12th KAIST ICPC Mock Competition

Sponsored By:

MOLOCO

NAVER D2

DEVSISTERS

SAMSUNG SOFTWARE MEMBERSHIP

STARTLINK

NEXON

Rules

- This contest is 12th KAIST ICPC Mock Competition.
- This contest is sponsored by MOLOCO, Naver D2, DEVSISTERS, Samsung Software Membership, STARTLINK and NEXON.
- You can only participate in a team of three students.
- Use of the network is prohibited during the competition, except for submitting source codes and accessing language reference sites, and using translation. Here are the allowed reference and translation sites.
  - `C/C++ : https://en.cppreference.com/w/`
  - `Java : https://docs.oracle.com/javase/8/docs/api/`
  - `Python : https://docs.python.org/`
  - `Kotlin : https://kotlinlang.org/docs/reference/`
  - `Naver Papago : https://papago.naver.com/`
  - `Google Translate: https://translate.google.com/`
- The contest lasts for 5 hours.
- The contest consists of 13 problems.
- Solutions to problems submitted for judging are called runs. Each run is judged as accepted or rejected, and the team is notified of the results.
- Penalty time is the sum of
- Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked by penalty time.
- Each problem has a time limit and memory limit. This means your problem should run in the given time and memory limit for each test case.
- Every problem is guaranteed to be solvable using one of the `C++17`. This is not guaranteed for any other language.
- Problems may have different time limits.
- The memory limit for every problem is 1024MiB.
- Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked first by least total time and, if need be, by the earliest time of submission of the last accepted run.
- The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submission of the first accepted run, rounded down to the minutes, plus 20 penalty minutes for every previously rejected run before the first accepted run for that problem. There is no time consumed for a problem that is not solved.

Language Guide

- You can choose your programming language from among the following:

```
C11   :  gcc (GCC) 11.1.0
C++17 :  g++ (GCC) 11.1.0
Java  :  openjdk version "16.0.1" 2021-04-20
Kotlin:  kotlinc-jvm 1.6.10 (JRE 1.8.0_201-b09)
PyPy 3:  Python 3.9.12, PyPy 7.3.9 with GCC 10.2.1 20210130 (Red Hat 10.2.1-11)
```

- In `Java`, your class name which includes the `main` method should be `Main`.
- Compilation commands are available at the contest system.

Sample Code

- Followings are sample codes which reads two space-separated integers from standard input, and prints their sum to standard output.

  - C11 / C++17

```
#include <stdio.h>

int main() {
    int a, b;
    scanf("%d%d", &a, &b);
    printf("%d\n", a+b);
    return 0;
}
```

  - Java

```
import java.util.*;

public class Main{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        int a, b;
        a = sc.nextInt();
        b = sc.nextInt();
        System.out.println(a + b);
    }
}
```

  - Kotlin

```
import java.util.Scanner

fun main(args: Array<String>) {
    val sc: Scanner = Scanner(System.`in`)
    var a = sc.nextInt()
    var b = sc.nextInt()
    println(a+b)
}
```

  - PyPy 3

```
import sys

def main():
    a, b = map(int, sys.stdin.readline().split())
    print(a+b)

if __name__ == '__main__':
    main()
```

Problem list

#	Problem Name	Time limit (All languages)
A	Building Bombing	3 seconds
B	Connecting Cables	2 seconds
C	Double-Colored Papers	3 seconds
D	Histogram Sequence 4	1 second
E	Making Number	1.5 seconds
F	One Path	1 second
G	Permutation Arrangement	1 second
H	Set and Sequence and Query	2 seconds
I	Similarity Graph	1 second
J	Squirrel Game	3 seconds
K	Two Paths	5 seconds
L	Village Planning	3 seconds
M	Window Arrangement	3 seconds

Problem A. Building Bombing

Time limit: 3 seconds

KAIST has a series of N buildings in a row, numbered from 1 to N, from left to right. Building i has a height of h_i. Building i is visible from the left if and only if every building on its left has a height strictly less than h_i.

Your lab is located in building number L. Since your favorite number is K, you want to make your lab building the K-th tallest building visible from the left. To achieve your goal, you will blow up some of the buildings.

For example, suppose there are $N = 7$ buildings in a row and their heights are $[10, 30, 90, 40, 60, 60, 80]$. Your lab is located at building number $L = 2$ and your favorite number is $K = 3$. After blowing up buildings 3 and 7, the buildings visible from the left will be buildings 1, 2, 4, and 5. Then your lab becomes the 3rd tallest building visible from the left, as desired.

