Problem A. Graph Partitioning

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Little Cyan Fish and Xiao Qing Yu are two good friends. Each of them has a rooted tree with n vertices. Each vertex is labeled from 1 to n. Recall that a tree is an undirected graph in which any two vertices are connected by exactly one path, or equivalently, a connected acyclic undirected graph. Let's denote $par_T(x)$ as the parent vertex of x in the tree T.

The tree owned by Little Cyan Fish is called T_1. It has the following properties:

- $T_1(V_1, E_1)$ is a tree rooted at vertex 1.
- $\forall 2 \le x \le n$, the index of the parent of x should be less than x.
  - Formally, $par_{T_1}(x) < x$ for all $2 \le x \le n$.

The tree owned by Xiao Qing Yu is called T_2. It has the following properties:

- $T_2(V_2, E_2)$ is a tree rooted at vertex n.
- $\forall 1 \le x < n$, the index of the parent of x should be greater than x.
  - Formally, $par_{T_2}(x) > x$ for all $1 \le x < n$.

Since they are good friends, they want to merge their own trees into a larger graph. Suppose $G = (V, E)$ is the new graph merged by their trees:

- V is the same as V_1 and V_2. In other words, the new graph also contains n vertices, and each vertex is labeled from 1 to n.
- E is the union of E_1 and E_2. If some edge appears in E_1 and E_2 simultaneously, it will appear in E twice.

Now you are given all the edges in G. Your task is to calculate how many different pairs of trees (T_1, T_2) could generate such a graph G. Two trees are considered different if and only if there exists an edge e that appears in one of the trees but not in the other one. Note that edges in G may appear more than once, and multiple edges are treated as different edges.

Since the answer can be quite large, you only need to output it modulo 998 244 353.

Input

The first line of the input contains a single integer n ($1 \le n \le 5 \times 10^5$).

The next $2n - 2$ lines of the input describe the edges in E. The i-th line contains two integers u_i and v_i, indicating an edge $(u_i, v_i) \in E$. Note that if some edge appears in E_1 and E_2 simultaneously, it will appear in E twice.

Note that there might be multiple edges or self-loops in the graph. And it is possible that there's no valid pair of (T_1, T_2).

Output

Print a single line contains a single integer, indicating the answer modulo 998 244 353.

Examples

standard input	standard output
2 1 2 1 2	2
1	1
3 1 2 2 3 1 3 2 2	0
6 3 4 1 3 3 5 1 6 6 5 4 2 5 4 1 2 4 1 5 3	2

Note

In the first test case, note that multiple edges might exist. Since the two edges are different, so there are two possible pairs of (T_1, T_2).

In the second test case, it is possible that the graph is empty. In this case, making both G_1 and G_2 empty is a possible choice.

In the third test case, note that there might be self-loops.

Problem B. Disjoint Set Union

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Recently, Little Cyan Fish has been learning about the Disjoint Set Union (DSU) data structure. It is a powerful data structure that allows you to add edges to a graph and test whether two vertices of the graph are connected.

The DSU maintains a rooted forest structure consisting of n vertices. Each vertex x ($1 \le x \le n$) has a unique parent $f[x]$. If $x = f[x]$, then x is the root of its subtree. Initially, each vertex forms a single rooted tree. That is, $f[x] = x$ for all $1 \le x \le n$.

The basic interface of DSU consists of these two operations:

- **find** x: returns the root of the tree where x is located.
- **unite** x y: let $x' \leftarrow$ **find**(x) and $y' \leftarrow$ **find**(y). If $x' = y'$, do nothing. Otherwise, modify the parent of x' to y'.

To speed up the **unite** operation, Little Cyan Fish uses an optimization called *Path Compression*:

- If we call **find**(x) for some vertex x, we set the parent of each vertex from x to the root directly to the root.

The following pseudocode describes the details of the DSU.

Algorithm 1 An implementation of DSU with Path Compression

```
1:  procedure FIND(f, x)
2:      if x = f[x] then
3:          return x
4:      end if
5:      f[x] ← find(f, f[x])
6:      return f[x]
7:  end procedure
8:  procedure UNITE(f, x, y)
9:      x ← FIND(f, x)
10:     y ← FIND(f, y)
11:     if x ≠ y then
12:         f[x] ← y
13:     end if
14: end procedure
```

Little Cyan Fish loves the DSU very much, so he would like to play with it. He got an array f of length n, where $f[i] = i$ in the beginning. Then, Little Cyan Fish did the following operations many times (possibly zero):

- Choose an integer $1 \le x \le n$, apply FIND(x).
- Choose two integers $1 \le x \le n$ and $1 \le y \le n$, apply UNITE(x, y).

He will give you the array f after all his operations. However, you would like to transform the array f into another given array g by using the DSU operations described above. You are wondering if it is possible to apply any additional operations so that $f[i] = g[i]$ for all $1 \le i \le n$.

