

Caderno de Provas



O Poderoso Balão

I Seletiva Interna UDESC 2025

Servidor BOCA:

<https://boca.joinville.udesc.br/>

Organização e Realização:



Patrocínio:

BECOMEX

Informações do ambiente de testes

- Em todas as questões, o limite de memória é de 1 gigabyte.
- Há um limite no tamanho na pilha de 100 megabytes.
- O código-fonte submetido pode ter no máximo 100 kilobytes.

Comandos de compilação utilizados

- C: `gcc -static -O2 <nome_do_arquivo>.c -lm -o <nome_do_programa>`
- C++: `g++ -std=c++20 -static -O2 -lm <nome_do_arquivo>.cpp -o <nome_do_programa>`
- Java: `javac <nomeDoArquivo>.java`
- Kotlin: `kotlinc -J-Xms1024m -J-Xmx1024m -J-Xss100m -include-runtime <nome_do_arquivo>.kt`

Comandos de entrada e saída de dados

- C: `scanf`, `getchar`, `printf`, `putchar`
- C++: as mesmas de C ou os objetos `cout` e `cin`
- Java: `Scanner`, `BufferedReader`, `BufferedWriter` e `System.out.println`
- Python: `read`, `readline`, `readlines`, `input`, `print`, `write`

Template exemplo

A correção é automatizada, portanto siga atentamente as exigências do problema quanto ao formato da entrada e saída de seu programa. Fornecemos um exemplo de problema e um template esperado para a sua resolução:

A entrada consiste em uma linha contendo um inteiro A.

A saída consiste de uma linha contendo o inteiro B, que é igual ao inteiro A multiplicado por 2.

- C:

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int A;
    scanf("%d", &A);
    int B = A * 2;
    printf("%d\n", B);
    return 0;
}
```

- C++:

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    int A;
    cin >> A;
    int B = A * 2;
    cout << B << "\n";
    return 0;
}
```

- Java:

```
import java.io.IOException;
import java.util.Scanner;

public class C {
    public static void main(String[] args) throws IOException {
        Scanner scan = new Scanner(System.in);
        int A;
        A = scan.nextInt();
        int B = A * 2;
        System.out.printf("%d\n", B);
    }
}
```

Obs: o nome do arquivo deve ser LETRA_DO_PROBLEMA.java e a classe deve ser LETRA_DO_PROBLEMA, como no exemplo acima, se o problema solucionado fosse o problema C.

- Python:

```
A = int(input())
B = A * 2;
print(B)
```

Obs: Caso a entrada contenha vários elementos em uma única linha, como por exemplo "uma linha contendo três inteiros A, B e C" usamos o seguinte comando:

```
A, B, C = map(int, input().split())
```

C/C++

- Seu programa deve retornar zero, executando, como último comando, `return 0` ou `exit(0)`.
- É sabido que em alguns casos de problemas com entradas muito grandes, os objetos `cin` podem ser lentos, por conta da sincronização de buffers da biblioteca `stdio`. Caso deseje utilizar `cin` e `cout`, um jeito de se contornar este problema é executando o comando `ios_base::sync_with_stdio(0)`, no início de sua função `main`. Note que, neste caso, o uso de `scanf` e `printf` no mesmo programa é contra-indicado, uma vez que, com buffers separados, comportamentos inadequados podem ocorrer.
- Comando para executar uma solução C/C++: `./<nome_do_programa>`.

Java

- Não declare “**package**” no seu programa Java.
- Note que a convenção para o nome do arquivo fonte deve ser obedecida, o que significa que o nome de sua classe pública deve ser uma letra maiúscula (A, B, C, ...).
- Comando para executar uma solução Java: `java -Xms1024m -Xmx1024m -Xss20m <nomeDoArquivo>`.

Kotlin

- Não declare “**package**” no seu programa Kotlin.
- Comando para executar uma solução Kotlin: `kotlin -J-Xms1024m -J-Xmx1024m -J-Xss100m <nomeDoArquivoKt>`.
- Não é garantido que soluções em Kotlin conseguirão executar dentro do tempo limite alocado.

Python

- O Python está na versão 3.10 e em erro de sintaxe, será retornado **Runtime Error**.
- Comando para executar uma solução Python: `python3 <nome_do_arquivoPy>.py`.
- Não é garantido que soluções em Python conseguirão executar dentro do tempo limite alocado.

Instruções para uso do sistema de submissão BOCA

Tudo no BOCA é feito pela interface web, então antes de fazer qualquer ação certifique-se que:

- O navegador está aberto e na URL do BOCA que se encontra na capa.
- O BOCA está logado com o login do time (com o nome de usuário e senha fornecidos).

Submissão de soluções

Para enviar uma solução para um problema, acesse a aba **Runs**, escolha o problema apropriado, a linguagem utilizada e envie o arquivo fonte.

Resultado da submissão

Para ver o resultado de uma submissão, verifique a aba **Runs**. Os vereditos que você pode receber dos juízes são:

- | | |
|-----------------------------|-----------------------------|
| 1. YES | 6. NO - Contact staff |
| 2. NO - Compilation error | 7. NO - Class name mismatch |
| 3. NO - Runtime error | 8. NO - Wrong language |
| 4. NO - Time limit exceeded | 9. NO - Problem mismatch |
| 5. NO - Wrong answer | |

Os significados de 1, 2, 3 e 4 são auto-explicativos.

- Sobre 1 e 5:
 - Se a saída da solução do time é exatamente igual à saída dos juízes, a resposta é “YES”;
 - Caso contrário, o veredito é “Wrong Answer”.
- Sobre 6: Usado em circunstâncias inesperadas. Neste caso, utilize o menu “Clarifications” e forneça o número da run para maiores esclarecimentos.
- Sobre 7: Apenas para submissões Java, veredito quando o time submete uma solução com nome da classe principal diferente do especificado, de forma que a execução falha. Não é usado no caso de submissões C/C++ ou Python.
- Sobre 8: Principalmente para Python3, mas pode ser usado também para outras linguagens.
- Sobre 9: Identificação errada de problema ao submeter a solução. Note que nem sempre é possível distinguir entre os vereditos 3, 7, 8 e 9. Por exemplo, quando um time submete um arquivo `B.java` no qual uma classe pública `A` é definida como solução para o problema `A`, o veredito pode ser “Class name mismatch” ou “Problem mismatch”.