What is the minimum number of buildings to blow up to make your lab building the K-th tallest building visible from the left?

Input

The first line contains three space-separated integers N, L, and K.

The second line contains N space-separated integers $h_1, \ldots, h_N$.

Output

Output the minimum number of buildings to blow up to make your lab building the K-th tallest building visible from the left. If it is impossible to do so, output -1 instead.

Constraints

- $1 \leq L \leq N \leq 100\,000$
- $1 \leq K \leq 10$
- $1 \leq h_i \leq 10^9$ $(1 \leq i \leq N)$

Examples

standard input	standard output
7 2 3 10 30 90 40 60 60 80	2
3 2 2 30 20 10	-1

Problem B. Connecting Cables

Time limit: 2 seconds

Recently, RUN was asked to connect cables between all pairs of the N areas of KAIST.

We treat the areas as regions on the 2-dimensional plane. The boundary of each region is a 4-sided polygon with 2 edges parallel to the x-axis, and 2 edges parallel to the y-axis. In other words, each region has a rectangular boundary with (x_1^i, y_1^i) as a lower left corner and (x_2^i, y_2^i) as a upper right corner. The regions **may overlap**.

The cables must be constructed along the x-axis or the y-axis, due to safety issues. So the cost of constructing a cable from (x_1, y_1) to (x_2, y_2) is $|x_1 - x_2| + |y_1 - y_2|$ won.

A cable connecting two areas A and B should connect two points, one from each region.

Find the minimum sum of the cost for connecting $\binom{N}{2}$ cables between all pairs of the areas.

Note that the cables must be constructed for all $\binom{N}{2}$ pairs of areas. This means, for example, even if two endpoints of a cable belong to more than one pair of areas, we do not consider it as connecting all such pairs.

Since the answer can be large, output it modulo 998 244 353. It can be proved that the answer is always a non-negative integer.

Input

The first line contains one integer, N.

The i-th of the following N lines contain space-separated four integers x_1^i, y_1^i, x_2^i, and y_2^i — indicating the positions of the lower left and the upper right corners of the region representing the i-th area.

Output

Output a single integer — the minimum cost to construct all cables in the unit of won, modulo 998 244 353. $998\,244\,353 = 119 \times 2^{23} + 1$ is a prime number.

Constraints

- $2 \le N \le 300\,000$
- $0 \le x_1^i < x_2^i \le 998\,244\,352$ $(1 \le i \le N)$
- $0 \le y_1^i < y_2^i \le 998\,244\,352$ $(1 \le i \le N)$

Examples

standard input	standard output
3 1 7 2 9 3 2 8 4 4 3 8 5	8
4 0 1 2 3 1 0 3 2 3 4 5 6 4 3 6 5	8

Problem C. Double-Colored Papers

Time limit: 3 seconds

In your factory, you are making two kinds of colored paper, one colored red, and the other colored blue.

Each red-colored paper has a string S written on it: it is made of $|S|$ unit squares in a row, and S_i is written on the ith square from the left.

Each blue-colored paper has a string T written on it: it is made of $|T|$ unit squares in a row, and T_i is written on the ith square from the left.

You plan to make a new kind of paper called *double-colored paper* out of red and blue paper. To do so, you will cut down a piece of red paper to only leave the continuous part with a positive integer length, and again with blue paper. Then, you will glue the ending part of the red paper to the starting part of the blue paper.

For example, suppose S is `abcde` and T is `fghij`. You can make a *double-colored paper* with string `bcdfg` or `abcij` written on it. However, you cannot make a *double-colored paper* with string `acdghij` or `fghij` written on it. (Here the underlined string denotes a part of the red paper, and the rest denotes a part of the blue paper.)

Among all possible *double-colored papers* that can be made, you want to know the one with the lexicographically K-th smallest string written on it. Note that there may be papers with the same strings written on them, but with different lengths of red paper: in this case, you may order them arbitrarily.

Input

The first line contains the string S.

The second line contains the string T.

The third line contains the integer K.

Output

If the total number of possible *double-colored paper*s is strictly less than K, output -1.

Otherwise, output the lexicographically K-th smallest string of all possible *double-colored paper*s that can be made.