Input

There are multiple test cases. The first line contains one integer T ($1 \le T \le 10^5$), representing the number of test cases.

For each test case, the first line contains one positive integer n ($3 \le n \le 1\,000$).

The next line contains n integers $f_1, f_2, \ldots, f_n$ denoting the array f after Little Cyan Fish's operations. It is guaranteed that the array f can be generated by using the operation above.

The following line contains n integers $g_1, g_2, \ldots, g_n$ denoting the array g that you would like to transform the array f into.

It is guaranteed that the sum of n^2 over all test cases does not exceed 5×10^6.

Output

For each test case, if it is impossible to transform the array f into the array g, print a single line `NO`.

Otherwise, the first line of the output should contain a single word `YES`.

The next line of the output should contain an integer m ($0 \le m \le 2 \cdot n^2$), indicating the number of operations you used.

The next m lines describe the operations you used. Each operation is described in the following format:

- 1 x: Call FIND(x).
- 2 x y: Call UNITE(x, y).

If there are multiple solutions, you may print any of them. It can be proved that if any solution exists, then there's a plan consisting of no more than $2 \cdot n^2$ operations.

Example

standard input	standard output
5 3 1 2 3 2 2 3 4 1 2 3 3 1 1 1 2 5 1 2 3 4 5 2 3 4 5 5 5 1 1 1 1 1 1 2 3 4 5 6 1 2 2 4 5 6 1 1 5 1 4 2	YES 1 2 1 2 YES 4 2 3 2 1 4 2 2 1 1 3 YES 4 2 1 2 2 1 3 2 2 4 2 3 5 NO YES 7 2 6 2 2 2 5 1 3 2 2 4 1 2 2 2 1 1 2

Problem C. DFS Order 3

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Little Cyan Fish has a tree with n vertices. Each vertex is labeled from 1 to n. Now he wants to start a depth-first search at each vertex x. The DFS order is the order of nodes visited during the depth-first search. A vertex appears in the j-th ($1 \le j \le n$) position in this order means it is visited after $j-1$ other vertex. Because sons of a node can be iterated in arbitrary order, multiple possible depth-first orders exist.

The following pseudocode describes the way to generate a DFS order. The function GENERATE(x) returns a DFS order starting at vertex x:

Algorithm 2 An implementation of depth-first search

```
1:  procedure DFS(vertex x)
2:      Append x to the end of dfs_order
3:      for each son y of x do                ▷ Sons can be iterated in arbitrary order.
4:          DFS(y)                            ▷ The order might be different in each iteration.
5:      end for
6:  end procedure
7:  procedure GENERATE(x)
8:      Root the tree at vertex x
9:      Let dfs_order be a global variable
10:     dfs_order ← empty list
11:     DFS(x)
12:     return dfs_order
13: end procedure
```

Let D_i be the returned array after calling GENERATE(x). Little Cyan Fish wrote down all the n sequences $D_1, D_2, \cdots, D_n$. Years later, he can no longer remember the structure of the original tree. Little Cyan Fish is wondering how to recover the original tree by using these n sequences. Please help him!

Input

There are multiple test cases. The first line contains one integer T ($1 \le T \le 10^5$), representing the number of test cases.

For each test case, the first line contains one positive integer n ($1 \le n \le 1\,000$), indicating the number of vertices of the tree.

The next n lines describe the DFS order of the original tree. In the i-th line of htese lines contains n integers $D_{i,1}, D_{i,2}, \cdots, D_{i,n}$, describes a DFS order. It is guaranteed that $D_{i,1} = i$ and D_i is a valid DFS order of the original tree.

It is guaranteed that the sum of n^2 over all test cases does not exceed 2×10^6.

Output

For each test case, you need to output $n-1$ lines, that describes the tree you recovered. In each of the $n-1$ lines, you need to output two integers u_i and v_i ($1 \le u_i, v_i \le n$), which means there's an edge between vertex u_i and vertex v_i. If there are multiple possible solutions, you may print any of them. It is guaranteed that at least one solution exists.

Example

standard input	standard output
4 2 1 2 2 1 3 1 2 3 2 1 3 3 2 1 4 1 2 3 4 2 1 3 4 3 2 4 1 4 2 1 3 5 1 2 4 3 5 2 4 1 3 5 3 5 1 2 4 4 2 1 3 5 5 3 1 2 4	1 2 1 2 2 3 1 2 2 3 2 4 1 2 1 3 2 4 3 5

Problem D. Flower's Land 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

f: "*Cyan Fish! Come and check out my new problem about bracket sequences!*"
Q: "*What?! Isn't that a well-known problem?*"
f: "*Ugh... Well, then how about this one?*"

Little Cyan Fish and Big Flower Letter are two close friends. In celebration of the 17th anniversary of the founding of Flower's Land, Little Cyan Fish has devised a new game:

- The game is played with a string $s = s_1 s_2 \cdots s_n$ composed of the digits 0, 1, and 2.
- The game proceeds as follows:
  1. The player selects an index i ($1 \le i < n$) such that $s_i = s_{i+1}$.
     - Notably, if no such i exists, the game ends immediately.
  2. The characters s_i and s_{i+1} are then removed, and the player returns to step 1.
- If the player can completely erase the string (leaving an empty string), he wins the game. If not, he loses.