Esclarecimentos

Para solicitar esclarecimentos sobre o enunciado de um problema, acesse a aba **Clarifications**, escolha o problema apropriado, escreva sua dúvida e submeta.

Placar

Para ver o placar local você deve acessar a aba **Score**.

O Poderoso Balão

I Seletiva Interna UDESC 2025

07 de Junho de 2025

Conteúdo

Problema A: Absolute Cinema	7
Problema B: Biblioteca Mágica	9
Problema C: Clichês	11
Problema D: Dentro da Matriz	13
Problema E: Enigma	15
Problema F: Frodo & Sam	16
Problema G: Godfather	18
Problema H: Hierarquia Cinematográfica	20
Problema I: Idioma Perdido	22
Problema J: John Wick	23
Problema K: Kill Bill Vol. 1	24
Problema L: Las Tortuguitas	26
Problema M: Movieguessr	28
Problema N: Ni	30
Problema O: O Marciano	31

Problema A: Absolute Cinema

Arquivo: A. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

Após o episódio marcante com a cafeteria El Café, Max e seu irmão Min resolveram tirar umas férias merecidas — desta vez, longe do café e perto das telonas. A paixão deles por cinema, especialmente pelos filmes de Martin Scorsese, os levou a criar uma nova tradição: **as maratonas contínuas de Scorsese**.

Durante o mês de descanso, os irmãos organizaram N filmes do diretor em ordem cronológica, numerando-os de 1 até N . Cada filme recebeu uma pontuação da crítica especializada; a pontuação do i -ésimo filme é a_i .

Como sempre foram competitivos e criativos, Max e Min decidiram analisar a qualidade de cada “maratona contínua” de forma curiosa:

- Max, como não podia deixar de ser, se preocupa com o máximo.
- Min, claro, quer saber do mínimo.
- Para eles, a qualidade de uma maratona é definida como o produto entre o maior e o menor valor da sequência de pontuações dos filmes assistidos naquela maratona.

Uma maratona contínua é qualquer subsequência contínua da lista original de filmes — pode ser um único filme ou todos os N filmes.

Max e Min, então, ficaram curiosos: qual seria o valor total da soma das qualidades de todas as maratonas contínuas possíveis?

Como esse valor pode ser muito grande, eles decidiram que só se importam com o resultado módulo $10^9 + 7$.

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^5$), que representa o número de filmes na lista organizada por Max e Min. A segunda linha contém N inteiros $a_1, a_2, \dots, a_N$ ($1 \leq a_i \leq 10^9$), representando as pontuações dadas pela crítica especializada para cada filme.

Saída

Imprima um único inteiro, que corresponde ao resto da divisão da soma total das qualidades de todas as maratonas contínuas possíveis na lista de filmes de Max e Min por $10^9 + 7$.

Exemplos

entrada padrão	saída padrão
3 2 1 3	22
5 5 9 1 2 3	230

Notas

Explicação do exemplo 1: Para cada subsequência contínua (maratona), calculamos a qualidade como o produto do valor máximo e do valor mínimo dos filmes dentro dela:

$$2 \rightarrow \text{Máximo} = 2, \text{Mínimo} = 2, \text{Qualidade} = 2 \cdot 2 = 4$$

$$2,1 \rightarrow \text{Máximo} = 2, \text{Mínimo} = 1, \text{Qualidade} = 2 \cdot 1 = 2$$

$$2,1,3 \rightarrow \text{Máximo} = 3, \text{Mínimo} = 1, \text{Qualidade} = 3 \cdot 1 = 3$$

$$1 \rightarrow \text{Máximo} = 1, \text{Mínimo} = 1, \text{Qualidade} = 1 \cdot 1 = 1$$

$$1,3 \rightarrow \text{Máximo} = 3, \text{Mínimo} = 1, \text{Qualidade} = 3 \cdot 1 = 3$$

$$3 \rightarrow \text{Máximo} = 3, \text{Mínimo} = 3, \text{Qualidade} = 3 \cdot 3 = 9$$

A soma de todas as qualidades é $4 + 2 + 3 + 1 + 3 + 9 = 22$.

Problema B: Biblioteca Mágica

Arquivo: B. [c|cpp|java|kt|py]
 Limite de tempo: 2 segundos

Na Escola de Magia e Bruxaria de Brutewarts, há uma biblioteca com conhecimento milenar sobre magia. Na biblioteca, há várias estantes, cada uma para uma categoria de livros diferente. A bibliotecária, Karina McGonagall, é muito preocupada com a indexação dos livros nas estantes, e sua categoria favorita é a categoria dos três porquinhos.

Em uma dada categoria de livros, ela gosta de manter os livros ordenados por título em ordem lexicográfica, ou seja, na ordem que apareceriam se fossem verbetes em um dicionário. Por exemplo, a palavra “abacate” aparece antes de “abacaxi” no dicionário, pois, ao olharmos para a primeira posição em que elas diferem, a letra “t” vem antes da letra “x” no alfabeto.

Depois de muito trabalho, Karina finalmente concluiu a organização de sua estante mais ambiciosa: a estante de todas as strings finitas contendo letras de “a” a “z”. Apesar de haver alguns livros com títulos repetidos, livros intitulados com toda string finita imaginável já estavam lá, perfeitamente ordenados lexicograficamente.

Nessa estante, ela notou que os estudantes de magia quase sempre leem um dos primeiros livros, posicionados perto do começo e que era raro alguém querer ir até o final.