Constraints

- $1 \leq |S| \leq 75\,000$
- $1 \leq |T| \leq 75\,000$
- S and T consists of lowercase alphabets.
- $1 \leq K \leq 8 \times 10^{18}$

Examples

standard input	standard output
tww wtw 21	wwtw

Note

All possible *double-colored papers* that can be made are `tt`, `ttw`, `tw`, `twt`, `twt`, `twtw`, `twtw`, `tww`, `twwt`, `twwt`, `twwtw`, `twwtw`, `twww`, `twwwt`, `twwwtw`, `wt`, `wtw`, `ww`, `wwt`, `wwt`, `wwtw`, `wwtw`, `www`, `wwwt`, `wwwtw`, in this order.

Problem D. Histogram Sequence 4

Time limit: 1 second

You had a histogram made of N axis-parallel rectangles sharing a common baseline: the i-th rectangle from the left had a width 1 and an integer height H_i.

Sadly, you lost your histogram! Moreover, you even forgot what your histogram looked like — the heights of the rectangles of the histogram. What you remember is the maximum area A of the axis-parallel rectangle inside the histogram, and the fact that $L \leq H_i \leq R$ for every H_i.

Your goal is to recover the histogram by finding any histogram satisfying all the requirements that you remember. As your memory might not be perfect, there may be no histogram satisfying the requirements.

Input

The first and only line contains four space-separated integers, N, A, L, R.

Output

If there is no histogram satisfying the requirements, output `NO`.

Otherwise, output `YES` in the first line. In the second line, output N integers in a single line, where the i-th value is the height H_i of the i-th rectangle. If there are multiple answers, print any.

Constraints

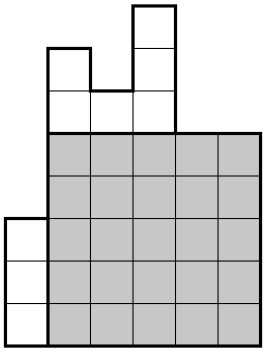
- $1 \leq N \leq 500\,000$
- $0 \leq A \leq 10^{18}$
- $0 \leq L \leq R \leq 10^{18}$

Examples

standard input	standard output
6 25 2 10	YES 3 7 6 8 5 5
1 0 0 1000000000000000000	YES 0
1 8213912883 0 28318	NO

Note

A histogram with heights $3, 7, 6, 8, 5, 5$ is shown below. The maximum area rectangle inside the histogram has an area of 25.



Problem E. Making Number

Time limit: 1.5 seconds

You are given two positive integers X and Y of the same length in base 10. Z is defined as the positive integer in base 10 satisfying the following conditions.

- The digits of Z should be a rearrangement of the digits of X. Leading zeros in Z are not allowed. For example, if $X = 1103$, Z can be 1103 or 3101, but Z cannot be 2110, 301, nor 0131.
- $Y \leq Z$.
- Z is the minimum value satisfying the above conditions.

You have to perform Q queries. Each query is one of the followings:

- Given i and x, change the i-th digit of Y to x.
- Given i, output the i-th digit of Z. If there is no such Z, print -1.

The i-th digit of a positive integer is defined from the left. For example, The third digit of 1234 is 3.

Input

The first line contains two space-separated integers, X and Y.

The second line contains a single integer, Q.

The following Q lines contain space-separated integers describing the queries. Each line has one of the following forms, where the first integer represents the type of the query:

- `1` i x : Change the i-th digit of Y to x.
- `2` i : Output the i-th digit of Z. If there is no such Z, print -1.

It is guaranteed that there is at least one query of type 2.

Output

For each query of type 2, output the answer for the query. The answers should be separated by newlines.

Constraints

Let $\texttt{len}(A)$ be the number of digits in a positive integer A.

- $1 \leq X, Y < 10^{100\,000}$
- $1 \leq Q \leq 100\,000$
- $\texttt{len}(X) = \texttt{len}(Y)$
- The first digits of X and Y are not 0.
- For a query of type 1, $1 \leq i \leq \texttt{len}(Y)$, $0 \leq x \leq 9$. If $i = 1$, $x \neq 0$.
- For a query of type 2, $1 \leq i \leq \texttt{len}(Y)$.

Examples

standard input	standard output
3304 1615 6 2 3 2 4 1 1 3 2 2 1 2 4 2 1	3 4 0 3
838046 780357 10 2 1 2 2 1 2 4 2 3 2 4 1 4 5 2 5 2 6 1 1 9 2 2	8 0 3 4 6 8 -1
2950 9052 4 2 1 2 2 2 3 2 4	9 0 5 2

Problem F. One Path

Time limit: 1 second

You are given a tree T consisting of N vertices. Each edge has a positive integer weight.