Keen to challenge Little Cyan Fish, Big Flower Letter presents him with a string $s = s_1 s_2 \cdots s_n$ of length n. She then proposes the following operations:

- `1` l r: For each $l \le i \le r$, update $s_i \leftarrow (s_i + 1) \bmod 3$.
- `2` l r: Assuming $t = s[l \cdots r] = s_l s_{l+1} \cdots s_{r-1} s_r$, Little Cyan Fish needs to determine if the player can win the game if they were to play it using the string t.

Can you help Little Cyan Fish respond to all the queries?

Input

The first line of the input contains two integers n and q ($1 \le n, q \le 5 \times 10^5$), representing the length of string s and the number of queries.

The next line of the input contains a string s, indicating the string that Big Flower Letter presents to Little Cyan Fish. It is guaranteed that $|s| = n$ and s is composed of the digits 0, 1, and 2.

The next q lines describe all the operations in the following format.

- `1` l r
- `2` l r

It is guaranteed that $1 \le l \le r \le n$.

Output

For each operation 2, if the player can win the game, output a single line containing a single word `Yes`. Otherwise, output a single line containing a single word `No`.

Example

standard input	standard output
8 9 01211012 2 4 5 2 3 6 1 6 8 1 6 8 2 3 6 2 1 8 1 1 1 1 7 7 2 1 8	Yes No Yes No Yes

Note

In the sample test, we start with $s =$ `01211012`.

Operation 1 is a query operation. With $s[4 \cdots 5] =$ `11`, we can select the index $i = 1$ in the first round to win the game. Therefore, we should output `Yes`.

Operation 2 is a query operation. For $s[3 \cdots 6] =$ `2110`, it's impossible to remove all characters. As a result, we should output `No`.

Operation 3 is an update operation. We need to update all s_i where $6 \leq i \leq 8$. After this modification, s becomes `01211120`.

Operation 4 is an update operation. We need to update all s_i where $6 \leq i \leq 8$. After this modification, s becomes `01211201`.

Operation 5 is a query operation. For $s[3 \cdots 6] =$ `2112`, we can select the index $i = 2$ in the first round. The string then becomes 22, and we can select the index $i = 1$ in the next round to win the game. Therefore, we should output `Yes`.

Operation 6 is a query operation. For $s[1 \cdots 8] =$ `01211201`, it's impossible to remove all characters. As a result, we should output `No`.

Operation 7 is an update operation. We need to update all s_i where $1 \leq i \leq 1$. After this modification, s becomes `11211201`.

Operation 8 is an update operation. We need to update all s_i where $7 \leq i \leq 7$. After this modification, s becomes `11211211`.

Operation 9 is a query operation. For $s[1 \cdots 8] =$ `11211211`, it's possible to remove all characters. Therefore, we should output `Yes`.

Problem E. CCPC String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

To prepare a task for the CCPC Final, Little Cyan Fish is studying basic string theory. Today, Little Cyan Fish has learned the concept of the CCPC string. A string s is called a CCPC string if and only if there exists a positive integer $t \geq 1$, such that $s = \texttt{c}^{2t}\texttt{p}\texttt{c}^{t}$.

Here, $\texttt{c}^{k}$ represents the string consisting of the character `c` repeated k times, and uv denotes the string obtained by concatenating strings u and v. For example, `ccpc`, `ccccpcc`, and `ccccccpccc` are CCPC strings, but `p`, `cpc`, `ccpcc`, `ccppc`, and `cccpc` are not.

Now, Little Cyan Fish has a string S consisting of `c`, `p`, and question marks (`?`). He wants to calculate the number of pairs of integers (l, r) that satisfy the following conditions:

- $1 \leq l \leq r \leq |S|$
- for the string $T = S[l \cdots r]$, it is possible to replace the question marks (`?`) to `c` or `p`, so that the string is an CCPC string.

Input

There are multiple test cases. The first line contains one integer T ($1 \leq T \leq 10^5$), representing the number of test cases.

For each test case, the first line contains a single string S. The string S consists only of the English letters `c`, `p`, and the question mark (?).

It is guaranteed that the sum of $|S|$ over all test cases does not exceed 10^6.

Output

For each test case, output a single line consists a single integer, indicating the answer.

Example

standard input	standard output
5 ?cpc ccp?? ???c??? ?c???cp?? ?c?????cccp????	1 1 4 5 14

Note

In the first example, all valid pairs of (l, r) are as follows.

$l =$	$r =$	$S[l \cdots r]$	**Replaced String**
1	4	?cpc	ccpc

In the second example, all valid pairs of (l, r) are as follows.