O título do livro favorito de Karina é S , e ela gostaria de reposicioná-lo em um lugar em que fosse mais lido. Não podendo mudar a ordenação, ela só quer mudar **um pouco** o título a fim de, ao reinseri-lo na posição correta da ordenação, deixá-lo o mais próximo possível do início. Para não prejudicar a compreensão, ela fará a seguinte alteração **no máximo uma vez**:

- Escolher duas posições i, j ($1 \leq i, j \leq N$) e trocar de lugar os caracteres S_i e S_j , ou seja, S_i vai para a posição j e vice-versa.

Como ela já está muito cansada depois de corrigir as provas de Transfiguração, a pobre bibliotecária pede a sua ajuda.

Dado o título do livro, qual título ele deve ter de forma a atender ao pedido de Karina?

Entrada

A entrada consiste em duas linhas. A primeira contém um número inteiro N ($1 \leq N \leq 10^5$), o tamanho do título do livro favorito de Karina. Em seguida, a segunda linha contém o título S , uma palavra apenas com caracteres minúsculos.

Saída

Imprima o novo título do livro após atender ao pedido de Karina.

Exemplos

entrada padrão	saída padrão
8 discreta	aiscreted
11 harrypotter	ahrrypotter
10 crepusculo	ccepusrulo
3 abc	abc

Problema C: Clichês

Arquivo: C. [c|cpp|java|kt|py]
 Limite de tempo: 1.5 segundos

Queliton Tarantino é uma figura lendária no mundo do cinema; seus filmes são conhecidos por diálogos afiados, reviravoltas espetaculares e, principalmente, por abusar de todos os clichês possíveis — sempre com aquele toque irônico.

Recentemente, Queliton está trabalhando no roteiro de sua nova obra-prima: um filme codificado como uma palavra S , composta apenas por letras minúsculas. Como todo bom roteirista de impacto, ele quer garantir que seu filme seja recheado de clichês clássicos do cinema — cada um deles representado por uma palavra.

Mas há um detalhe: Queliton quer fazer uma única jogada ousada — inverter, no máximo uma vez, um trecho contínuo do filme. Ele sabe que esse truque pode transformar uma cena comum em um momento icônico, criando novas ocorrências dos clichês e, quem sabe, garantindo sua décima Palma de Ouro.

Sua missão é ajudá-lo a descobrir qual é o máximo número de ocorrências de clichês que ele pode inserir no filme após inverter, no máximo uma vez, um segmento qualquer do roteiro.

Cada ocorrência conta, mesmo que o mesmo clichê apareça várias vezes, inclusive sobrepondo-se a outras ocorrências ou a si próprio.

Entrada

A primeira linha contém a palavra S ($1 \leq |S| \leq 2000$), o roteiro atual do filme.

A segunda linha contém um inteiro M ($1 \leq M \leq 2000$) — o número de clichês cinematográficos.

Cada uma das próximas M linhas contém uma palavra P_i ($1 \leq |P_i| \leq 2000$), representando um clichê.

O somatório total dos comprimentos das strings P_i não excede 2000.

Saída

Imprima um único inteiro: o máximo número de ocorrências de clichês que Queliton pode obter após inverter, no máximo uma vez, um trecho contínuo qualquer de S .

Exemplos

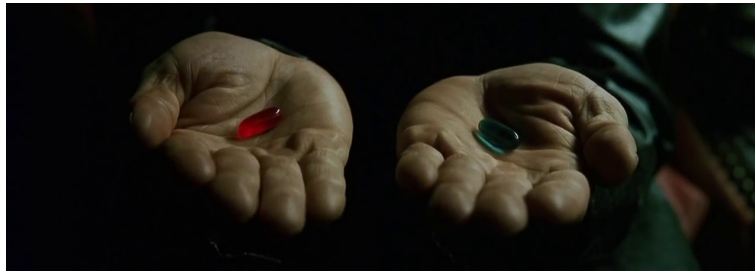
entrada padrão	saída padrão
abbaca 3 cab bbac aa	2
abcda 2 da bc	2
aaaaaabaa 1 abaaa	1
aabababb 4 ab bab abab baba	12

Problema D: Dentro da Matriz

Arquivo: D. [c|cpp|java|kt|py]

Limite de tempo: 1 segundo

Lucas Thomas Anderson trabalhava em um escritório e mantinha uma vida normal, exceto pelo fato de ser um hacker no tempo livre e descobrir que está preso em uma matriz $N \times M$! Ele está atualmente na célula $(1, 1)$ e, para sair da matriz, precisa chegar até Morfeu, que se encontra na célula (N, M) .



Porém, Agente Gui também quer encontrar Morfeu para acabar com os planos de Thomas antes que ele saia da matriz e, para isso, altera a realidade, preparando um desafio centrado no OR bitwise, pois Gui sabe que Thomas, apesar de ser um hacker experiente, tem mais afinidade com outras operações.

Nessa realidade alterada pelo Agente Gui, o tempo que qualquer pessoa demora para andar por um caminho na matriz é o OR bitwise dos valores escritos nas células por onde se passou. Também, como Gui é um programa movido a inteligência artificial, ele consegue facilmente andar pela matriz e encontrar o caminho mais rápido para onde quer chegar.

Agora, o Agente Gui está na célula (X, Y) e quer chegar a Morfeu antes de Thomas. Gui não está no mesmo lugar que Thomas nem Morfeu. Tanto Gui quanto Thomas podem andar de uma célula a outra se as duas compartilham uma aresta (ou seja, para cima, para baixo, para a esquerda ou para a direita).

O destino da humanidade está em suas mãos! Dadas a posição de Agente Gui e o valor de cada célula, escreva um programa que diz para Thomas o menor tempo que ele pode levar para chegar até Morfeu e também o tempo que Gui levaria.

Entrada

A primeira entrada contém dois inteiros N e M ($3 \leq N \cdot M \leq 10^5$), as dimensões da matriz.

A segunda linha da entrada contém dois inteiros X e Y , a posição do Agente Gui na matriz.

