesma informação, nesse caso, a empresa *fonte* é aquela que aparece primeiro dentre esse conjunto.

Entrada

A primeira linha contém os inteiros N ($1 \leq N \leq 500$), a quantidade de empresas investigadas e M ($1 \leq M \leq 500$), a quantidade de usuários cuja informação foi vendida.

Seguem N linhas, cada uma descrevendo a empresa i ($0 \leq i < N$) em ordem. Cada linha contém um inteiro $|U_i|$ ($1 \leq |U_i| \leq M$), a quantidade de usuários vazados pela empresa i, e $|U_i|$ inteiros U_{ij} ($1 \leq U_{ij} \leq M$) distintos representando um usuário com informação vazada.

Saída

Imprima o i de todas as empresas *fonte* em ordem ascendente, um em cada linha.

Exemplos

entrada padrão	saída padrão
6 5 2 2 1 3 3 1 2 1 1 1 4 2 4 5 1 4	2 3
4 4 2 1 2 2 2 3 2 2 4 2 3 4	0 1 2 3

Problema E. Esperantaj Verbkonjugacioj

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

O Esperanto é a língua planejada mais falada no mundo. Ele foi criado em 1887 pelo polonês Ludwik Lejzer Zamenhof, com a intenção de criar uma língua franca mundial. Existem hoje por volta de 200 a 2000 falantes nativos da língua, mais de 100 mil falantes fluentes e mais de 10 milhões de falantes não fluentes. A Internet propiciou uma nova onda de falantes, vindo de aplicativos como o Duolingo e o Lernu que permitem que qualquer pessoa aprenda Esperanto.

O alfabeto Esperanto é uma versão modificada do alfabeto latino, sendo ele composto das 28 letras:

a b c ĉ d e f g ĝ h ĥ i j ĵ k l m n o p r s ŝ t u ŭ v z

Para permitir que a comunicação pudesse ser feita apenas em ASCII, foi inventado alguns sistemas de transliteração, como o sistema-h. Nele, a letra ĉ vira ch, a letra ĝ vira gh, a letra ĵ vira jh, a letra ŝ vira sh, a letra ĥ vira hh e a letra ŭ tem seu acento removido. Com a adoção do Unicode, porém, esse sistema se torna cada vez mais desnecessário.

O vocabulário e a semântica são importados de línguas indo-europeias ocidentais, algo que é alvo de críticas pois deixa o aprendizado de falantes de línguas não-europeias mais difícil. Porém, isso permite que o aprendizado de linguagens como o Francês seja facilitado a falantes de Esperanto.

Como a língua foi inventada do zero, a fonologia, a morfologia e a gramática tem regras simples e são regulares. A pronúncia de cada uma das letras por exemplo é sempre a mesma, e a sílaba tônica nas palavras é sempre a penúltima. A conjugação de verbos usa pequeno conjunto de regras que se aplica a todos os verbos universalmente, como mostra a tabela a seguir.

Terminação	Modo	Exemplo	Tradução
-i	Infinitivo	lerni	aprender
-is	Pretérito do Indicativo	mi legis	eu li
-as	Presente do Indicativo	ili manĝas	eles comem
-os	Futuro do Indicativo	ni faros	nós faremos
-u	Imperativo	vi kantu!	cante!
-us	Condicional	ŝi laborus	ela trabalharia

Sam está aprendendo Esperanto e precisa de sua ajuda para descobrir qual conjugação foi utilizada em algumas palavras.

Entrada

A primeira linha contém um inteiro N ($4 \leq N \leq 50$).

A segunda linha contém uma sequência de N caracteres do alfabeto latino minúsculo, um verbo em Esperanto usando o sistema-h.

Saída

Imprima o modo do verbo por meio da impressão da terminação (i, is, as, os, u, us).

Exemplos

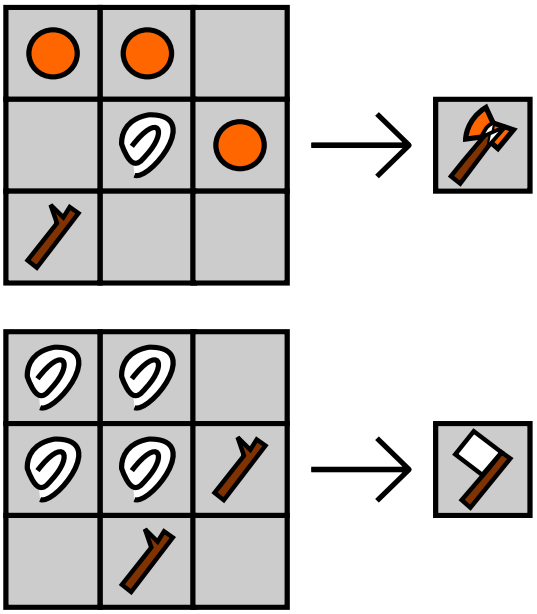
entrada padrão	saída padrão
8 komputus	us
20 kontraudiskriminaciu	u
7 manghas	as