$l =$	$r =$	$S[l \cdots r]$	**Replaced String**
1	4	ccp?	ccpc

In the third example, all valid pairs of (l, r) are as follows.

$l =$	$r =$	$S[l \cdots r]$	**Replaced String**
1	4	???c	ccpc
3	6	?c??	ccpc
4	7	c???	ccpc
1	7	???c???	ccccpcc

In the fourth example, all valid pairs of (l, r) are as follows.

$l =$	$r =$	$S[l \cdots r]$	**Replaced String**
1	4	?c??	ccpc
2	5	c???	ccpc
3	6	???c	ccpc
5	8	?cp?	ccpc
3	9	???cp??	ccccpcc

In the fifth example, all valid pairs of (l, r) are as follows.

$l =$	$r =$	$S[l \cdots r]$	**Replaced String**
1	4	?c??	ccpc
2	5	c???	ccpc
3	6	????	ccpc
4	7	????	ccpc
5	8	???c	ccpc
9	12	ccp?	ccpc
12	15	????	ccpc
1	7	?c?????	ccccpcc
2	8	c?????c	ccccpcc
3	9	?????cc	ccccpcc
7	13	?cccp??	ccccpcc
1	10	?c?????ccc	ccccccpccc
5	14	???cccp???	ccccccpccc
3	15	?????cccp????	ccccccccpcccc

Problem F. Chase Game 3

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

After becoming the *Chinese Elephant Chess Champion*, Teacher $\mathcal{D}$ has designed a new two-player game called Tie-Tie.

In the Tie-Tie Game, there are n vertices numbered from 1 to n. Two bidirectional chains L_1 and L_2 connect these n vertices. The i-th edge of L_1 connects node i and $i+1$ $(1 \le i \le n-1)$. The i-th edge of L_2 connects node p_i and p_{i+1} $(1 \le i \le n-1)$.

The two players in the game are called Little Cyan Fish and Xiao Qing Yu. Before the game starts, Little Cyan Fish must choose a starting node A, and Xiao Qing Yu must choose a starting node B. After that, they will take turns acting, with Little Cyan Fish going first:

- Little Cyan Fish can choose to stay in place or move to another vertex along an edge of L_1;
- Xiao Qing Yu can choose to stay in place or move to another vertex along an edge of L_2.

If at some point Little Cyan Fish and Xiao Qing Yu are at the same vertex, then a tie-tie will occur. Xiao Qing Yu loves tie-ties very much, but Little Cyan Fish does not. Therefore, Xiao Qing Yu will try to make the tie-tie happen, and Little Cyan Fish will try to prevent it. Both players are smart enough to adopt the optimal strategy for the game.

Teacher $\mathcal{D}$ is also a fan of Tie-Tie. If **no matter which initial nodes the two players choose**, Xiao Qing Yu has a strategy to achieve a tie-tie with Little Cyan Fish within a finite number of steps, then Teacher $\mathcal{D}$ will be happy. Please help Teacher $\mathcal{D}$ determine whether a tie-tie will occur in all possible initial states.

Input

There are multiple test cases. The first line contains one integer T $(1 \le T \le 10^5)$, representing the number of test cases.

For each test case, the first line contains one positive integer n $(2 \le n \le 4 \times 10^5)$.

The next line contains n integers $p_1, p_2, \cdots, p_n$. It is guaranteed that p is a permutation of $[1, n]$.

It is guaranteed that the sum of n over all test cases does not exceed 4×10^5.

Output

For each test case, if no matter which initial nodes the two players choose, Xiao Qing Yu has a strategy to achieve a tie-tie with Little Cyan Fish within a finite number of steps, output a single line consists a single word `Yes`. Otherwise, output a single line consists of a single word `No`.

Example

standard input	standard output
5 2 1 2 3 2 3 1 4 1 4 3 2 5 1 5 2 3 4 6 1 2 3 4 5 6	Yes Yes No No Yes

Problem G. Recover the String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

There is a string s consisting of lowercase English letters. Little Cyan Fish is fond of this string, so he noted down all its unique substrings and added directed edges between them. Specifically, for two distinct substrings s_1 and s_2, if there is a letter c such that $s_2 = s_1 + c$ or $s_2 = c + s_1$, he drew a directed edge from s_1 to s_2. Afterward, he removed all the duplicated edges. As a result, this forms a Directed Acyclic Graph (DAG).

Unfortunately, Little Cyan Fish has forgotten the original string s. Even worse, he also can't remember which substring corresponds to each node in the DAG. Now that he only has the DAG, your task is to help Little Cyan Fish recover the original string s. Since there could be multiple possible solutions, Little Cyan Fish is only interested in the one with the smallest lexicographical order.

Input

There are multiple test cases. The first line contains one integer T ($1 \le T \le 10^5$), representing the number of test cases.

For each test case, the first line of the input contains two integers n, m ($1 \le n \le 10^6, 0 \le m \le 2 \times 10^6$), representing the number of nodes and edges.