Seguem N linhas. A i -ésima dessas ($1 \leq i \leq N$) contém M valores e o j -ésimo desses contém o valor na célula (i, j) da matriz ($0 \leq V_{i,j} \leq 2 \cdot 10^9$).

É garantido que o valor das células onde estão Thomas, Gui e Morfeu é 0.

Saída

Imprima dois valores, o tempo que Thomas e Gui levam para chegar até Morfeu.

Exemplos

entrada padrão	saída padrão
3 3 1 3 0 2 0 1 5 6 1 1 0	1 3
3 4 3 1 0 2 4 8 4 8 2 0 0 2 3 0	6 3
3 4 3 1 0 2 3 8 4 8 2 0 0 2 4 0	3 6
3 3 1 2 0 0 1 1 1 1 1 1 0	1 1

Problema E: Enigma

Arquivo: E. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

Leo Turing é um cientista com descobertas incríveis, um dos mais renomados da Ciência da Computação; seu trabalho mais influente é o seu estudo sobre criptografia em mecanismos de busca.

Nesse exato momento, ele está tentando quebrar o código da criptografia para descobrir o histórico de buscas do inimigo na Segunda Hackathon Mundial e se deparou com um desafio novo.

Turing quer encontrar dois valores desconhecidos A e B , muito importantes para decifrar buscas inimigas. Porém, os únicos valores que ele conhece são X e Y , que representam o seguinte:

- $X = A + B$
- $Y = |A - B|$

Para encontrar esses valores, ele pediu a sua ajuda. Dados X e Y , imprima dois valores A e B que satisfaçam as condições acima. Turing afirma que A e B existem e são unicamente determináveis para as entradas que ele colocará no seu programa.

Entrada

A entrada contém duas linhas. A primeira dessas contém o número X ($0 \leq X \leq 10^9$) e a segunda, o número Y ($0 \leq Y \leq 10^9$).

Saída

Imprima dois valores A e B que satisfaçam as condições do problema. Você pode imprimi-los em qualquer ordem.

Exemplos

entrada padrão	saída padrão
10 2	4 6
3 7	-2 5
10 10	0 10

Problema F: Frodo & Sam

Arquivo: F. [c|cpp|java|kt|py]

Limite de tempo: 1 segundo

Frodo e Sam, os **grandes** hobbits da Terra-média, estão em uma aventura para destruir o Um Anel. A viagem os leva às traiçoeiras Minas de Moria, onde habita um Balrog — uma das criaturas mais temidas de Morgoth.



Ao adentrarem as profundezas de Moria, algo extraordinário acontece — **é aqui que o plot muda!** O Balrog desperta e lança uma magia poderosa sobre Frodo e Sam, transportando-os para outro plano da realidade.

Frodo e Sam foram levados para o topo de uma montanha, e nela existiam dois caminhos de rochas mágicas, onde Frodo estava à esquerda desses caminhos e Sam à direita. Os caminhos das rochas podem ser representados por duas permutações P e Q de tamanho N , onde o tipo da i -ésima rocha da esquerda para a direita é P_i no primeiro caminho e Q_i no segundo caminho.

Para escapar deste mundo ilusório e retomar a jornada rumo a Mordor, Frodo e Sam precisam destruir algumas dessas rochas de modo que as duas permutações P e Q dos tipos de rochas nos caminhos se tornem idênticas, mas **não podem destruir todas as rochas dos dois caminhos**.

Frodo e Sam começam a destruir as rochas ao mesmo tempo. Para cada um dos dois caminhos, Frodo pode destruir quantas rochas quiser a partir da esquerda de ambos os caminhos, e Sam pode fazer o mesmo a partir da direita. Mais formalmente, Frodo irá escolher dois inteiros (a, b) e destruir as rochas $P_1, P_2, \dots, P_a$ e $Q_1, Q_2, \dots, Q_b$, já Sam irá escolher dois inteiros (c, d) e destruir as rochas $P_N, P_{N-1}, \dots, P_c$ e $Q_N, Q_{N-1}, \dots, Q_d$. Note que Frodo e/ou Sam podem escolher não destruir nenhuma rocha.

Sabendo que Frodo demora F tempo para destruir uma rocha, Sam demora S para fazer o mesmo e que o tempo de deslocamento entre duas rochas quaisquer é 0, qual é o menor tempo que eles precisam gastar destruindo rochas para sair da realidade mágica do Balrog?

Entrada

A primeira linha da entrada consiste em três inteiros N ($1 \leq N \leq 10^5$), F e S ($1 \leq F, S \leq 10^9$), a quantidade de rochas que separam Frodo e Sam, o tempo que Frodo demora para destruir uma rocha e o tempo que Sam demora para destruir uma rocha, respectivamente.

A segunda linha da entrada contém uma permutação de N inteiros $P_1, P_2, \dots, P_N$, ($1 \leq P_i \leq N$) onde P_i representa o tipo da i -ésima rocha do primeiro caminho da esquerda para a direita.

A terceira linha da entrada contém uma permutação de N inteiros $Q_1, Q_2, \dots, Q_N$, ($1 \leq Q_i \leq N$) onde Q_i representa o tipo da i -ésima rocha do segundo caminho da esquerda para a direita.

Saída

A saída deve conter um inteiro, a resposta para o problema.

Exemplos

entrada padrão	saída padrão
3 1 2 1 2 3 3 1 2	2
1 1 2 1 1	0
8 24 16 7 4 8 1 2 3 6 5 3 2 7 4 5 8 1 6	144
7 9 6 7 5 1 6 2 3 4 3 2 5 1 6 4 7	30

Notas

No terceiro caso, Frodo e Sam podem remover rochas até deixar $P = [2]$ e $Q = [2]$, assim, Frodo precisa destruir 5 rochas no total, demorando 120 unidades de tempo, e Sam precisa destruir 9 rochas no total, demorando 144 unidades de tempo.

Já no quarto caso, Frodo e Sam podem remover rochas até deixar $P = [5, 1, 6]$ e $Q = [5, 1, 6]$, assim, Frodo precisa destruir 3 rochas no total, demorando 27 unidades de tempo, e Sam precisa destruir 5 rochas no total, demorando 30 unidades de tempo.