Problema F. Fabricação Facilitada

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

Na Linerônia, o jogo *FabriMine* hoje é muito popular entre os jovens. Dentro desse jogo, existe um sistema de fabricação de itens que se utiliza de uma grade de espaços. Dentro do inventário do próprio jogador, a área para fabricação é de 2x2, e utilizando-se de uma Mesa de Fabricação, a área de fabricação se torna 3x3. O sistema de fabricação se baseia em receitas, sendo que cada receita diz qual item é criado e em qual quantidade dado os ingredientes em um certo padrão.

Um machado de cobre, por exemplo, é fabricado juntando um pedaço de madeira, um fio de corda e três pedaços de metal de cobre em uma disposição específica. Já uma bandeira, é feita usando dois pedaços de madeira, quatro fios de corda em outra disposição. O jogador pode fabricar quantos itens desse ele quiser, dado que ele possua os ingredientes em seu inventário.



Porém, esse sistema de fabricação foi considerado complexo demais para uma versão mais amigável do jogo para consoles, então foi decidido criar um livro de receitas para ajudar os jogadores a fabricarem itens sem saber das suas receitas. O livro mostra apenas o que é possível fazer dado os ingredientes disponíveis no inventário do jogador, e em qual quantidade.

Como desenvolvedor(a) nato e que acha que jogar no computador é superior, você decidiu modificar o seu jogo e adicionar o livro de receitas. Dado um inventário e as receitas disponíveis no jogo, mostre as receitas que são possíveis de fazer e em qual quantidade.

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^3$), a quantidade de receitas e um inteiro K ($1 \leq K \leq 10^4$), a quantidade de ingredientes diferentes.

Seguem N linhas, cada uma representando uma receita i ($1 \leq i \leq N$). Cada linha contém um inteiro M_i ($1 \leq M_i \leq 9$), a quantidade de ingredientes da receita, seguido de M_i inteiros G_{ij} ($1 \leq G_{ij} \leq K$ e $1 \leq j \leq M_i$), cada ingrediente daquela receita. Um mesmo ingrediente pode aparecer várias vezes na mesma receita.

Depois das receitas, na próxima linha há um inteiro V ($1 \leq V \leq 10^4$), o tamanho do inventário do jogador. Seguem V linhas, cada uma descrevendo uma posição do inventário do jogador, cada linha contém um

inteiro G ($1 \leq G \leq K$), um ingrediente de receita naquela posição, e um inteiro Q ($1 \leq Q \leq 100$), a quantidade de itens naquela posição. O inventário do jogador geralmente não é nada organizado.

Saída

Imprima uma linha para cada receita que é possível de ser fabricada usando os ingredientes no inventário do jogador na mesma ordem da entrada. Esta linha deve conter o inteiro i ($1 \leq i \leq N$), o identificador da receita e um inteiro Q, a quantidade de itens que podem ser fabricados.

Exemplo

entrada padrão	saída padrão
3 4 5 1 1 2 1 3 4 1 4 4 4 6 2 2 2 2 3 3 7 2 30 3 21 1 7 1 3 3 9 1 2 4 2	1 4 3 7