The i-th of the next m lines contains two integers u_i, v_i ($1 \le u_i, v_i \le n$), representing a directed edge from u_i to v_i. It is guaranteed that at least one valid solution exists.

It is guaranteed that the sum of n over all test cases does not exceed 10^6, and the sum of m over all test cases does not exceed 2×10^6.

Output

For each test case, output a single line contains a string consists of lowercase English letters, representing the answer.

Example

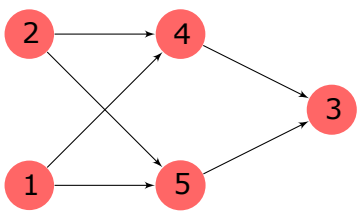
standard input	standard output
3 1 0 5 6 2 4 2 5 5 3 4 3 1 5 1 4 8 11 1 2 1 4 1 6 2 5 3 4 3 6 4 5 4 7 5 8 6 7 7 8	a aba aaba

Note

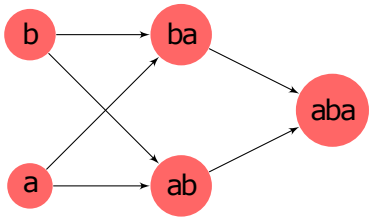
The DAG corresponding to the first sample test case is as follows.

The string corresponding to each node is as follows.

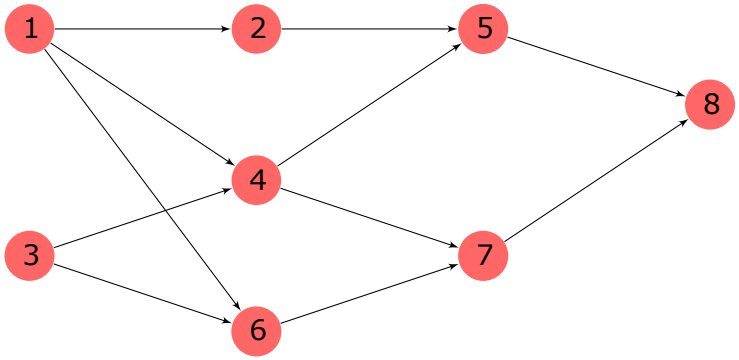
The DAG corresponding to the second sample test case is as follows.



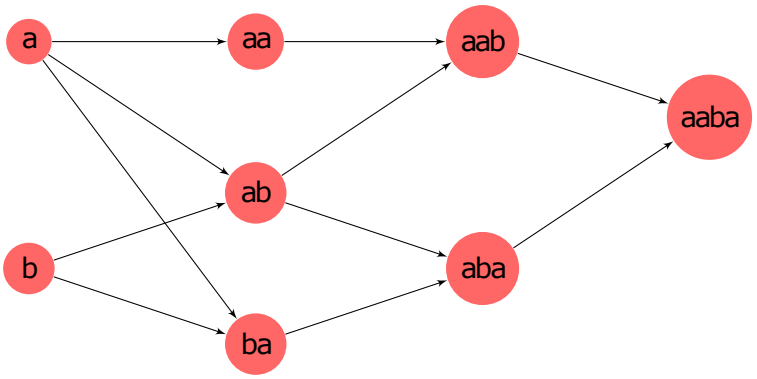
The string corresponding to each node is as follows.



The DAG corresponding to the third sample test case is as follows.



The string corresponding to each node is as follows.



Problem H. This is not an Abnormal Team!

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

Hope for normalcy,
all gains to be had,
And a bright future,
no longer sad.

(Not Little Cyan Fish)

There are n_1 boys and n_2 girls in the XCPC Training team, and m pairs of boys and girls are in relationships. One person may be in a relationship with multiple partners. However, there will not be any relationships between people of the same gender.

Little Cyan Fish is the coach of the XCPC Training team, and he wants to divide them into several XCPC teams. Each team can have no more than 3 members. To make the teams not being *abnormal*, if there is more than one member in a team, at least one member must be in a relationship with all other team members.

However, participating in XCPC alone is not as enjoyable, so Little Cyan Fish aims to minimize the number of single-person teams. Additionally, when one person is in a relationship with the other two teammates, the team can become unstable. Therefore, Little Cyan Fish also wants to minimize the number of three-person teams after minimizing the number of single-person teams.

The Little Cyan Fish is very anxious to find a way to divide the teams. To make everything back to normal, you need to help him to complete the dividing team task so that his teams can fight for the champion. XCPC is his favorite business, please help him!

Input

The first line of the input contains three integers $n_1, n_2, m (1 \le n_1, n_2 \le 10^5, 1 \le m \le 2 \times 10^5)$, representing the number of boys, the number of girls, and the number of relationships.

The i-th of the next m lines contains two integers u_i, v_i $(1 \le u_i \le n_1, 1 \le v_i \le n_2)$, representing a relationship between the u_i-th boy and the v_i-th girl.

