A. Catch the Coin

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

Monocarp visited a retro arcade club with arcade cabinets. There got curious about the "Catch the Coin" cabinet.

The game is pretty simple. The screen represents a coordinate grid such that:

- the X-axis is directed from left to right;
- the Y-axis is directed from bottom to top;
- the center of the screen has coordinates $(0, 0)$.

At the beginning of the game, the character is located in the center, and n coins appear on the screen — the i-th coin is at coordinates (x_i, y_i). The coordinates of all coins are different and not equal to $(0, 0)$.

In one second, Monocarp can move the character in one of eight directions. If the character is at coordinates (x, y), then it can end up at any of the coordinates $(x, y + 1)$, $(x + 1, y + 1)$, $(x + 1, y)$, $(x + 1, y - 1)$, $(x, y - 1)$, $(x - 1, y - 1)$, $(x - 1, y)$, $(x - 1, y + 1)$.

If the character ends up at the coordinates with a coin, then Monocarp collects that coin.

After Monocarp makes a move, all coins fall down by 1, that is, they move from (x, y) to $(x, y - 1)$. You can assume that the game field is infinite in all directions.

Monocarp wants to collect at least one coin, but cannot decide which coin to go for. Help him determine, for each coin, whether he can collect it.

Input

The first line contains a single integer n ($1 \le n \le 500$) — the number of coins.

In the i-th of the next n lines, two integers x_i and y_i ($-50 \le x_i, y_i \le 50$) are written — the coordinates of the i-th coin. The coordinates of all coins are different. No coin is located at $(0, 0)$.

Output

For each coin, print "YES" if Monocarp can collect it. Otherwise, print "NO".

Standard Input	Standard Output
5 24 42 -2 -1 -1 -2 0 -50 15 0	YES YES NO NO YES

Note

Pay attention to the second coin in the example. Monocarp can first move from $(0, 0)$ to $(-1, -1)$. Then the coin falls 1 down and ends up at $(-2, -2)$. Finally, Monocarp moves to $(-2, -2)$ and collects the coin.

,

B. Substring and Subsequence

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given two strings a and b, both consisting of lowercase Latin letters.

A *subsequence* of a string is a string which can be obtained by removing several (possibly zero) characters from the original string. A *substring* of a string is a contiguous subsequence of that string.

For example, consider the string abac:

- a, b, c, ab, aa, ac, ba, bc, aba, abc, aac, bac and abac are its subsequences;
- a, b, c, ab, ba, ac, aba, bac and abac are its substrings.

Your task is to calculate the minimum possible length of the string that contains a as a substring and b as a subsequence.

Input

The first line contains a single integer t ($1 \le t \le 10^3$) — the number of test cases.

The first line of each test case contains a string a ($1 \le |a| \le 100$), consisting of lowercase Latin letters.

The second line of each test case contains a string b ($1 \le |b| \le 100$), consisting of lowercase Latin letters.

Output

For each test case, print a single integer — the minimum possible length of the string that contains a as a substring and b as a subsequence.

Standard Input	Standard Output
5 aba cb er cf mmm mmm contest test cde abcefg	4 4 3 7 7

Note

In the examples below, the characters that correspond to the subsequence equal to b are bolded.

In the first example, one of the possible answers is **c**a**b**a.

In the second example, one of the possible answers is e**r****cf**.

In the third example, one of the possible answers is **mmm**.

In the fourth example, one of the possible answers is `con`**`test`**.

In the fifth example, one of the possible answers is **`abc`**`d`**`efg`**.

,

C. Two Movies

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

A movie company has released 2 movies. These 2 movies were watched by n people. For each person, we know their attitude towards the first movie (liked it, neutral, or disliked it) and towards the second movie.

If a person is asked to leave a review for the movie, then:

- if that person liked the movie, they will leave a positive review, and the movie's rating will increase by 1;
- if that person disliked the movie, they will leave a negative review, and the movie's rating will decrease by 1;
- otherwise, they will leave a neutral review, and the movie's rating will not change.

Every person will review exactly one movie — and for every person, you can choose which movie they will review.

The company's rating is the minimum of the ratings of the two movies. Your task is to calculate the maximum possible rating of the company.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-1 \le a_i \le 1$), where a_i is equal to -1 if the first movie was disliked by the i-th viewer; equal to 1 if the first movie was liked; and 0 if the attitude is neutral.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($-1 \le b_i \le 1$), where b_i is equal to -1 if the second movie was disliked by the i-th viewer; equal to 1 if the second movie was liked; and 0 if the attitude is neutral.

Additional constraint on the input: the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print a single integer — the maximum possible rating of the company, if for each person, choose which movie to leave a review on.

Standard Input	Standard Output
4 2 -1 1 -1 -1 1 -1 -1 5 0 -1 1 0 1 -1 1 0 0 1 4	0 -1 1 1

-1 -1 -1 1 -1 1 1 1	

,

D. Smithing Skill

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 256 megabytes

You are playing a famous computer game (that just works) where you have various skills you can level up. Today, you focused on the "Smithing" skill. Your tactic is obvious: forging weapons from ingots and then melting them back to return the materials partially. For simplicity, every time you create an item, you get 1 experience point, and every time you melt an item, you also get 1 experience point.