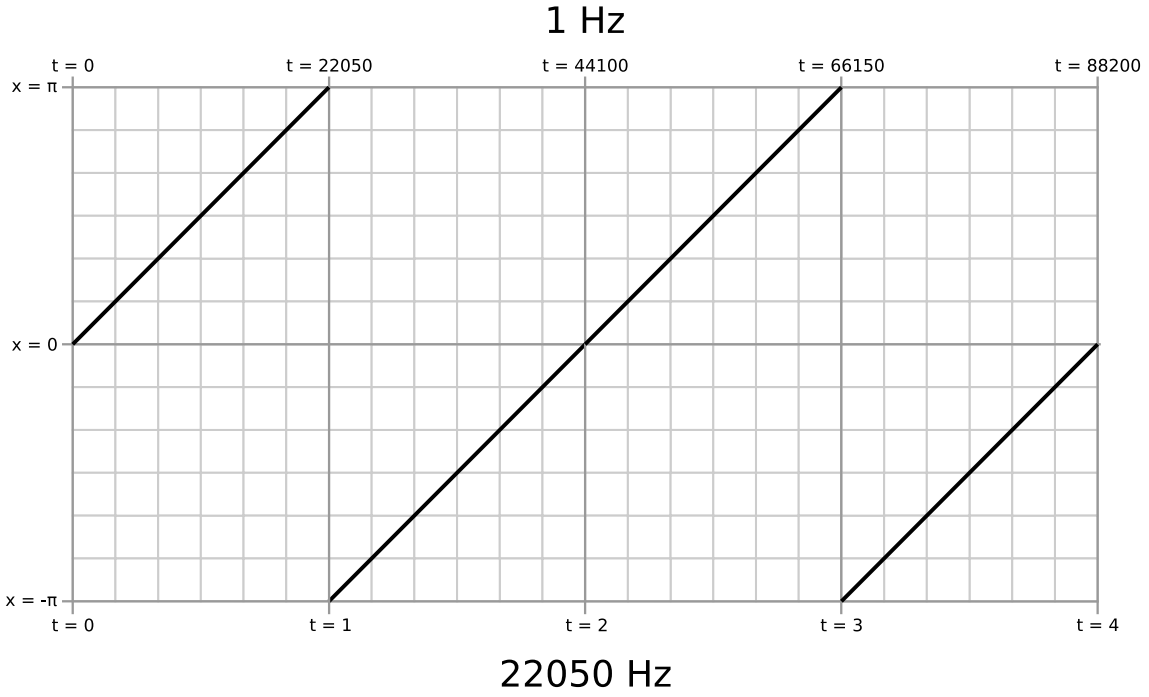
Problema G. Gerador de Ondas

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

A divisão de anúncios da Elogog pensou em um jeito para chamar os músicos para usar o seu navegador de Internet. A divisão propôs então uma forma de implementar um sintetizador dentro do seu navegador! O sintetizador deverá ser capaz de gerar ondas senoidais, quadradas, dente de serra e triangulares, definidas da seguinte maneira:

- **Seno**: $f(x) = \sin x$.
- **Quadrado**: $f(x) = \begin{cases} 1 & \text{para } 0 \leq x < \pi \\ -1 & \text{para } -\pi \leq x < 0. \end{cases}$
- **Dente de serra**: $f(x) = \frac{x}{\pi}$.
- **Triângulo**: $f(x) = \begin{cases} \frac{2}{\pi}x & \text{para } 0 \leq x < \frac{\pi}{2} \\ 1 - \frac{2}{\pi}(x - \frac{\pi}{2}) & \text{para } \frac{\pi}{2} \leq x < \pi \\ -f(-x) & \text{para } -\pi \leq x < 0. \end{cases}$

O sintetizador opera com os padrões de áudio atuais. A cada segundo, ele produz 44100 amostras, inteiros de 32 bits com sinal (entre -32768 e 32767). As amostras são resultado da aplicação de $f(x)$ onde $-\pi \leq x < \pi$, sendo que x varia de acordo com a frequência (basicamente, a nota musical) que se quer obter, seguindo uma função periódica.



Por exemplo para a frequência de 1 Hz, começamos com $x = 0$ para o instante de tempo $t = 0$. x vai crescendo até π até chegarmos no tempo $t = 22050$. Depois disso, x recomeça de $-\pi$, e chega a π de novo apenas em $t = 66150$. Note então que para 1 Hz, repetimos o intervalo de $t = 0$ até $t = 44100$. Para 22050 Hz, podemos perceber que esse intervalo repetido é reduzido, repetimos a cada 2, começando em $t = 0$ até $t = 2$. Porém, lembre que esse intervalo nem sempre é inteiro.

Sabendo destes detalhes, ajude então os engenheiros da Elogog a implementar o sintetizador!

Entrada

Na primeira linha são encontrados um caractere W que descreve o tipo da onda e um inteiro F ($1 \leq F \leq 10^6$), a frequência da onda a ser sintetizada. O caractere W pode ser `'S'`, a onda senoidal, `'Q'`, a onda quadrada, `'W'`, a onda dente de serra e `'T'`, a onda triangular.

Na segunda linha são encontrados dois inteiros A e B ($0 \leq A < B \leq 10^9$), que indicam o intervalo de amostras do sintetizador. É garantido que $(B - A) \leq 10^5$.

