A. Feeding Chicken

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Long is a huge fan of CFC (Codeforces Fried Chicken). But the price of CFC is increasing, so he decides to breed the chicken on his own farm.

His farm is presented by a rectangle grid with r rows and c columns. Some of these cells contain rice, others are empty. k chickens are living on his farm. **The number of chickens is not greater than the number of cells with rice on the farm.**

Long wants to give his chicken playgrounds by assigning these farm cells to his chickens. He would like to satisfy the following requirements:

- Each cell of the farm is assigned to **exactly one** chicken.
- Each chicken is assigned **at least one** cell.
- The set of cells assigned to every chicken forms a connected area. More precisely, if two cells (x, y) and (u, v) are assigned to the same chicken, this chicken is able to walk from (x, y) to (u, v) by passing only its cells and moving from each cell to another cell sharing a side.

Long also wants to prevent his chickens from fighting for food. Hence he wants the difference between the maximum and the minimum number of cells with rice assigned to a chicken to be as small as possible. Please help him.

Input

Each test contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 2 \cdot 10^4$). Description of the test cases follows.

The first line of each test case contains three integers r, c and k ($1 \le r, c \le 100, 1 \le k \le 62$), representing the size of Long's farm and the number of chickens Long has.

Each of the next r lines contains c characters, each is either "." or "R", representing an empty cell or a cell with rice. It is guaranteed that **the number of chickens is not greater than the number of cells with rice on the farm.**

It is guaranteed that the sum of $r \cdot c$ over all test cases does not exceed $2 \cdot 10^4$.

Output

For each test case, print r lines with c characters on each line. Each character should be either a lowercase English character, an uppercase English character, or a digit. Two characters should be equal if and only if the two corresponding cells are assigned to the same chicken. **Uppercase and lowercase characters are considered different, so "A" and "a" belong to two different chickens.**

If there are multiple optimal answers, print any.

Example

input

```
4
3 5 3
..R..
...R.
....R
6 4 6
R..R
R..R
RRRR
RRRR
R..R
R..R
5 5 4
RRR..
R.R..
RRR..
R..R.
R...R
2 31 62
RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
```

output

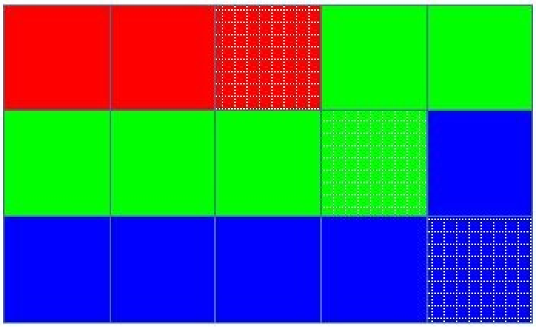
```
11122
22223
33333
```

```
aacc
aBBc
aBBc
CbbA
CbbA
CCAA
11114
22244
32444
33344
33334
abcdefghijklmnopqrstuvwxyzABCDE
FGHIJKLMNOPQRSTUVWXYZ0123456789
```

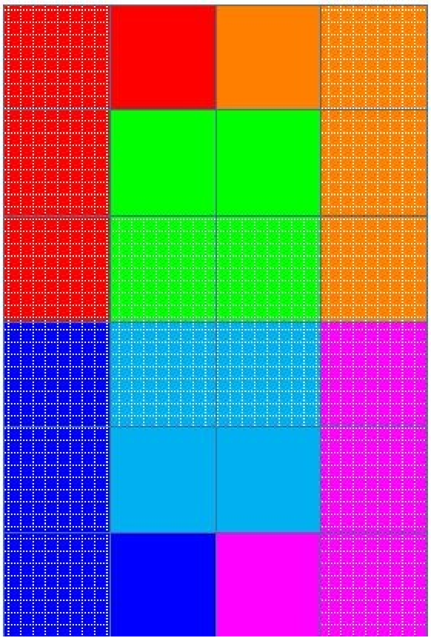
Note

These pictures explain the sample output. Each color represents one chicken. Cells filled with patterns (not solid colors) contain rice.

