A. Nezzar and Board

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

n **distinct** integers $x_1, x_2, \ldots, x_n$ are written on the board. Nezzar can perform the following operation multiple times.

- Select two integers x, y (not necessarily distinct) on the board, and write down $2x - y$. Note that you don't remove selected numbers.

Now, Nezzar wonders if it is possible to have his favorite number k on the board after applying above operation multiple times.

Input

The first line contains a single integer t ($1 \le t \le 10^5$) — the number of test cases.

The first line of each test case contains two integers n, k ($2 \le n \le 2 \cdot 10^5$, $-10^{18} \le k \le 10^{18}$).

The second line of each test case contains n distinct integers $x_1, x_2, \ldots, x_n$ ($-10^{18} \le x_i \le 10^{18}$).

It is guaranteed that the sum of n for all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print "YES" on a single line if it is possible to have k on the board. Otherwise, print "NO".

You can print each letter in any case (upper or lower).

Example

input

```
6
2 1
1 2
3 0
2 3 7
2 -1
31415926 27182818
2 1000000000000000000
1 1000000000000000000
2 -1000000000000000000
-1000000000000000000 123
6 80
-5 -20 13 -14 -2 -11
```

output

```
YES
YES
NO
YES
YES
NO
```

Note

In the first test case, the number 1 is already on the board.

In the second test case, Nezzar could perform the following operations to write down $k = 0$ on the board:

- Select $x = 3$ and $y = 2$ and write down 4 on the board.
- Select $x = 4$ and $y = 7$ and write down 1 on the board.
- Select $x = 1$ and $y = 2$ and write down 0 on the board.

In the third test case, it is impossible to have the number $k = -1$ on the board.

B. Nezzar and Binary String

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Nezzar has a binary string s of length n that he wants to share with his best friend, Nanako. Nanako will spend q days inspecting the binary string. At the same time, Nezzar wants to change the string s into string f during these q days, because it looks better.

It is known that Nanako loves consistency so much. On the i-th day, Nanako will inspect a segment of string s from position l_i to position r_i inclusive. If the segment contains both characters '0' and '1', Nanako becomes unhappy and throws away the string.

After this inspection, at the i-th night, Nezzar can secretly change **strictly less** than half of the characters in the segment from l_i to r_i inclusive, otherwise the change will be too obvious.

Now Nezzar wonders, if it is possible to avoid Nanako being unhappy and at the same time have the string become equal to the string f at the end of these q days and nights.

Input

The first line contains a single integer t ($1 \le t \le 2 \cdot 10^5$) — the number of test cases.

The first line of each test case contains two integers n, q ($1 \le n \le 2 \cdot 10^5$, $0 \le q \le 2 \cdot 10^5$).

The second line of each test case contains a binary string s of length n.

The third line of each test case contains a binary string f of length n.

Then q lines follow, i-th of them contains two integers l_i, r_i ($1 \le l_i \le r_i \le n$) — bounds of the segment, that Nanako will inspect on the i-th day.

It is guaranteed that the sum of n for all test cases doesn't exceed $2 \cdot 10^5$, and the sum of q for all test cases doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print "YES" on the single line if it is possible to avoid Nanako being unhappy and have the string f at the end of q days and nights. Otherwise, print "NO".

You can print each letter in any case (upper or lower).

Example

input

```
4
5 2
00000
00111
1 5
1 3
2 1
00
01
1 2
10 6
1111111111
0110001110
1 10
5 9
7 10
1 7
3 5
```

```
6 10
5 2
10000
11000
2 5
1 3
```

output

```
YES
NO
YES
NO
```

Note

In the first test case, $\underline{00000} \to \underline{000}11 \to 00111$ is one of the possible sequences of string changes.

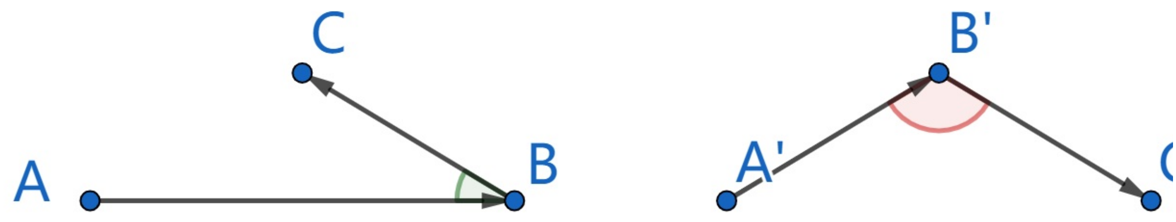
In the second test case, it can be shown that it is impossible to have the string f at the end.