Saída

Imprima $(B - A)$ linhas, cada amostra do sintetizador para o instante de tempo t no intervalo $[A, B)$. A amostra deve estar no intervalo $[-32768, 32767]$ e erros de arredondamento são tolerados.

Exemplos

entrada padrão	saída padrão
Q 22050 0 3	32767 -32768 32767
W 11025 0 4	0 16384 -32768 -16384
S 3675 999999993 1000000000	-32768 -28377 -16384 0 16384 28377 32767
T 3675 12950 12954	21845 32767 21845 10922
S 440 0 14	0 2053 4098 6126 8131 10104 12036 13922 15753 17522 19222 20847 22390 23845
T 1 992241 992255	27 24 21 18 15 12 9 6 3 0 -3 -6 -9 -12

Problema H. Hashiwokakero Bicolor

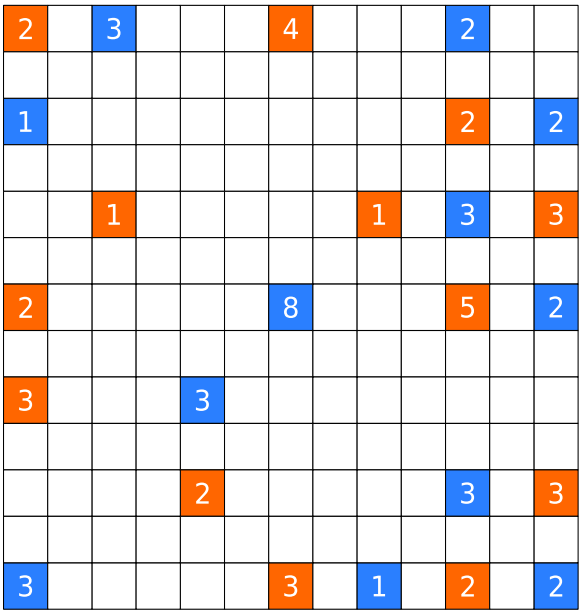
Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	2 segundos
Limite de memória:	256 megabytes

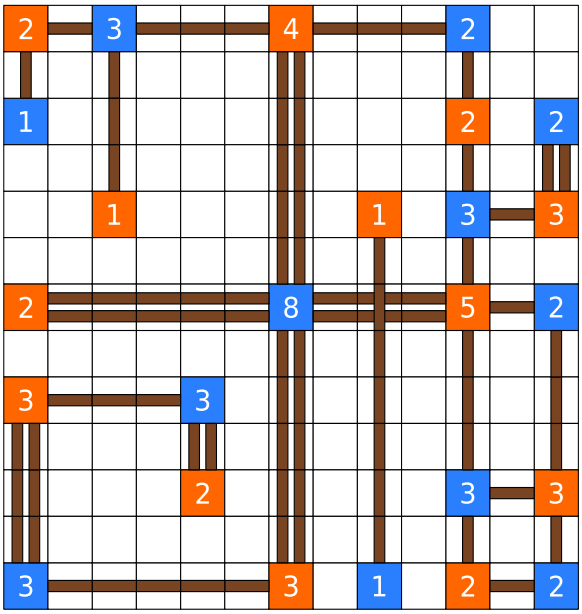
O *Hashiwokakero* é um quebra-cabeça lógico japonês que literalmente significa "construa pontes!". O jogo é composto de um tabuleiro retangular quadriculado de tamanho arbitrário. Alguns quadrados são vazios e outros tem um círculo com um número entre 1 e 8, sendo esses chamados de ilhas. O objetivo é conectar todas as ilhas construindo pontes entre elas, seguindo algumas regras.

Laos estava brincando com esses puzzle e teve uma ideia de criar uma variação que chamou de Hashiwokakero Bicolor, com as seguintes regras:

- As pontes devem começar e terminar em ilhas de cores diferentes, sendo as pontes linhas retas ortogonais;
- No máximo duas pontes conectam um par de ilhas;
- O número de pontes conectadas a cada ilha tem que bater com o número naquela ilha.

Laos desenhou um exemplo de um tabuleiro de Hashiwokakero Bicolor que tem solução, mas ele ficou pensando se era possível fazer um programa de computador que pudesse lhe dizer isso.





Ajude Laos! Crie um programa que dado um tabuleiro de Hashiwokakero Bicolor, imprime se existe solução ou não.

Entrada

A primeira linha contém dois inteiros: N ($1 \leq N \leq 40$) e M ($1 \leq M \leq 40$). As N linhas a seguir descrevem o tabuleiro.