It is guaranteed that all (u_i, v_i) are pairwise distinct.

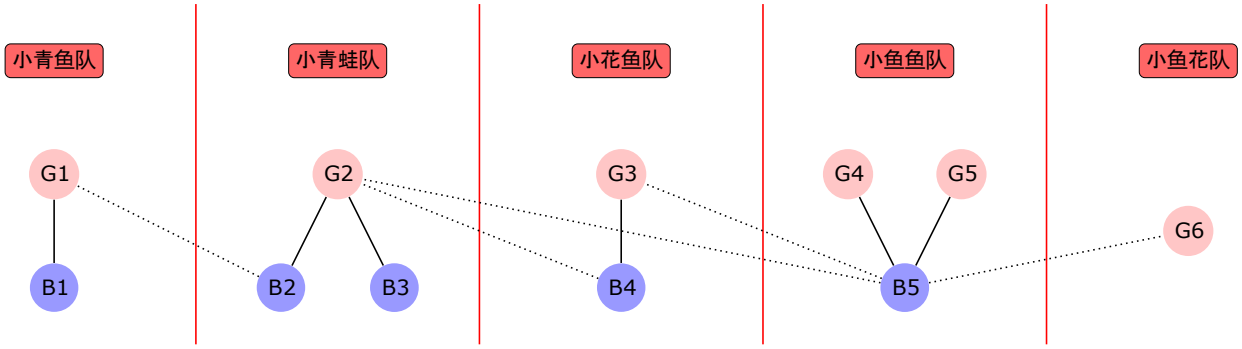
Output

Output a single line contains two integers, indicating the number of teams with 1 member and the number of teams with 3 members.

Example

standard input	standard output
5 6 10 1 1 2 1 2 2 3 2 4 2 4 3 5 3 5 4 5 5 5 6	1 2

Note



Problem I. Not Another Range Query Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

What age is it that you are still solving traditional range query problems?

Little Cyan Fish, the biggest fan of range query problems, would like to find the best range query task in the year 3202. Luckily, he discovers a problem with binary strings. He adores binary strings, as they led to the *Best Problem* back in 2202.

For a binary string $s = s_1 s_2 \cdots s_n$ of length n, let's denote $L(s)$ as the set of all indices i ($1 \le i \le n$) that satisfy $i = 1$ or $s_i \neq s_{i-1}$. For instance, $L(\texttt{0011100}) = \{1, 3, 6\}$, $L(\texttt{00101101}) = \{1, 3, 4, 5, 7, 8\}$, and $L(\texttt{000}) = \{1\}$. Specifically, if $s = \varepsilon$ is an empty string, then we have $L(\varepsilon) = \varnothing$, which is an empty set.

Let $f(s)$ be the string obtained by removing all characters in s whose indices are in $L(s)$. The table below shows several examples of the function f.

Binary String s	**The set** $L(s)$	**Removed Characters**	**Binary String** $f(s)$
0011100	$\{1, 3, 6\}$	$\underline{0}0\underline{1}1\underline{0}0$	0110
00101101	$\{1, 3, 4, 5, 7, 8\}$	$\underline{0}0\underline{1}0\underline{1}1\underline{0}1$	01
000	$\{1\}$	$\underline{0}00$	00
01	$\{1, 2\}$	$\underline{01}$	ε
ε	$\{\}$		ε

The problem that Little Cyan Fish found asks him to determine the length of the string $f^k(s)$, where $f^k(s)$ is defined as follows:

$$f^k(s) = \begin{cases} f^{k-1}(f(s)) & k \ge 1 \\ s & k = 0 \end{cases}$$

Little Cyan Fish would like to create a range query problem with it. So, he gives you a string $s = s_1 s_2 \cdots s_n$ and q queries. In each query, you are given three integers l, r, and k. Your task is to output the length of the string $f^k(s[l \cdots r])$, where $s[l \cdots r]$ is $s_l s_{l+1} \cdots s_r$.

Input

The first line contains two integers n and q ($1 \le n, q \le 5 \times 10^5$), representing the length of the string s and the number of queries Little Cyan Fish would like to perform.

The next line of the input contains a binary string s ($|s| = n$, $s_i \in \{0, 1\}$).

The following q lines describe the queries. For the i-th line of these lines, it contains three integers l, r, k ($1 \le l \le r \le n, 0 \le k \le n$).

Output

For each query, output a single line contains a single integer, indicating the length of $f^k(s[l \cdots r])$.

Example

standard input	standard output
9 7 100110001 2 5 1 3 6 1 4 8 2 2 7 1 1 9 1 1 9 0 1 9 8	2 1 1 3 4 9 0

Note

In the first query, $s[2 \cdots 5] = \texttt{0011}$, and $f(\texttt{0011}) = \texttt{01}$. So the answer is 2.

In the second query, $s[3 \cdots 6] = \texttt{0110}$, and $f(\texttt{0110}) = \texttt{1}$. So the answer is 1.