You can perform the following operation on the given tree.

- Delete an edge from the graph, then add a new edge between any two distinct vertices. The weight of the new edge must be the same as the weight of the deleted edge. The resulting graph need not be a tree.

We define the weight of a path as the sum of the weights of the edges on the path. The distance between two vertices u and v is defined as the weight of the *shortest path* from u to v — having the minimum weight. If there is no such path, we define the distance as 0.

The weight of a graph is the maximum of the weights between any two vertices.

Your task is to find the largest weight of the graph that can be obtained by performing the operation exactly i times, for $i = 0, 1, \ldots, K$.

Input

The first line contains two space-separated integers, N and K.

The i-th of the following $N-1$ lines contains three space-separated integers u_i, v_i, and w_i — representing an undirected edge that connects two different vertices u_i and v_i with a weight of w_i.

It is guaranteed that the edges form a tree.

Output

Output $K+1$F space-separated integers. The i-th integer should be equal to the largest weight of the graph that can be obtained by performing the operation exactly $i-1$ times.

Constraints

- $2 \leq N \leq 2\,000$
- $0 \leq K \leq 2\,000$
- $1 \leq u_i < v_i \leq N$ $(1 \leq i \leq N-1)$
- $1 \leq w_i \leq 10^9$ $(1 \leq i \leq N-1)$

Examples

standard input	standard output
5 1 1 3 2 4 5 4 3 4 3 2 3 7	14 16
7 2 1 2 4 2 3 6 2 4 2 4 5 5 2 6 1 4 7 3	13 20 21

Problem G. Permutation Arrangement

Time limit: 1 second

You are given an array a of length N. Each element of a is either -1 or an integer between 1 and N. Each number between 1 and N appears at most once in a. Also, no two adjacent elements of a differ by exactly 1.

You are to find the lexicographically smallest permutation p of $\{1, 2, \cdots, N\}$ satisfying the following.

- If $a_i \neq -1$, then $a_i = p_i$ $(1 \leq i \leq N)$
- $|p_i - p_{i+1}| \neq 1$ $(1 \leq i \leq N-1)$

Input

The first line contains one integer, N.

The second line contains space-separated N integers — elements of the array a.

Output

If there is no permutation p satisfying the condition, then output a single integer -1.

Otherwise, output the lexicographically smallest permutation p.

Constraints

- $1 \leq N \leq 200,000$
- $1 \leq a_i \leq N$ or $a_i = -1$ $(1 \leq i \leq N)$
- $a_i \neq a_j$ or $a_i = -1$ $(1 \leq i < j \leq N)$
- $|a_i - a_{i+1}| \neq 1$ $(1 \leq i \leq N-1)$

Examples

standard input	standard output
10 3 -1 10 -1 8 -1 -1 -1 -1 -1	3 1 10 2 8 4 6 9 5 7
2 -1 -1	-1

Problem H. Set and Sequence and Query

Time limit: 2 seconds

Takina and Chisato are playing a game with a set of positive integers.

This game is about making *continuous increasing sequence*s using the numbers from the set.

A *continuous increasing sequence* is defined as a sequence $a_1, a_2, \ldots, a_k$ of positive length k, satisfying $a_{i+1} = a_i + 1$ for all $1 \leq i \leq k - 1$.

The game begins with an empty set and consists of Q turns. In each turn, Takina can either insert a new integer into the set or delete an integer from the set.

Every time a change is made to the set, Chisato has to count how many different *continuous increasing sequence*s can be made using the numbers from the set.

Your task is to help Chisato.

Input

The first line contains the number of turns, Q.

The following Q lines contain two integers, describing Takina's move. Each line has one of the following forms:

- `1` x : Insert x into the set. It is guaranteed that x was not in the set.
- `2` x : Delete x from the set. It is guaranteed that x was in the set.

Output

Output Q integers separated by newlines, the number of *continuous increasing sequence*s in the set after each Takina's move.

Constraints

- $1 \leq Q \leq 300\,000$
- $1 \leq x \leq 10^9$

Examples

standard input	standard output
3 1 1 1 2 2 1	1 3 1

Problem I. Similarity Graph

Time limit: 1 second

Let p and q be two permutations of $\{1, 2, \ldots, N\}$.

Similarity graph of p and q, $S(p, q)$, is defined as following:

- $S(p, q)$ has N labeled vertices, numbered from 1 to N.
- There is a edge between vertex i and j if and only if $p_i < p_j$ and $q_i < q_j$ are both true, or both false. $(1 \leq i < j \leq N)$

You are given a simple undirected graph G with N labeled vertices, numbered from 1 to N.