There are n classes of weapons you can forge and m types of metal ingots.

You can create one weapon of the i-th class, spending a_i ingots of metal of *the same type*. Melting a weapon of the i-th class (which you crafted earlier) returns you b_i ingots of the type of metal *it was made of*.

You have c_j metal ingots of the j-th type, and you know that you can craft a weapon of any class from any metal type. Each combination of a weapon class and a metal type can be used any number of times.

What is the maximum total amount of experience you can earn by crafting and melting weapons?

Input

The first line contains two integers n and m ($1 \le n, m \le 10^6$) — the number of weapon classes and metal types.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$), where a_i is the number of ingots you need to forge one weapon of the i-th class.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i < a_i$), where b_i is the number of ingots you return by melting one weapon of the i-th class you forged earlier.

The fourth line contains m integers $c_1, c_2, \ldots, c_m$ ($1 \le c_j \le 10^9$) — the number of ingots you have of the corresponding metal type.

Output

Print one integer — the maximum total experience points you can gain by repeatedly forging and melting weapons.

Standard Input	Standard Output
5 3 9 6 7 5 5 8 4 5 1 2 10 4 7	12
3 4 10 20 20 0 0 0 9 10 19 20	8
1 5 3	4999999990

1 1000000000 1000000000 1000000000 1000000000 1000000000	

Note

In the first example, you can do the following:

1. craft one weapon of the 1-st class from the 1-st type of metal, spending 9 ingots;
2. melt that weapon, returning 8 ingots of the 1-st metal type;
3. again, craft and melt one weapon of the 1-st class from the 1-st metal type;
4. craft and melt one weapon of the 3-rd class from the 1-st metal type;
5. craft and melt one weapon of the 3-rd class from the 3-rd metal type;
6. craft and melt one weapon of the 4-th class from the 1-st metal type;
7. craft and melt one weapon of the 5-th class from the 3-rd metal type;

In the end you'll have $c = [2, 4, 2]$ ingots left. In total, you've crafted 6 weapons and melted 6 weapons, gaining 12 experience points in total.

,

E. Distance to Different

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

Consider an array a of n integers, where every element is from 1 to k, and every integer from 1 to k appears **at least once**.

Let the array b be constructed as follows: for the i-th element of a, b_i is the distance to the closest element in a which is not equal to a_i. In other words, $b_i = \min\limits_{j \in [1,n], a_j \neq a_i} |i - j|$.

For example, if $a = [1, 1, 2, 3, 3, 3, 3, 1]$, then $b = [2, 1, 1, 1, 2, 2, 1, 1]$.

Calculate the number of different arrays b you can obtain if you consider all possible arrays a, and print it modulo 998244353.

Input

The only line of the input contains two integers n and k ($2 \le n \le 2 \cdot 10^5$; $2 \le k \le \min(n, 10)$).

Output

Print one integer — the number of different arrays b you can obtain, taken modulo 998244353.

Standard Input	Standard Output
2 2	1
4 3	3
6 2	20
6 5	3
133 7	336975971

,

F. Simultaneous Coloring

```
Input file:     standard input
Output file:    standard output
Time limit:     6 seconds
Memory limit:   512 megabytes
```

You are given a matrix, consisting of n rows and m columns.

You can perform two types of actions on it:

- paint the entire column in blue;
- paint the entire row in red.

Note that you cannot choose which color to paint the row or column.

In one second, you can perform either one action or multiple actions at the same time. If you perform one action, it will be free. If you perform $k > 1$ actions at the same time, it will cost k^2 coins. When multiple actions are performed at the same time, for each cell affected by actions of both types, the color can be chosen independently.

You are asked to process q queries. Before each query, all cells become colorless. Initially, there are no restrictions on the color of any cells. In the i-th query, a restriction of the following form is added:

- x_i y_i c_i — the cell in row x_i in column y_i should be painted in color c_i.

Thus, after i queries, there are i restrictions on the required colors of the matrix cells. After each query, output the minimum cost of painting the matrix according to the restrictions.

Input

The first line contains three integers n, m and q ($1 \le n, m, q \le 2 \cdot 10^5$) — the size of the matrix and the number of queries.

In the i-th of the next q lines, two integers x_i, y_i and a character c_i ($1 \le x_i \le n$; $1 \le y_i \le m$; $c_i \in$ {'R', 'B'}, where 'R' means red, and 'B' means blue) — description of the i-th restriction. The cells in all queries are pairwise distinct.

Output

Print q integers — after each query, output the minimum cost of painting the matrix according to the restrictions.

Standard Input	Standard Output
2 2 4 1 1 R 2 2 R 1 2 B 2 1 B	0 0 0 16
3 5 10 1 1 B 2 5 B 2 2 B	0 0 0 0

2 3 R 2 1 B 3 2 R 3 3 B 1 2 R 1 3 B 3 1 B	0 0 16 16 25 25