In the third query, $s[4 \cdots 8] = \texttt{11000}$, and $f^2(\texttt{11000}) = f(\texttt{100}) = \texttt{0}$. So the answer is 1.

In the fourth query, $s[2 \cdots 7] = \texttt{001100}$, and $f(\texttt{001100}) = \texttt{010}$. So the answer is 3.

In the fifth query, $s[1 \cdots 9] = \texttt{100110001}$, and $f(\texttt{100110001}) = \texttt{0100}$. So the answer is 4.

In the sixth query, $s[1 \cdots 9] = \texttt{100110001}$, and $f^0(\texttt{100110001}) = \texttt{100110001}$. So the answer is 9.

In the seventh query, $s[1 \cdots 9] = \texttt{100110001}$, and $f^8(\texttt{100110001}) = f^7(\texttt{0100}) = f^6(\texttt{0}) = f^5(\varepsilon) = \varepsilon$. So the answer is 0.

Problem J. Best Carry Player 3

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

After learning the bitwise XOR operation, Little Cyan Fish would like to practice it by playing the following game.

Little Cyan Fish has an integer X. He wants to transform X into another integer Y using addition, subtraction, or bitwise XOR operations.

Because he is not a *Best Carry Player*, he cannot understand the operations of addition and subtraction between large numbers. Thus, he can only perform the following operations:

- $(+)$: Change X to $X + 1$.
- $(-)$: Change X to $X - 1$. This operation is not available if $X = 0$.
- $(\oplus)$: Choose an integer $0 \le t \le K$, change X to $X \oplus t$, where $\oplus$ is the bitwise XOR operator.

Given integers X, Y and K, you need to calculate the minimum number of operations that Little Cyan Fish needs to perform to transform X into Y.

Input

There are multiple test cases. The first line contains one integer T ($1 \le T \le 10^5$), representing the number of test cases.

For each test case, the first line contains three integers X, Y and K ($0 \le X, Y, K < 2^{60}$).

Output

For each test case, output a single line contains a single integer, indicating the minimum number of operations that Little Cyan Fish needs to perform.

Example

standard input	standard output
8 4 5 0 5 8 3 9 2 6 15 28 5 97 47 8 164 275 38 114514 1919 810 0 1152921504606846975 1	1 2 3 5 11 6 331 1152921504606846975

Note

In the first test case, the optimal plan is:

- $(+)$: Change X from 4 to $4 + 1 = 5$.

So the answer is 1.

In the second test case, the optimal plan is:

- ($\oplus 2$): Choose $t = 2$, change X from 5 to $5 \oplus 2 = 7$.
- ($+$): Change X from 7 to $7 + 1 = 8$.

So the answer is 2.

In the third test case, the optimal plan is:

- ($-$): Choose X from 9 to $9 - 1 = 8$.
- ($-$): Change X from 8 to $8 - 1 = 7$.
- ($\oplus 5$): Choose $t = 5$, Change X from 7 to $7 \oplus 5 = 2$.

So the answer is 3.

Problem K. Balancing Sequences

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

After traveling to Gensokyo, Little Cyan Fish obtained two sequences a_1 and a_2. Each sequence contains n integers ranging from 1 to $2n$. All of the $2n$ integers are pairwise distinct.

He wants to transform a_1, a_2 into b_1, b_2. Unfortunately, the sequences have a self-balancing system, so the only operation he can perform is to choose four integers (x_1, x_2, y_1, y_2) and swap the elements a_{x_1,x_2} and a_{y_1,y_2}. To protect the self-balancing system, these chosen integers must satisfy:

- $x_1, y_1 \in [1, 2]$ and $x_2, y_2 \in [1, n]$.
- $x_2 \neq y_2$.
- $a_{x_1,x_2} > a_{3-x_1,x_2}$.
- $a_{y_1,y_2} > a_{3-y_1,y_2}$.

Little Cyan Fish would like to know whether he can transform a_1, a_2 into b_1, b_2, so he asked you for help. If it is possible, you need to provide a plan to guide him.

Input

There are multiple test cases. The first line contains one integer T ($1 \leq T \leq 10^5$), representing the number of test cases.

For each test case, the first line contains one integer n ($2 \leq n \leq 2 \times 10^3$), indicating the size of a_1, a_2, b_1, and b_2.

The next line contains n elements, describing a_1 ($1 \leq a_{1,i} \leq 2n$).

The next line contains n elements, describing a_2 ($1 \leq a_{2,i} \leq 2n$). All the $2n$ integers in the sequences a_1 and a_2 are pairwise distinct.

The next line contains n elements, describing b_1 ($1 \leq b_{1,i} \leq 2n$).

The next line contains n elements, describing b_2 ($1 \leq b_{2,i} \leq 2n$). All the $2n$ integers in the sequences b_1 and b_2 are pairwise distinct.

It is guaranteed that the sum of n^2 over all test cases does not exceed 4×10^6.