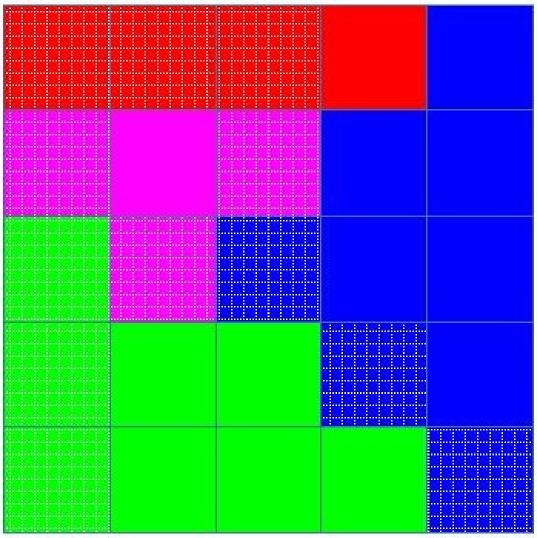
In the first test case, each chicken has one cell with rice. Hence, the difference between the maximum and the minimum number of cells with rice assigned to a chicken is 0.



In the second test case, there are 4 chickens with 3 cells of rice, and 2 chickens with 2 cells of rice. Hence, the difference between the maximum and the minimum number of cells with rice assigned to a chicken is $3-2=1$.



In the third test case, each chicken has 3 cells with rice.



In the last test case, since there are 62 chicken with exactly 62 cells of rice, each chicken must be assigned to exactly one cell. The sample output is one of the possible way.

B1. Send Boxes to Alice (Easy Version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the easier version of the problem. In this version, $1 \le n \le 10^5$ and $0 \le a_i \le 1$. You can hack this problem only if you solve and lock both problems.

Christmas is coming, and our protagonist, Bob, is preparing a spectacular present for his long-time best friend Alice. This year, he decides to prepare n boxes of chocolate, numbered from 1 to n. Initially, the i-th box contains a_i chocolate pieces.

Since Bob is a typical nice guy, he will not send Alice n empty boxes. In other words, **at least one of $a_1, a_2, \ldots, a_n$ is positive**. Since Alice dislikes coprime sets, she will be happy only if there exists some integer $k > 1$ such that the number of pieces in each box is divisible by k. Note that Alice won't mind if there exists some empty boxes.

Charlie, Alice's boyfriend, also is Bob's second best friend, so he decides to help Bob by rearranging the chocolate pieces. In one second, Charlie can pick up a piece in box i and put it into either box $i-1$ or box $i+1$ (if such boxes exist). Of course, he wants to help his friend as quickly as possible. Therefore, he asks you to calculate the minimum number of seconds he would need to make Alice happy.

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the number of chocolate boxes.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 1$) — the number of chocolate pieces in the i-th box.

It is guaranteed that at least one of $a_1, a_2, \ldots, a_n$ is positive.

Output

If there is no way for Charlie to make Alice happy, print -1.

Otherwise, print a single integer x — the minimum number of seconds for Charlie to help Bob make Alice happy.

Examples

input

```
3
1 0 1
```

output

```
2
```

input

```
1
1
```

output

```
-1
```

B2. Send Boxes to Alice (Hard Version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the harder version of the problem. In this version, $1 \le n \le 10^6$ and $0 \le a_i \le 10^6$. You can hack this problem if you locked it. But you can hack the previous problem only if you locked both problems

Christmas is coming, and our protagonist, Bob, is preparing a spectacular present for his long-time best friend Alice. This year, he decides to prepare n boxes of chocolate, numbered from 1 to n. Initially, the i-th box contains a_i chocolate pieces.

Since Bob is a typical nice guy, he will not send Alice n empty boxes. In other words, **at least one of $a_1, a_2, \ldots, a_n$ is positive**. Since Alice dislikes coprime sets, she will be happy only if there exists some integer $k > 1$ such that the number of pieces in each box is divisible by k. Note that Alice won't mind if there exists some empty boxes.