Problema G: Godfather

Arquivo: G. [c|cpp|java|kt|py]

Limite de tempo: 2 segundos

Joãozão e Eric, dois renomados roteiristas, estão trabalhando juntos no filme *Godfather*. Ao chegarem à etapa de criar o clímax da obra, perceberam que precisavam de uma cena realmente insana para surpreender o público.

Joãozão quer deixar a cena o mais insana possível, tentando **maximizar** o nível de loucura da história. Por outro lado, Eric quer manter o equilíbrio e impedir exageros, tentando **minimizar** esse nível. Assim, o diretor do filme — Bras, decidiu usar as diferenças de ideias dos dois para chegar a um ponto de equilíbrio criativo.

O processo criativo ocorre ao longo de N **dias**. Bras usa uma palavra binária S para auxiliá-lo. No i -ésimo dia, Bras escolhe quem entre Joãozão e Eric fará a alteração nos plots seguindo essa palavra:

- Se $S_i = 1$, Joãozão faz a mudança.
- Se $S_i = 0$, Eric faz a mudança.

Para iniciar o processo, Bras escolheu dois plot twists: um **plot principal** com nível de loucura A , e um **plot auxiliar** com nível de loucura B .

Quem for escolhido para fazer a alteração no dia i , vai começar o dia com dois valores X e Y , o nível de loucura atual do plot principal e do auxiliar, respectivamente. Então, essa pessoa cria um novo plot principal com valor $X + Y$, e escolhe um entre X e Y para continuar como valor do plot auxiliar. Note que sempre existe exatamente um plot principal e um plot auxiliar.

Bras ainda não tem certeza de como escolher quem vai mudar o plot em qual dia, por isso, ele quer fazer uma simulação de possíveis cenários. Dado o S inicial, ele irá realizar Q operações, cada uma troca as escolhas nos dias de um intervalo $[L, R]$. Isto é, para todo i tal que $L \leq i \leq R$, o valor de S_i é invertido: se era 0 vira 1, se era 1 vira 0.

Sua tarefa é fazer essa simulação; após cada uma das Q mudanças, determine qual será o nível de loucura final do plot principal, sabendo que Joãozão quer maximizar a loucura final e Eric quer minimizar a mesma. Como o nível de loucura pode ser muito grande, Bras estará contente em saber apenas o resultado módulo 998244353.

Entrada

A primeira linha da entrada contém três inteiros N ($1 \leq N \leq 2 \cdot 10^5$), A e B ($1 \leq A, B < 10^6$), a quantidade de dias para construir o plot, o nível de loucura do plot principal e auxiliar, respectivamente.

A segunda linha da entrada contém a palavra binária S de tamanho N que Bras usa para escolher quem irá fazer a mudança no plot.

A terceira linha da entrada contém um inteiro Q ($1 \leq Q \leq 2 \cdot 10^5$), a quantidade de alterações que Bras irá pedir.

As próximas Q linhas contém dois inteiros L e R ($1 \leq L \leq R \leq N$), significando que para todo i tal que $L \leq i \leq R$, se antes Joãozão modificava o plot no i -ésimo dia, agora Eric é quem muda, e vice-versa.

Saída

A saída deve conter Q linhas; a i -ésima delas deve conter a loucura do plot twist principal depois das primeiras i atualizações módulo 998244353.

Exemplos

entrada padrão	saída padrão
8 4 5 11101011 8 1 8 1 6 4 7 4 6 1 8 2 6 8 8 5 7	131 162 226 213 131 153 153 170
1 8 2 1 5 1 1 1 1 1 1 1 1 1 1	10 10 10 10 10
6 3 19 111111 3 1 5 2 5 4 5	37 191 122

Problema H: Hierarquia Cinematográfica

Arquivo: H. [c|cpp|java|kt|py]
 Limite de tempo: 2 segundos

O BRUTE, um grupo originalmente conhecido por sua performance em competições de programação competitiva, decidiu mudar de rumo e se aventurar no mundo do cinema. Risa, uma renomada diretora do grupo, recebeu a missão de selecionar alguns atores dentro de N candidatos para o primeiro filme produzido pelo BRUTE.

Para isso, Risa entrou em contato com todos os atores e coletou suas exigências para participarem do projeto. Cada ator, além de estipular um valor em dinheiro, indicou nomes de outros atores **mais famosos** com os quais gostaria de trabalhar, exigindo que esses também fossem incluídos no elenco. Cada ator é identificado por um número, sendo que o i -ésimo ator menos famoso é representado pelo número i , ou seja, o ator mais famoso é representado pelo número N , o próximo mais famoso pelo número $N - 1$, e assim por diante.

Após organizar todas as exigências, Risa percebeu duas coisas:

- Cada ator aparece em **no máximo uma** exigência de outro ator.
- Existe **exatamente um** ator que não é exigido por nenhum outro, sendo ele o ator menos famoso entre todos.

Sabendo que o i -ésimo ator gera um lucro de a_i caso seja contratado (valores negativos representam prejuízo), qual é o maior lucro possível que o BRUTE pode obter se Risa escolher otimamente os atores a serem contratados? Note que Risa pode escolher não contratar nenhum ator.

Entrada

A primeira linha da entrada contém um inteiro N ($2 \leq N \leq 5 \cdot 10^5$), representando o número de candidatos que Risa tem à disposição.

A segunda linha contém N inteiros $a_1, a_2, \dots, a_N$ ($-10^9 \leq a_i \leq 10^9$), onde a_i representa o lucro que o i -ésimo ator gerará caso seja contratado para o filme.

A terceira e última linha contém $N - 1$ inteiros $s_2, s_3, \dots, s_N$ ($1 \leq s_i < i$), indicando que o ator s_i exigiu a participação do ator i no elenco.

Saída

A saída deve conter exatamente uma linha, o maior lucro possível que o BRUTE pode obter se Risa escolher otimamente os atores a serem contratados.