Output

If it's not possible to transform the arrays a_1 and a_2 into b_1 and b_2, output a single line consists a single integer -1.

Otherwise, output a single integer s ($s \in [0, 5n]$) representing the number of steps required to transform a_1 and a_2 into b_1 and b_2.

In the next s lines, for each line, output four numbers x_1, x_2, y_1, y_2 ($1 \leq x_1, y_1 \leq 2$, $1 \leq x_2, y_2 \leq n$), indicating that you should swap a_{x_1,x_2} and a_{y_1,y_2} in this step.

It can be proven that if the transformation is possible, then it can be completed within $5n$ steps.

Example

standard input	standard output
2 2 1 2 3 4 4 3 2 1 3 1 2 4 3 5 6 1 2 4 5 3 6	-1 1 2 2 2 1

Problem L. Completely Multiplicative Function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A completely multiplicative function is an arithmetic function (that is, a function whose domain is the natural numbers), such that $f(1) = 1$ and $f(ab) = f(a)f(b)$ holds for all positive integers a and b.

Little Cyan Fish loves the concept of the completely multiplicative function, so he wants to find a completely multiplicative function $f : \mathbb{N} \to \{-1, 1\}$ satisfying the sum of the value of $f(i)$ for all $1 \le i \le n$ must be exactly k.

Formally, you need to find a function $f(x)$ satisfying:

- $f(x) \in \{-1, 1\}$ for all integers x.
- $f(x)f(y) = f(xy)$ for all integers x and y.
- $f(1) + f(2) + \cdots + f(n) = k$ for given integers n and k.

To test if you have truly understood the beauty of completely multiplicative functions, Little Cyan Fish has asked you to find such a function f.

Input

There are multiple test cases. The first line contains one integer T ($1 \le T \le 10^5$), representing the number of test cases.

For each of the test case, the first line contains two integers n and k ($0 \le k \le n \le 10^6$, $n \ge 1$).

It is guaranteed that the sum of n over all test cases does not exceed 2×10^6.

Output

For each test case:

If there does not exist a solution, output one line containing a single integer -1.

Otherwise, you should output a single line containing $f(1), f(2), \cdots, f(n)$, separated by spaces. The value of each $f(i)$ must be either 1 and -1, and the sum of these values must be exactly k.

If there are multiple solutions, you may print any of them.

Example

standard input	standard output
4 4 2 10 0 10 1 10 10	1 -1 1 1 1 -1 -1 1 1 1 -1 -1 1 -1 -1 1 1 1 1 1 1 1 1 1 1

Problem M. Expression 3

Input file:	standard input
Output file:	standard output
Time limit:	5 seconds
Memory limit:	1024 megabytes

Little Cyan Fish has n numbers $a_1, a_2, \cdots, a_n$ and $n-1$ operators("+", "-") $op_1, op_2, \cdots, op_{n-1}$, which are arranged in the form $a_1\ op_1\ a_2\ op_2\ a_3\ \cdots\ a_n$.

He wants to erase numbers one by one. In the i-th round, there are $n+1-i$ numbers remaining. He can erase two adjacent numbers and the operator between them and then put a new number (derived from this operation) in this position. After $n-1$ rounds, only one number remains. The result of this sequence of operations is the last number remaining.

He wants to know the sum of the results of all different sequences of operations. The number can be large, output it modulo 998 244 353. Two sequences of operations are considered different if and only if he chooses different numbers in one round.

Input

The first line of the input contains one integer n ($2 \le n \le 2 \times 10^5$).

The second line of the input contains n integers $a_1, a_2, \cdots, a_n$ ($0 \le a_i \le 10^9$).

The third line of the input contains a string with length $n-1$ consisting of "+", "-", which represents the operator sequence.

Output

Output the answer modulo 998 244 353.

Examples

standard input	standard output
4 9 1 4 1 -+-	46
5 1 2 3 4 5 +-+-	998244313

Note

In the first example, there are six possible ways to erase numbers:

- $\underline{9-1}+4-1 \Longrightarrow \underline{8+4}-1 \Longrightarrow \underline{12-1} \Longrightarrow 11$
- $\underline{9-1}+4-1 \Longrightarrow 8+\underline{4-1} \Longrightarrow \underline{8+3} \Longrightarrow 11$
- $9-\underline{1+4}-1 \Longrightarrow \underline{9-5}-1 \Longrightarrow \underline{4-1} \Longrightarrow 3$
- $9-\underline{1+4}-1 \Longrightarrow 9-\underline{5-1} \Longrightarrow \underline{9-4} \Longrightarrow 5$
- $9-1+\underline{4-1} \Longrightarrow \underline{9-1}+3 \Longrightarrow \underline{8+3} \Longrightarrow 11$
- $9-1+\underline{4-1} \Longrightarrow 9-\underline{1+3} \Longrightarrow \underline{9-4} \Longrightarrow 5$

So the answer is $11+11+3+5+11+5=46$.