Charlie, Alice's boyfriend, also is Bob's second best friend, so he decides to help Bob by rearranging the chocolate pieces. In one second, Charlie can pick up a piece in box i and put it into either box $i-1$ or box $i+1$ (if such boxes exist). Of course, he wants to help his friend as quickly as possible. Therefore, he asks you to calculate the minimum number of seconds he would need to make Alice happy.

Input

The first line contains a single integer n ($1 \le n \le 10^6$) — the number of chocolate boxes.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^6$) — the number of chocolate pieces in the i-th box.

It is guaranteed that at least one of $a_1, a_2, \ldots, a_n$ is positive.

Output

If there is no way for Charlie to make Alice happy, print -1.

Otherwise, print a single integer x — the minimum number of seconds for Charlie to help Bob make Alice happy.

Examples

input
```
3
4 8 5
```
output
```
9
```

input
```
5
3 10 2 1 5
```
output
```
2
```

input
```
4
0 5 15 10
```
output
```
0
```

input
```
1
1
```
output
```
-1
```

Note

In the first example, Charlie can move all chocolate pieces to the second box. Each box will be divisible by 17.

In the second example, Charlie can move a piece from box 2 to box 3 and a piece from box 4 to box 5. Each box will be divisible by 3.

In the third example, each box is already divisible by 5.

In the fourth example, since Charlie has no available move, he cannot help Bob make Alice happy.

C. Point Ordering

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is an interactive problem.

Khanh has n points on the Cartesian plane, denoted by $a_1, a_2, \ldots, a_n$. All points' coordinates are integers between -10^9 and 10^9, inclusive. No three points are collinear. He says that these points are vertices of a convex polygon; in other words, there exists a permutation $p_1, p_2, \ldots, p_n$ of integers from 1 to n such that the polygon $a_{p_1} a_{p_2} \ldots a_{p_n}$ is convex and vertices are listed in counter-clockwise order.

Khanh gives you the number n, but hides the coordinates of his points. Your task is to guess the above permutation by asking multiple queries. In each query, you give Khanh 4 integers t, i, j, k; where either $t = 1$ or $t = 2$; and i, j, k are three **distinct** indices from 1 to n, inclusive. In response, Khanh tells you:

- if $t = 1$, the area of the triangle $a_i a_j a_k$ **multiplied by** 2.
- if $t = 2$, the *sign* of the *cross product* of two **vectors** $\overrightarrow{a_i a_j}$ and $\overrightarrow{a_i a_k}$.

Recall that the *cross product* of vector $\overrightarrow{a} = (x_a, y_a)$ and vector $\overrightarrow{b} = (x_b, y_b)$ is the **integer** $x_a \cdot y_b - x_b \cdot y_a$. The *sign* of a number is 1 it it is positive, and -1 otherwise. It can be proven that the cross product obtained in the above queries can not be 0.

You can ask at most $3 \cdot n$ queries.

Please note that Khanh fixes the coordinates of his points and does not change it while answering your queries. You do not need to guess the coordinates. In your permutation $a_{p_1} a_{p_2} \ldots a_{p_n}$, p_1 should be equal to 1 and the indices of vertices should be **listed in**

counter-clockwise order.

Interaction

You start the interaction by reading n ($3 \le n \le 1\,000$) — the number of vertices.

To ask a query, write 4 integers t, i, j, k ($1 \le t \le 2$, $1 \le i, j, k \le n$) in a separate line. i, j and k should be distinct.

Then read a single integer to get the answer to this query, as explained above. It can be proven that the answer of a query is always an integer.

When you find the permutation, write a number 0. Then write n integers $p_1, p_2, \ldots, p_n$ in the same line.

After printing a query do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Hack format

To hack, use the following format:

The first line contains an integer n ($3 \le n \le 1\,000$) — the number of vertices.

The i-th of the next n lines contains two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) — the coordinate of the point a_i.