C. Nezzar and Nice Beatmap

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Nezzar loves the game osu!.

osu! is played on beatmaps, which can be seen as an array consisting of **distinct** points on a plane. A beatmap is called nice if for any three consecutive points A, B, C listed in order, the angle between these three points, centered at B, is **strictly less than** 90 degrees.



Points A, B, C on the left have angle less than 90 degrees, so they can be three consecutive points of a nice beatmap; Points A', B', C' on the right have angle greater or equal to 90 degrees, so they cannot be three consecutive points of a nice beatmap.

Now Nezzar has a beatmap of n **distinct** points $A_1, A_2, \ldots, A_n$. Nezzar would like to reorder these n points so that the resulting beatmap is nice.

Formally, you are required to find a permutation $p_1, p_2, \ldots, p_n$ of integers from 1 to n, such that beatmap $A_{p_1}, A_{p_2}, \ldots, A_{p_n}$ is nice. If it is impossible, you should determine it.

Input

The first line contains a single integer n ($3 \le n \le 5000$).

Then n lines follow, i-th of them contains two integers x_i, y_i ($-10^9 \le x_i, y_i \le 10^9$) — coordinates of point A_i.

It is guaranteed that all points are distinct.

Output

If there is no solution, print -1.

Otherwise, print n integers, representing a valid permutation p.

If there are multiple possible answers, you can print any.

Example

input

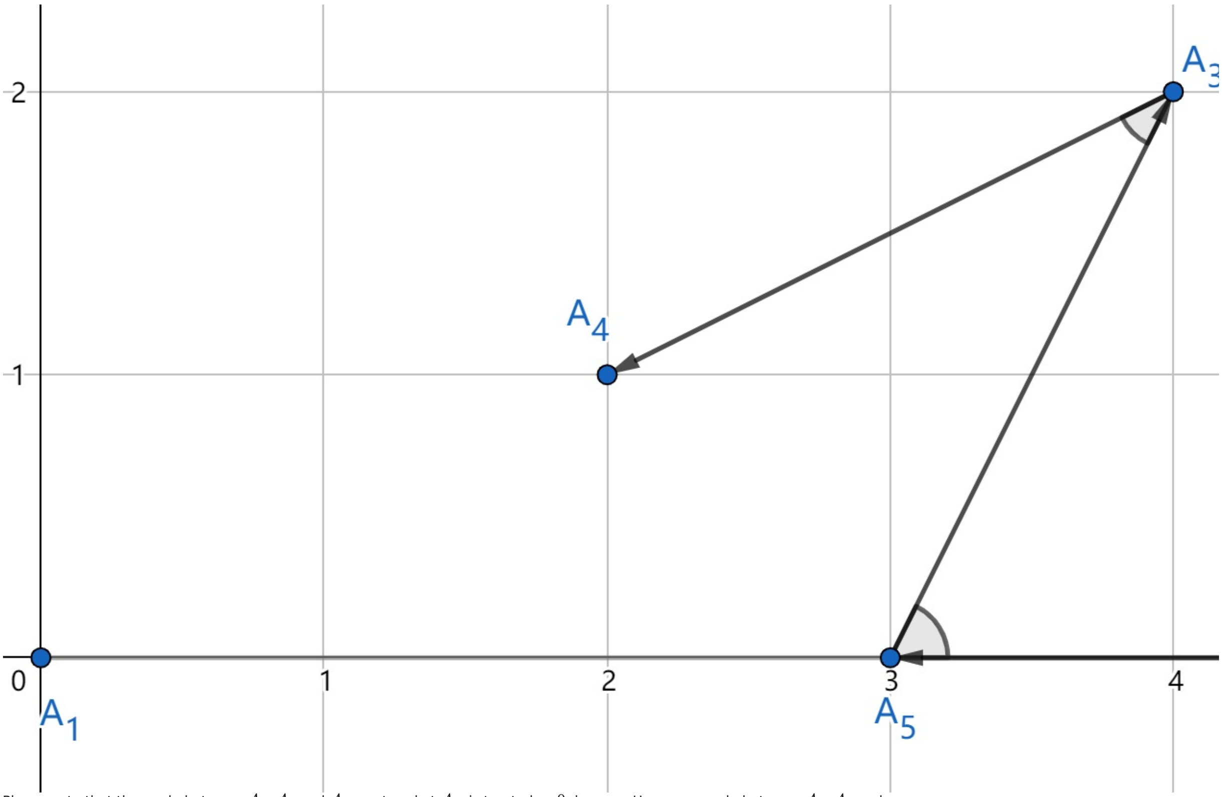
```
5
0 0
5 0
4 2
2 1
3 0
```

output

```
1 2 5 3 4
```

Note

Here is the illustration for the first test:



Please note that the angle between A_1, A_2 and A_5, centered at A_2, is treated as 0 degrees. However, angle between A_1, A_5 and A_2, centered at A_5, is treated as 180 degrees.

D. Nezzar and Hidden Permutations

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Nezzar designs a brand new game "Hidden Permutations" and shares it with his best friend, Nanako.

At the beginning of the game, Nanako and Nezzar both know integers n and m. The game goes in the following way:

- Firstly, Nezzar hides two permutations $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots, q_n$ of integers from 1 to n, and Nanako secretly selects m unordered pairs $(l_1, r_1), (l_2, r_2), \ldots, (l_m, r_m)$;
- After that, Nanako sends his chosen pairs to Nezzar;
- On receiving those m unordered pairs, Nezzar checks if there exists $1 \le i \le m$, such that $(p_{l_i} - p_{r_i})$ and $(q_{l_i} - q_{r_i})$ have different signs. If so, Nezzar instantly loses the game and gets a score of -1. Otherwise, the score Nezzar gets is equal to the number of indices $1 \le i \le n$ such that $p_i \ne q_i$.

However, Nezzar accidentally knows Nanako's unordered pairs and decides to take advantage of them. Please help Nezzar find out two permutations p and q such that the score is maximized.

Input

The first line contains a single integer t ($1 \le t \le 5 \cdot 10^5$) — the number of test cases.

The first line of each test case contains two integers n, m ($1 \le n \le 5 \cdot 10^5, 0 \le m \le \min(\frac{n(n-1)}{2}, 5 \cdot 10^5)$).

Then m lines follow, i-th of them contains two integers l_i, r_i ($1 \le l_i, r_i \le n$, $l_i \ne r_i$), describing the i-th unordered pair Nanako chooses. It is guaranteed that all m unordered pairs are distinct.

It is guaranteed that the sum of n for all test cases does not exceed $5 \cdot 10^5$, and the sum of m for all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, print two permutations $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots, q_n$ such that the score Nezzar gets is maximized.

Example

input

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

output

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

Note

For first test case, for each pair given by Nanako:

- for the first pair $(1, 2)$: $p_1 - p_2 = 1 - 2 = -1$, $q_1 - q_2 = 3 - 4 = -1$, they have the same sign;
- for the second pair $(3, 4)$: $p_3 - p_4 = 3 - 4 = -1$, $q_3 - q_4 = 1 - 2 = -1$, they have the same sign.

As Nezzar does not lose instantly, Nezzar gains the score of 4 as $p_i \ne q_i$ for all $1 \le i \le 4$. Obviously, it is the maximum possible score Nezzar can get.

E. Nezzar and Tournaments

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In the famous Oh-Suit-United tournament, two teams are playing against each other for the grand prize of precious pepper points.

The first team consists of n players, and the second team consists of m players. Each player has a potential: the potential of the i-th player in the first team is a_i, and the potential of the i-th player in the second team is b_i.

In the tournament, **all** players will be on the stage in some order. There will be a scoring device, initially assigned to an integer k, which will be used to value the performance of all players.

The scores for all players will be assigned in the order they appear on the stage. Let the potential of the current player be x, and the potential of the previous player be y (y **equals** x **for the first player**). Then, $x - y$ is added to the value in the scoring device, Afterwards, if the value in the scoring device becomes negative, **the value will be reset to** 0. Lastly, the player's score is assigned to the current value on the scoring device. The score of a team is the sum of the scores of all its members.

As an insane fan of the first team, Nezzar desperately wants the biggest win for the first team. He now wonders what is the maximum difference between scores of the first team and the second team.

Formally, let the score of the first team be $score_f$ and the score of the second team be $score_s$. Nezzar wants to find the maximum value of $score_f - score_s$ over all possible orders of players on the stage.