Cada linha do tabuleiro é composta de M sequências de dois caracteres. As posições vazias são representadas por dois caracteres `'.'` da seguinte forma: `'..'`. Se a posição não for vazia, ela é composto por um caractere D ($1 \leq D \leq 8$) que representa o dígito, e um caractere C, que pode ser igual a `'a'` se a posição for azul, ou `'b'` se a posição for laranja.

Saída

Imprima `'YES'` se existe uma solução para o tabuleiro, ou `'NO'` se não existe.

Exemplo

entrada padrão	saída padrão
7 7 2b3a..4b..2a.. 1a.....2b2a ..1b....1b3a3b 2b....8a..5b2a 3b..3a..... ....2b....3a3b 3a....3b1a2b2a	YES

Problema I. Indentador de BF

Arquivo de entrada:	`entrada padrão`
Arquivo de saída:	`saída padrão`
Tempo limite:	1 segundo
Limite de memória:	256 megabytes

A linguagem brainf*ck é uma linguagem de programação esotérica desenhada parar desafiar e divertir programadores. Ela foi criada em 1993 por Urban Müller, e pode ser usada para descrever uma família de máquinas de Turing. A linguagem é executada usando uma fita unidimensional de octetos (chamados de células) e um ponteiro para esta fita. A linguagem possui oito comandos:

- `‘>’` incrementa o ponteiro (move o ponteiro para a direita)
- `‘<’` decrementa o ponteiro (move o ponteiro para a esquerda)
- `‘+’` incrementa a célula apontada pelo ponteiro
- `‘-’` decrementa a célula apontada pelo ponteiro
- `‘.’` imprime na tela o valor da célula apontada pelo ponteiro
- `‘,’` lê da entrada para a célula apontada pelo ponteiro
- `‘[’` se a célula apontada pelo ponteiro for zero, pula para o `‘]’` correspondente
- `‘]’` volta para o `‘[’` correspondente

Quaisquer outros caracteres são ignorados, o que permite que o código brainf*ck seja “comentado”. Você decidiu escrever alguns programas em brainf*ck, mas como os espaços e quebras de linhas são ignorados mesmo, você começou a escrever tudo junto na mesma linha. Para permitir que seu código fique mais “legível” (como se isso fosse possível numa linguagem assim), você decidiu codar um indentador. O funcionamento é simples:

- Se existir pelo menos um caractere não-espaço entre dois comandos, é necessário que o primeiro comando seja seguido de um espaço, do “comentário” e uma quebra de linha, e em seguida, o segundo comando.
- Se existir apenas um caractere de espaço entre dois comandos, este espaço é removido.
- Os comandos `‘[’` e `‘]’` criam um nível de indentação de um espaço apenas se entre os comandos existir pelo menos um caractere ignorado não-espaço. Um nível de indentação significa que todas as linhas entre estes dois comandos devem estar com seu nível de indentação incrementado, e os comandos `‘[’` e `‘]’` ficam isolados em sua própria linha contendo o seu próprio comentário.

Implemente um programa que pega um código-fonte em brainf*ck em uma única linha e devolve o código-fonte indentado.

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^6$), que determina o tamanho do código-fonte brainf*ck. A segunda linha é código-fonte que tem exatamente N caracteres que podem ser comandos, espaços, números e letras latinas minúsculas. É garantido que o primeiro caractere é um comando. O código-fonte é bem formado em relação aos espaços: Não possui espaços no começo, no final, nem sequências de mais de um espaço.

Saída

Devem ser impresso o código-fonte indentado. Cada linha deve ser composta de um número C ($0 \leq C$), o nível de indentação, um espaço, e o texto daquela linha.

Exemplos

entrada padrão
70 ++c0 eh 2>+++++c1 eh 5[<+adiciona 1 a c0-subtrai 1 de c1] <.imprime c0

saída padrão
0 ++ c0 eh 2 0 >+++++ c1 eh 5 0 [1 <+ adiciona 1 a c0 1 - subtrai 1 de c1 0] 0 <. imprime c0

entrada padrão
72 >>c2[-]zera c2<<c0[enquanto c0-subtrai>>c2+incrementa c2<<c0]fim do laco

saída padrão
0 >> c2 0 [-] zera c2 0 << c0 0 [enquanto c0 1 - subtrai 1 >> c2 1 + incrementa c2 1 << c0 0] fim do laco

entrada padrão
47 [[]+[laco infinito dentro do laco infinito]-[]]

saída padrão
0 [1 []+ 1 [laco infinito dentro do laco infinito 1] 1 -[] 0]