Problema I: Idioma Perdido

Arquivo: I. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

Rondônia Jones é um arqueólogo e explorador conhecido por seu amplo conhecimento sobre civilizações antigas e suas linguagens.

Em uma de suas expedições, encontrou uma pedra do povo brutense (datada de 998244353 A.C.) com um breve texto introdutório seguido de N inscrições misteriosas: todas eram palavras contendo letras do alfabeto latino (de “a” a “z”). O texto explicava, em um idioma há muito não falado, que todas essas palavras eram não-decrescentes. Isso levou Jones a esperar que elas se parecessem com “abcdef”, ou com “aacgmz”, palavras não decrescentes se considerarmos a ordem do alfabeto latino. Diferentemente do esperado, as letras tinham ordens estranhas entre si, que ele não reconhecia.

Agora, Jones quer saber se existe alguma ordem para as letras do alfabeto latino que faça com que as palavras inscritas respeitem a regra do texto. Se sim, ele pede a você que imprima uma possível ordem do alfabeto que torne isso verdade, pois isso seria crucial para seus estudos sobre o povo brutense.

Entrada

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 10^5$).

Seguem N linhas. A i -ésima delas contém uma palavra s_i .

É garantido que o somatório de $|s_i|$ é menor ou igual a 10^6 .

Saída

Imprima uma linha com “SIM” ou “NAO”, se existir ou não resposta para o problema.

Em caso positivo, imprima mais uma linha com uma permutação das 26 letras do alfabeto latino que satisfaça os requisitos. Se houver múltiplas respostas possíveis, imprima qualquer uma.

Exemplos

entrada padrão	saída padrão
3 abc def ghi	SIM abcdefghijklmnpqrstuvwxy
3 acb zcb ghi	SIM adefghijklmnpqrstuvwxyzb
4 brute bruta bruto trem	NAO

Problema J: John Wick

Arquivo: J. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

John Wick Fröhlich finalmente conseguiu escapar do misterioso labirinto, mas ao se afastar percebeu que havia derrubado sua bússola de navegação — um item vital — exatamente no corredor central, uma estreita passagem que lentamente se fecha. Agora, ele precisa se afastar **exatamente** K segundos para recuperar o artefato e voltar antes que a passagem se torne intransponível.

O corredor é delimitado por duas paredes, representadas pelo caractere “#”. Entre elas, existem pedras indestrutíveis (caractere “o”) e espaços vazios (caractere “.”), que já faziam parte do labirinto.

A cada segundo, a parede **esquerda** avança uma posição para a **direita**, empurrando imediatamente qualquer pedra (o) que estiver no espaço à sua frente.

Pedras empurradas podem, por sua vez, empurrar outras pedras em cadeia, mas **nunca** são destruídas ou sobrepostas. As paredes também são indestrutíveis e não podem ser sobrepostas.

Antes de partir, John Fröhlich quer saber como ficará a configuração do corredor **após exatamente** K segundos de movimento das paredes, para garantir que conseguirá retornar em segurança.

Entrada

A primeira linha contém dois inteiros N e K ($3 \leq N \leq 500$, $0 \leq K \leq 500$) — o comprimento inicial do corredor e o tempo em segundos que John Fröhlich ficará afastado.

A segunda linha contém uma string de comprimento N , composta por:

- Exatamente dois caracteres “#” nas extremidades (as paredes móveis),
- Uma combinação de caracteres “o” (pedras) e “.” (espaços vazios) entre elas.

Saída

Imprima uma única linha com a string de comprimento N que representa o estado do corredor **após** K segundos de movimentos.

Exemplos

entrada padrão	saída padrão
8 3 #o..o.o#	...#ooo#
6 3 #....#	...#.#
6 2 #oooo#	#oooo#
13 4 #...o..o.ooo#	#o.o.ooo#

Problema K: Kill Bill Vol. 1

Arquivo: K. [c|cpp|java|kt|py]

Limite de tempo: 1 segundo

Quando a Noiva pediu a Hanzo, o maior e melhor forjador de katanas que o mundo já viu, para que fizesse uma nova arma, o mestre ficou apreensivo, já que fazia décadas desde sua última fabricação e o processo era deveras complicado.



Mesmo assim, por simpatizar-se com a história e missão da Noiva, Hanzo resolveu ligar para você — a pessoa mais inteligente que conhece — e explicar passo a passo seu método de forja:

- Hanzo inicialmente monta a katana anexando uma lâmina de aço japonês num cabo e amola a lâmina da maneira convencional.
- Após isso, o mestre divide a lâmina em N partes e designa um valor a cada uma delas com base em seu nível de afiação;
- Agora, Hanzo busca minimizar a quantidade de valores de afiação distintos na lâmina escolhendo duas partes distintas da lâmina que possuem o mesmo nível de afiação e modificando todas as partes entre as duas escolhidas para que tenham esse mesmo nível.

Mais formalmente, sendo a_i o valor de afiação da i -ésima parte da lâmina de Hanzo, a operação é da seguinte forma:

1. Escolha dois índices i e j ($1 \leq i < j \leq N$), designando duas partes da lâmina sendo $a_i = a_j$.
2. Para todo k ($i < k \leq j$), atribua $a_k := a_i$.

Hanzo afirmou que a afiação da lâmina é a parte mais importante para a fabricação de uma boa katana. Porém, também é de longe a etapa mais difícil e ele sente que nem sempre consegue

tirar máximo proveito da lâmina. Portanto, ajude-o a minimizar o número de valores distintos de afiação desenvolvendo um programa que recebe os níveis de afiação iniciais da lâmina e realiza a operação de Hanzo qualquer número de vezes.

Entrada

A entrada possui duas linhas. A primeira contém apenas um inteiro N ($1 \leq N \leq 2 \cdot 10^5$), o número de partes da lâmina divididas por Hanzo. A segunda linha possui N números ($1 \leq a_i \leq 10^9$), os valores de afiação de cada uma das partes da lâmina.