However, situation often changes and there are q events that will happen. There are three types of events:

- 1 pos x — change a_{pos} to x;
- 2 pos x — change b_{pos} to x;
- 3 x — tournament is held with $k = x$ and Nezzar wants you to compute the maximum value of $score_f - score_s$.

Can you help Nezzar to answer the queries of the third type?

Input

The first line contains three integers n, m, and q ($1 \le n, m \le 2 \cdot 10^5, 1 \le q \le 5 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^6$).

The third line contains m integers $b_1, b_2, \ldots, b_m$ ($0 \le b_i \le 10^6$).

The following q lines contain descriptions of events, described in the statement, each in one of the three possible formats:

- 1 pos x ($1 \le pos \le n$, $0 \le x \le 10^6$);
- 2 pos x ($1 \le pos \le m$, $0 \le x \le 10^6$);
- 3 x ($0 \le x \le 10^6$).

Output

For each query of the third type print the answer to this query.

Examples

input

```
3 4 3
1 2 7
3 4 5 6
3 5
1 1 10
3 5
```

output

```
-4
9
```

input

```
7 8 12
958125 14018 215153 35195 90380 30535 204125
591020 930598 252577 333333 999942 1236 9456 82390
3 123458
2 4 444444
3 123456
1 2 355555
3 123478
3 1111
2 6 340324
3 1111
2 8 999999
2 7 595959
3 222222
3 100
```

output

```
1361307
1361311
1702804
1879305
1821765
1078115
1675180
```

Note

In the first query of the first test, the tournament is held with $k = 5$. It would be optimal to arrange players in such way (here their potentials are written):

$\underline{7}$, 3, 5, 4, 6, $\underline{1}$, $\underline{2}$ (underlined numbers are potentials of players that are from the first team).

The individual scores of players, numbered in the order of their appearance, are:

- $\max(5 + (7 - 7), 0) = 5$ for the $\underline{1}$-st player;
- $\max(5 + (3 - 7), 0) = 1$ for the 2-nd player;
- $\max(1 + (5 - 3), 0) = 3$ for the 3-rd player;
- $\max(3 + (4 - 5), 0) = 2$ for the 4-th player;
- $\max(2 + (6 - 4), 0) = 4$ for the 5-th player;
- $\max(4 + (1 - 6), 0) = 0$ for the $\underline{6}$-th player;
- $\max(0 + (2 - 1), 0) = 1$ for the $\underline{7}$-th player.

So, $score_f = 5 + 0 + 1 = 6$ and $score_s = 1 + 3 + 2 + 4 = 10$. The score difference is $6 - 10 = -4$. It can be proven, that it is the maximum possible score difference.

F. Nezzar and Chocolate Bars

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Nezzar buys his favorite snack — n chocolate bars with lengths $l_1, l_2, \ldots, l_n$. However, chocolate bars might be too long to store them properly!

In order to solve this problem, Nezzar designs an interesting process to divide them into small pieces. Firstly, Nezzar puts all his chocolate bars into a black box. Then, he will perform the following operation repeatedly until the maximum length over all chocolate bars does not exceed k.

- Nezzar picks a chocolate bar from the box with probability proportional to its length x.
- After step 1, Nezzar uniformly picks a real number $r \in (0, x)$ and divides the chosen chocolate bar into two chocolate bars with lengths r and $x - r$.
- Lastly, he puts those two new chocolate bars into the black box.

Nezzar now wonders, what is the expected number of operations he will perform to divide his chocolate bars into small pieces.

It can be shown that the answer can be represented as $\frac{P}{Q}$, where P and Q are coprime integers and $Q \not\equiv 0 \pmod{998\,244\,353}$. Print the value of $P \cdot Q^{-1} \mod 998\,244\,353$.

Input

The first line contains two integers n and k ($1 \le n \le 50, 1 \le k \le 2000$).

The second line contains n integers $l_1, l_2, \ldots, l_n$ ($1 \le l_i$, $\sum_{i=1}^{n} l_i \le 2000$).

Output

Print a single integer — the expected number of operations Nezzar will perform to divide his chocolate bars into small pieces modulo $998\,244\,353$.

Examples

input

```
1 1
2
```

output

```
4
```

input

```
1 1
1
```

output

```
0
```

input

```
1 5
1234
```

output

```
15630811
```

input

```
2 1
2 3
```

output

```
476014684
```

input

```
10 33
10 20 30 40 50 60 70 80 90 100
```

output

```
675105648
```