Find a pair (p, q) of permutations of $\{1, 2, \ldots, N\}$, satisfying $S(p, q) = G$.

Input

The first line contains one integer, N.

The next N lines contain space-separated N integers. The j-th integer of the i-th line is $E(i, j)$. $E(i, j)$ is 1 if there is an edge between vertices i and j, 0 otherwise.

Output

If it is impossible to find p and q satisfying the condition, output `NO`.

Otherwise, output `YES` on the first line. On the following two lines, output p and q. If there are multiple answers, output any.

Constraints

- $1 \leq N \leq 100$
- $0 \leq E(i, j) \leq 1$ $(1 \leq i, j \leq N)$
- $E(i, j) = E(j, i)$ $(1 \leq i < j \leq N)$
- $E(i, i) = 0$ $(1 \leq i \leq N)$

Examples

standard input	standard output
4 0 1 0 1 1 0 0 0 0 0 0 1 1 0 1 0	YES 1 2 3 4 2 4 1 3
6 0 1 0 1 0 1 1 0 0 0 1 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0	NO

Problem J. Squirrel Game

Time limit: 3 seconds

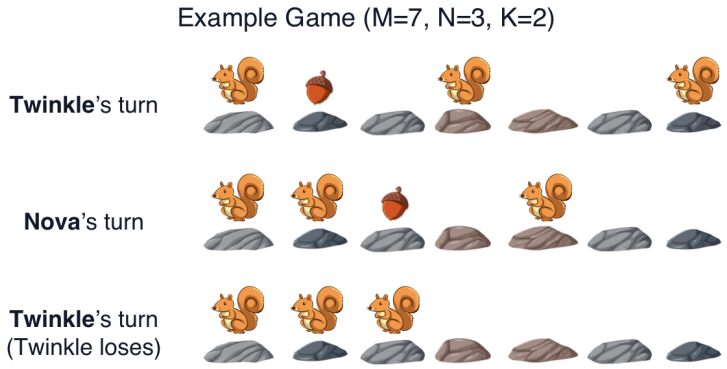
Twinkle and Nova are walking in a national park. There are M stones laid out in the park at positions $1, \cdots, M$, from left to right. There are also N squirrels on the stones at $x_1, \cdots, x_N$, from left to right. The squirrels are on different stones from each other, and they are all facing left.

Twinkle suggests the following game to Nova. Twinkle and Nova take turns alternately. On each turn, a player has to place an acorn on one of the stones without a squirrel. Also, there must be at least one squirrel to the right of the acorn.

After placing an acorn, the leftmost K squirrels among the squirrels to the right of the acorn start running towards the acorn at the same time. (If there are less than K squirrels to the right of the acorn, all of them start running.) All the squirrels run at the same speed. Once any of the squirrels reach the acorn, all the squirrels immediately stop. The squirrel who has reached the acorn puts the acorn into its cheek pouch, effectively removing the acorn on the stone.

If there is no valid stone to place an acorn on, the player currently taking the turn immediately loses.

Twinkle goes first. Determine who will win if both players are playing optimally.



Input

The first line contains three space-separated integers, M, N, and K.

The second line contains N space-separated integers $x_1, \cdots, x_N$.

Output

If Twinkle wins, output `Twinkle`. Otherwise, output `Nova`.

Constraints

- $1 \leq N \leq M \leq 100\,000$
- $1 \leq K \leq 10$
- $1 \leq x_1 < \cdots < x_N \leq M$

Examples

standard input	standard output
7 3 2 1 4 7	Nova
7 3 1 1 4 7	Twinkle

Problem K. Two Paths

Time limit: 5 seconds

You are given a tree T consisting of N vertices. Each edge has a positive integer weight. The weight of a path P in T is defined as the sum of weights of edges in P, denoted by $W(P)$.

You are given a total of Q queries, each containing two vertices u, v, and two integers A and B. For each query, you are to find two simple paths P_1 and P_2 in T satisfying these requirements.

- P_1 and P_2 doesn't share a vertex.
- P_1 starts from u, and P_2 starts from v.
- Among all P_1 and P_2 satisfying the conditions above, the value of $A \times W(P_1) + B \times W(P_2)$ should be maximized.

You should output the value of $A \times W(P_1) + B \times W(P_2)$ for each query.

Input

The first line contains two space-separated integers N and Q.