Example

input

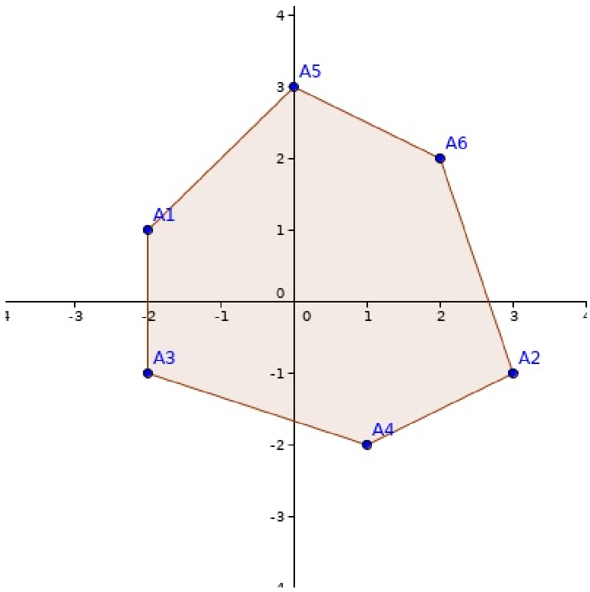
```
6
15
-1
1
```

output

```
1 1 4 6
2 1 5 6
2 2 1 4
0 1 3 4 2 6 5
```

Note

The image below shows the hidden polygon in the example:



The interaction in the example goes as below:

- Contestant reads $n = 6$.
- Contestant asks a query with $t = 1$, $i = 1$, $j = 4$, $k = 6$.
- Jury answers 15. The area of the triangle $A_1A_4A_6$ is 7.5. Note that the answer is **two times** the area of the triangle.
- Contestant asks a query with $t = 2$, $i = 1$, $j = 5$, $k = 6$.
- Jury answers -1. The cross product of $\overrightarrow{A_1A_5} = (2, 2)$ and $\overrightarrow{A_1A_6} = (4, 1)$ is -2. The sign of -2 is -1.

- Contestant asks a query with $t = 2$, $i = 2$, $j = 1$, $k = 4$.
- Jury answers 1. The cross product of $\overrightarrow{A_2A_1} = (-5, 2)$ and $\overrightarrow{A_2A_4} = (-2, -1)$ is 1. The sign of 1 is 1.
- Contestant says that the permutation is $(1, 3, 4, 2, 6, 5)$.

D. Tree Queries

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

Hanh is a famous biologist. He loves growing trees and doing experiments on his own garden.

One day, he got a tree consisting of n vertices. Vertices are numbered from 1 to n. A tree with n vertices is an undirected connected graph with $n - 1$ edges. Initially, Hanh sets the value of every vertex to 0.

Now, Hanh performs q operations, each is either of the following types:

- Type 1: Hanh selects a vertex v and an integer d. Then he chooses some vertex r **uniformly at random**, lists all vertices u such that the path from r to u passes through v. Hanh then increases the value of all such vertices u by d.
- Type 2: Hanh selects a vertex v and calculates the expected value of v.

Since Hanh is good at biology but not math, he needs your help on these operations.

Input

The first line contains two integers n and q ($1 \le n, q \le 150\,000$) — the number of vertices on Hanh's tree and the number of operations he performs.

Each of the next $n - 1$ lines contains two integers u and v ($1 \le u, v \le n$), denoting that there is an edge connecting two vertices u and v. It is guaranteed that these $n - 1$ edges form a tree.

Each of the last q lines describes an operation in either formats:

- $1\ v\ d$ ($1 \le v \le n, 0 \le d \le 10^7$), representing a first-type operation.
- $2\ v$ ($1 \le v \le n$), representing a second-type operation.

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

Output

For each operation of the second type, write the expected value on a single line.

Let $M = 998244353$, it can be shown that the expected value can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod M$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod M$.