Saída

Imprima um único inteiro, a quantidade mínima de valores distintos de afiação da lâmina utilizando o processo descrito pelo mestre Hanzo.

Exemplos

entrada padrão	saída padrão
5 1 2 2 2 1	1
7 1 2 3 1 4 5 4	2
3 1 2 3	3

Problema L: Las Tortuguitas

Arquivo: L. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

Dudu está em uma viagem mágica dentro da Fantástica Fábrica de Chocolate e descobriu uma sala secreta cheia de máquinas que produzem doces raríssimos chamados “Tortuguitas”. Essa sala possui N máquinas numeradas de 1 a N , e toda máquina produz exatamente K Tortuguitas, enumeradas de 1 a K .

Uma máquina é representada pela raridade das Tortuguitas que ela fabrica. Mais formalmente, a i -ésima máquina produz K Tortuguitas com raridade $r_{i,1}, r_{i,2}, \dots, r_{i,K}$. Sendo X o valor de raridade da Tortugueta mais simples, a raridade de uma Tortugueta pode ser calculada da seguinte maneira.

- $r_{i,j} = j \cdot X$ para todo $(1 \leq j \leq K)$, quando $i = 1$.

- $r_{i,j} = \sum_{t=1}^j r_{i-1,t}$, quando $i > 1$.

Dudu ficou maravilhado com essas máquinas mágicas e pediu sua ajuda para responder algumas perguntas sobre elas.

Ele vai te dar Q perguntas do tipo: Na i -ésima máquina, qual é a soma das raridades das Tortuguitas com número maior ou igual a L e menor ou igual a R ?

Mais formalmente, Dudu vai pedir o resultado de $\sum_{j=L}^R r_{i,j}$.

Como esse resultado pode ser absurdamente grande, ele pediu para você apenas o resto dessa soma quando dividida por 998244353.

Entrada

A primeira linha da entrada consiste de 4 inteiros N , K , Q e X ($1 \leq N, K, Q \leq 3 \cdot 10^5$) ($1 \leq X < 998244353$), o número de máquinas, a quantidade de Tortuguitas em cada máquina, a quantidade de perguntas que Dudu fez e a raridade da Tortugueta mais simples, respectivamente.

As próximas Q linhas contêm 3 inteiros i , L e R ($1 \leq i \leq N$) ($1 \leq L \leq R \leq K$), o número da máquina e o intervalo que Dudu deseja saber a soma.

Saída

A saída deve conter Q linhas, cada linha deve ter o resultado da soma módulo 998244353.

Exemplos

entrada padrão	saída padrão
3 10 3 5 1 4 7 2 4 7 3 4 7	110 370 975
50 100 2 7 4 1 10 50 3 100	14014 574802200
1 1 1 1 1 1 1	1

Problema M: Movieguessr

Arquivo: M. [c|cpp|java|kt|py]
 Limite de tempo: 1 segundo

O Ano é 62442 e Machado e Marcel estavam jogando o jogo favorito deles, **Movieguessr**.

Movieguessr é um jogo famoso por teleportar seus jogadores para o universo de um filme, onde eles têm um tempo determinado para descobrir em qual obra estão.

Quando Machado e Marcel acharam que seria apenas mais um round do jogo, o inacreditável aconteceu — os servidores de Movieguessr travaram, e eles ficaram presos dentro de uma das escolas mais clássicas dos filmes da era antiga, **Hogwarts**.

Os dois estavam na famosa escadaria mágica de Hogwarts, porém em pontos distintos dela, mais especificamente:

- A escadaria é representada por N degraus enumerados de 1 a N e uma lista a para representá-los. O primeiro degrau está no nível do chão ($a_1 = 0$), o resto pode estar acima ($a_i = 1$) ou abaixo ($a_i = -1$) do degrau anterior: ao subir um degrau, a altura da pessoa aumenta em **uma** unidade; ao descer, diminui em **uma** unidade.
- Marcel encontra-se no K -ésimo degrau.
- Machado encontrou uma varinha mágica que lhe permite teleportar-se para qualquer degrau vazio da escada — ou seja, todo degrau, exceto aquele que Marcel ocupa.

Machado deseja usar essa magia para localizar seu amigo. Sabendo que Machado só consegue enxergá-lo na escadaria **se e somente se** sua altura for **maior ou igual** à de Marcel, e considerando que Machado tem A unidades de altura e Marcel, B , sua missão é ajudar Machado: existe algum degrau em que Machado vai conseguir ver Marcel? Se sim, em qual degrau ele deve se posicionar para finalmente encontrar Marcel?

Entrada

A primeira linha da entrada contém 3 inteiros N , A e B ($2 \leq N \leq 2 \cdot 10^5$) ($1 \leq A, B \leq 10^9$), a quantidade de degraus na escadaria, a altura de Machado e de Marcel, respectivamente.

A segunda linha da entrada contém N inteiros $a_1, a_2, \dots, a_N$, onde a_i é 1 se o i -ésimo degrau está acima do degrau anterior e -1 se está abaixo; além disso, $a_1 = 0$.

A terceira linha da entrada contém 1 inteiro K ($1 \leq K \leq N$), o degrau que Marcel está ocupando.

Saída

A primeira linha da saída deve conter “SIM” (sem aspas) caso exista algum degrau em que Machado consiga encontrar Marcel ou “NAO” (sem aspas) caso contrário. Se a resposta for “SIM”, a segunda linha da saída deve conter um número, a posição que Machado deve se posicionar para encontrar Marcel; se existir mais de um degrau possível, você pode usar qualquer um deles como resposta.

Exemplos

entrada padrão	saída padrão
5 10 8 0 1 -1 -1 1 2	SIM 1
11 2 4 0 1 1 -1 -1 -1 1 1 -1 1 -1 10	NAO
8 19 16 0 -1 1 1 1 1 1 1 8	SIM 5

Problema N: Ni

Arquivo: N. [c|cpp|java|kt|py]

Limite de tempo: 1 segundo

Em um dia comum passeando pela floresta, Luiza se depara com um grupo curioso de pessoas: os Cavaleiros que Dizem “Ni”.