Each of the following $N - 1$ lines contains three space-separated integers u, v, w. This means that there is an edge in T, connecting vertices u and v with weight w.

Each of the following Q lines contains four space-separated integers u, v, A, B, denoting a single query.

Output

For each query, output the maximum possible value of $A \times W(P_1) + B \times W(P_2)$. The answers should be separated by newlines.

Constraints

- $2 \leq N \leq 200\,000$
- $1 \leq Q \leq 500\,000$
- $1 \leq u < v \leq N$ for both edges and queries
- $1 \leq w \leq 10\,000$
- $1 \leq A, B \leq 2 \times 10^9$

Examples

standard input	standard output
6 4 1 2 4 2 5 5 2 3 7 3 6 5 3 4 4 1 4 1 1 1 4 2 1 5 6 1 1 5 6 1 10	18 32 18 160

Problem L. Village Planning

Time limit: 3 seconds

As the mayor of the RUN town, you are planning to build a new village. The village consists of houses and roads connecting two different houses. Roads are organized in a way such that no two pairs of roads connect the same pair of houses. In other words, the village can be treated as a simple graph where each house corresponds to vertices, and each road corresponds to edges. Note that the village may be disconnected.

You want your village to be as simple as possible. Therefore, for any distinct houses i and j, there should be at most K simple paths from house i to house j.

Let N be the number of houses. The score of the village is $\prod_{1 \le i < j \le N} A_{f(i,j)}$, where $f(i, j)$ is the number of simple paths from house i to house j.

While the number of houses is not determined yet, you know that it will be an integer between 2 and M. You should calculate the sum of the scores for all possible villages with N houses for each N from 2 to M.

Since the answers can be large, output them modulo 998 244 353.

Input

The first line contains an two space-separated integers M and K.

The second line contains $K + 1$ space-separated integers $A_0, \ldots, A_K$.

Output

For each N from 2 to M, output the sum of the scores for all possible villages with N houses, modulo 998 244 353. The answers should be separated by a space. $998\,244\,353 = 119 \times 2^{23} + 1$ is a prime number.

Constraints

- $2 \le M \le 100\,000$
- $0 \le K \le 3$
- $1 \le A_i < 998\,244\,353$ $(0 \le i \le K)$

Examples

standard input	standard output
4 0 2	2 8 64
5 1 3 4	7 327 96721 169832849
6 2 5 6 7	11 1566 3000672 306031599 466869291
7 3 8 9 10 11	17 5427 31856976 326774674 449014006 997476587

Problem M. Window Arrangement

Time limit: 3 seconds

KAIST is running out of budget — they need some money! They thought the dormitories were way too luxurious compared to the other buildings of KAIST; they planned to sell all the dormitory buildings and build a new completely non-aesthetic one.

The new dormitory will be of a grid-shape — it can't be more boring than this — of size $N \times M$, each cell being the room for the students. We are going to add some windows, because we want students to get some sunlight during the daytime!

We plan to have exactly $w_{i,j}$ windows for the room (i, j). A window can be built on the side of an edge of a grid, and at most one window can be built on each side of an edge. A window is one-sided: a window on the opposite side of an edge does not count as a window of the room.

Unfortunately, students will experience huge discomfort when their privacy is watched by someone else through the window. **Total discomfort** is the number of set of students $\{a, b\}$, such that a and b can see each other's privacy through the window.

Precisely, if an edge has windows on both sides, total discomfort increases by the product of the number of people living in two houses sharing the window.

You're given $w_{i,j}$, and $p_{i,j}$, the number of people living in the room (i, j). Your task is to find the minimum total discomfort that can be achieved by arranging the windows properly.

Input

The first line contains the size of the grid, N and M.

Following N lines contains M space-separated integers $p_{i,j}$.

Following N lines contains M space-separated integers $w_{i,j}$.

Output

Output the minimum total discomfort.

Constraints

- $1 \leq N \leq 50$
- $1 \leq M \leq 50$
- $1 \leq p_{i,j} \leq 1\,000$ $(1 \leq i \leq N, 1 \leq j \leq M)$
- $0 \leq w_{i,j} \leq 4$ $(1 \leq i \leq N, 1 \leq j \leq M)$

Examples

standard input	standard output
4 3 1 7 10 7 2 8 7 9 10 4 6 4 3 3 3 3 2 4 4 3 4 2 2 3	178
4 3 2 2 9 9 8 4 8 4 5 7 5 2 0 1 0 1 0 1 0 0 1 0 1 0	0