Example

input

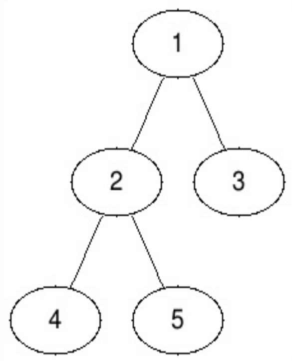
```
5 12
1 2
1 3
2 4
2 5
1 1 1
2 1
2 2
2 3
2 4
2 5
1 2 2
2 1
2 2
2 3
2 4
2 5
```

output

```
1
199648871
399297742
199648871
199648871
598946614
199648873
2
2
2
```

Note

The image below shows the tree in the example:



For the first query, where $v = 1$ and $d = 1$:

- If $r = 1$, the values of all vertices get increased.
- If $r = 2$, the values of vertices 1 and 3 get increased.
- If $r = 3$, the values of vertices 1, 2, 4 and 5 get increased.
- If $r = 4$, the values of vertices 1 and 3 get increased.
- If $r = 5$, the values of vertices 1 and 3 get increased.

Hence, the expected values of all vertices after this query are $(1, 0.4, 0.8, 0.4, 0.4)$.

For the second query, where $v = 2$ and $d = 2$:

- If $r = 1$, the values of vertices 2, 4 and 5 get increased.
- If $r = 2$, the values of all vertices get increased.
- If $r = 3$, the values of vertices 2, 4 and 5 get increased.
- If $r = 4$, the values of vertices 1, 2, 3 and 5 get increased.
- If $r = 5$, the values of vertices 1, 2, 3 and 4 get increased.

Hence, the expected values of all vertices after this query are $(2.2, 2.4, 2, 2, 2)$.

E. Send Tree to Charlie

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Christmas was knocking on the door, and our protagonist, Bob, was preparing a spectacular present for his long-time second best friend Charlie. As chocolate boxes are lame, he decided to decorate a tree instead. Bob's tree can be represented as an undirected connected graph with n nodes (numbered 1 to n) and $n - 1$ edges. Initially, Bob placed a decoration with label i on node i, for each $1 \le i \le n$. However, as such a simple arrangement is lame, he decided to shuffle the decorations a bit. Formally, Bob did the following steps:

- First, he listed the $n - 1$ edges in some order.
- Then, he considered the edges one by one in that order. For each edge (u, v), he swapped the decorations of node u with the one of node v.

After finishing, Bob seemed satisfied with the arrangement, so he went to sleep.

The next morning, Bob wakes up only to find out that his beautiful arrangement has been ruined! Last night, Bob's younger brother Bobo dropped some of the decorations on the floor while he was playing with the tree. Fortunately, no decorations were lost, so Bob can repair the tree in no time. However, he completely forgets how the tree looked like yesterday. Therefore, given the labels of the decorations still on the tree, Bob wants to know the number of possible configurations of the tree. As the result can be quite large, Bob will be happy if you can output the result modulo 1000000007 ($10^9 + 7$). Note that, it is possible that there exists no possible configurations.

Input

The first line contains a single integer n ($2 \le n \le 500\,000$) — the number of nodes.

Each of the next $n - 1$ lines contains two integers u and v ($1 \le u, v \le n$), denoting that there is an edge connecting two nodes u and v. It is guaranteed that the given edges form a tree.

The last line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$). For each i, $a_i = 0$ means that the decoration of node i has been dropped on the floor. Otherwise, a_i is the label of the decoration of node i. It is guaranteed that no label appears more than once.

Output

Output the number of possible configurations modulo 1000000007 ($10^9 + 7$).

Examples

input
4

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

output

```
2
```

input

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

output

```
12
```

input

```
3
1 2
1 3
1 0 0
```

output

```
0
```

Note

In the first example, the possible configurations of the tree are $[2, 4, 1, 3]$ and $[3, 4, 2, 1]$.

In the second example, note that while there are $4! = 24$ possible permutations of the edges, each of them results in a possible configuration, there are only 12 different configurations.

In the third example, it is easy to see that the decoration 1 cannot stay at node 1 after the swaps.