Luiza ficou atordoada com o tanto de vezes que eles disseram a palavra “Ni”, mas ela notou que eles disseram “Ni” exatamente K vezes.

Escreva um programa que imprima exatamente o que eles disseram.

Entrada

A única linha da entrada contém um inteiro K ($1 \leq K \leq 5$), a quantidade de vezes que foi dito “Ni”.

Saída

Imprima a resposta do problema. Cada ocorrência de “Ni” precisa estar separada da outra por espaço em branco e você pode imprimir com qualquer capitalização.

Exemplos

entrada padrão	saída padrão
5	Ni Ni Ni Ni Ni
3	Ni Ni Ni

Problema O: O Marciano

Arquivo: 0. [c|cpp|java|kt|py]

Limite de tempo: 1 segundo

Ao chegar em Marte, o astronauta Mark Watney e sua equipe descobriram que o planeta não era exatamente como achavam. Seu terreno, na verdade, era repleto de grandes ravinas, buracos no chão, que dificultavam muito a travessia da equipe pelo planeta vermelho.

Por sorte, os astronautas trouxeram consigo um conjunto de escadas dos mais diversos tamanhos; escadas essas que permitiam a travessia das ravinas quando estendidas e colocadas na horizontal se seu tamanho, quando estendida, fosse maior ou igual ao tamanho do buraco.

A equipe possui uma tabela informando uma coordenada de interesse que devem visitar para cada dia e, para realizar essas visitas, escolhem uma das escadas do conjunto para ajudar nas travessias de ravinas. Após chegar ao ponto de interesse, os astronautas devem voltar pelo mesmo caminho que vieram (até chegarem na região em que eles residem) e descartar a escada usada por desgaste, não podendo utilizá-la em futuros dias.

Sabendo disso, Mark ficou encarregado de desenvolver um programa que, tendo sido fornecido um mapa com as regiões separadas pelos buracos no terreno, o tamanho das ravinas entre regiões, o conjunto de escadas trazidas, e as cidades que a equipe deseja visitar em cada dia, maximiza a quantidade de regiões de interesse visitadas durante o período inteiro da missão. Mark, como programador, é um ótimo biólogo; e, portanto, ligou para você pedindo ajuda.

O mapa de Marte é dado por um grafo de N regiões enumeradas de 1 a N e M ravinas enumeradas de 1 a M que conectam as regiões, onde a i -ésima ravina tem w_i metros de comprimento. Também é fato que se fosse possível usar uma escada de tamanho infinito, seria possível viajar de qualquer região para qualquer outra região apenas usando as ravinas.

Além disso, a missão irá durar D dias, onde a região que deve ser visitada no i -ésimo dia é d_i e a equipe possui exatamente D escadas enumeradas de 1 a D onde o tamanho da i -ésima escada é e_i . Note que uma escada pode ser utilizada em qualquer dia, mas apenas uma vez e; durante um dia, a escada escolhida pode ser utilizada quantas vezes for necessário.

Sabendo que a equipe reside na região 1, crie o programa e ajude a equipe de Mark em sua missão!

Entrada

A primeira linha do input consiste em dois números N e M ($2 \leq N \leq 2 \cdot 10^5$) ($1 \leq M \leq 2 \cdot 10^5$), o número de regiões e ravinas em Marte, respectivamente.

As próximas M linhas são compostas de três números u_i , v_i e w_i , ($1 \leq u_i, v_i \leq N$) ($1 \leq w_i \leq 10^9$) que indicam a existência de uma ravina de tamanho w_i entre as regiões u_i e v_i . É garantido que existe apenas uma ravina que conecta duas regiões, e uma ravina nunca conecta uma região consigo mesma.

Depois, é dada uma linha com um único número D ($1 \leq D \leq N$), o número de dias da missão; seguida por duas linhas com D inteiros cada.

A primeira dessas linhas contém $d_1, d_2, \dots, d_D$ ($1 \leq d_i \leq N$), informando que a i -ésima região a ser visitada é a de número d_i .

A próxima linha contém $e_1, e_2, \dots, e_D$ ($1 \leq e_i \leq 10^9$), informando que a i -ésima escada tem e_i metros de comprimento.

Saída

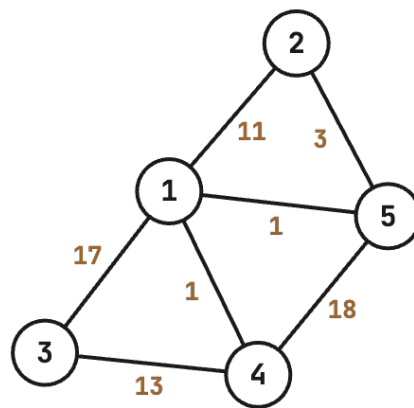
Imprima uma única linha com o número máximo de regiões que podem ser visitadas por Mark e sua equipe.

Exemplos

entrada padrão	saída padrão
6 10 5 3 4 3 6 3 1 6 2 4 1 7 2 1 15 6 4 19 3 2 10 5 1 7 4 2 17 5 2 5 6 1 2 3 4 5 6 3 4 3 5 5 5	5
5 7 2 5 3 5 4 18 5 1 1 4 1 1 4 3 13 3 1 17 2 1 11 3 5 3 2 1 11 1	2
5 6 2 1 13 4 2 1 5 2 1 5 1 2 3 2 11 4 1 5 2 2 3 1 1	0

Notas

Abaixo o mapa para o segundo exemplo:



Podemos utilizar uma das escadas de tamanho dois para chegar até o nodo 5 no primeiro dia; depois, podemos ignorar o ponto de interesse no segundo dia e utilizar a escada de tamanho 11 para chegar até o nodo 2 no terceiro dia (pelo caminho $1 - 5 - 2$). Note que pode haver mais de um jeito de obter a resposta ótima.