Codeforces Round #601 (Div. 2)

A. Changing Volume

1 second, 256 megabytes

Bob watches TV every day. He always sets the volume of his TV to b. However, today he is angry to find out someone has changed the volume to a. Of course, Bob has a remote control that can change the volume.

There are six buttons ($-5, -2, -1, +1, +2, +5$) on the control, which in one press can either increase or decrease the current volume by 1, 2, or 5. The volume can be arbitrarily large, but can never be negative. In other words, Bob cannot press the button if it causes the volume to be lower than 0.

As Bob is so angry, he wants to change the volume to b using as few button presses as possible. However, he forgets how to do such simple calculations, so he asks you for help. Write a program that given a and b, finds the minimum number of presses to change the TV volume from a to b.

Input

Each test contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 1\,000$). Then the descriptions of the test cases follow.

Each test case consists of one line containing two integers a and b ($0 \le a, b \le 10^9$) — the current volume and Bob's desired volume, respectively.

Output

For each test case, output a single integer — the minimum number of presses to change the TV volume from a to b. If Bob does not need to change the volume (i.e. $a = b$), then print 0.

input
3 4 0 5 14 3 9
output
2 3 2

In the first example, Bob can press the -2 button twice to reach 0. Note that Bob can not press -5 when the volume is 4 since it will make the volume negative.

In the second example, one of the optimal ways for Bob is to press the $+5$ twice, then press -1 once.

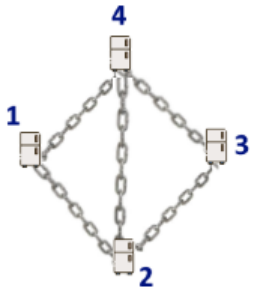
In the last example, Bob can press the $+5$ once, then press $+1$.

B. Fridge Lockers

1 second, 256 megabytes

Hanh lives in a shared apartment. There are n people (including Hanh) living there, each has a private fridge.

n fridges are secured by several steel chains. Each steel chain connects two **different** fridges and is protected by a digital lock. The owner of a fridge knows passcodes of all chains connected to it. A fridge can be open only if all chains connected to it are unlocked. For example, if a fridge has no chains connected to it at all, then any of n people can open it.



For exampe, in the picture there are $n = 4$ people and 5 chains. The first person knows passcodes of two chains: $1 - 4$ and $1 - 2$. The fridge 1 can be open by its owner (the person 1), also two people 2 and 4 (acting together) can open it.

The weights of these fridges are $a_1, a_2, \ldots, a_n$. To make a steel chain connecting fridges u and v, you have to pay $a_u + a_v$ dollars. Note that the landlord allows you to create **multiple chains connecting the same pair of fridges**.

Hanh's apartment landlord asks you to create exactly m steel chains so that all fridges are private. A fridge is private if and only if, among n people living in the apartment, only the owner can open it (i.e. no other person acting alone can do it). In other words, the fridge i is not private if there exists the person j ($i \neq j$) that the person j can open the fridge i.

For example, in the picture all the fridges are private. On the other hand, if there are $n = 2$ fridges and only one chain (which connects them) then both fridges are not private (both fridges can be open not only by its owner but also by another person).

Of course, the landlord wants to minimize the total cost of all steel chains to fulfill his request. Determine whether there exists any way to make exactly m chains, and if yes, output any solution that minimizes the total cost.

Input

Each test contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 10$). Then the descriptions of the test cases follow.

The first line of each test case contains two integers n, m ($2 \le n \le 1000$, $1 \le m \le n$) — the number of people living in Hanh's apartment and the number of steel chains that the landlord requires, respectively.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^4$) — weights of all fridges.

Output

For each test case:

- If there is no solution, print a single integer -1.
- Otherwise, print a single integer c — the minimum total cost. The i-th of the next m lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \neq v_i$), meaning that the i-th steel chain connects fridges u_i and v_i. An arbitrary number of chains can be between a pair of fridges.

If there are multiple answers, print any.

input
3 4 4 1 1 1 1 3 1 1 2 3 3 3 1 2 3

output

```
8
1 2
4 3
3 2
4 1
-1
12
3 2
1 2
3 1
```

C. League of Leesins

2 seconds, 256 megabytes

Bob is an avid fan of the video game "League of Leesins", and today he celebrates as the League of Leesins World Championship comes to an end!

The tournament consisted of n ($n \geq 5$) teams around the world. Before the tournament starts, Bob has made a prediction of the rankings of each team, from 1-st to n-th. After the final, he compared the prediction with the actual result and found out that the i-th team according to his prediction ended up at the p_i-th position ($1 \leq p_i \leq n$, all p_i are unique). In other words, p is a permutation of $1, 2, \ldots, n$.

As Bob's favorite League player is the famous "3ga", he decided to write down every 3 consecutive elements of the permutation p. Formally, Bob created an array q of $n-2$ triples, where $q_i = (p_i, p_{i+1}, p_{i+2})$ for each $1 \leq i \leq n-2$. Bob was very proud of his array, so he showed it to his friend Alice.

After learning of Bob's array, Alice declared that she could retrieve the permutation p even if Bob rearranges the elements of q and the elements within each triple. Of course, Bob did not believe in such magic, so he did just the same as above to see Alice's respond.

For example, if $n = 5$ and $p = [1, 4, 2, 3, 5]$, then the original array q will be $[(1, 4, 2), (4, 2, 3), (2, 3, 5)]$. Bob can then rearrange the numbers within each triple and the positions of the triples to get $[(4, 3, 2), (2, 3, 5), (4, 1, 2)]$. Note that $[(1, 4, 2), (4, 2, 2), (3, 3, 5)]$ is not a valid rearrangement of q, as Bob is not allowed to swap numbers belong to different triples.

As Alice's friend, you know for sure that Alice was just trying to show off, so you decided to save her some face by giving her **any permutation** p that is consistent with the array q she was given.

Input

The first line contains a single integer n ($5 \leq n \leq 10^5$) — the size of permutation p.

The i-th of the next $n-2$ lines contains 3 integers $q_{i,1}, q_{i,2}, q_{i,3}$ ($1 \leq q_{i,j} \leq n$) — the elements of the i-th triple of the **rearranged** (shuffled) array q_i, in random order. Remember, that the numbers within each triple can be rearranged and also the positions of the triples can be rearranged.

It is guaranteed that there is at least one permutation p that is consistent with the input.

Output

Print n distinct integers $p_1, p_2, \ldots, p_n$ ($1 \leq p_i \leq n$) such that p is consistent with array q.

If there are multiple answers, print any.

input

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

output

```
1 4 2 3 5
```

D. Feeding Chicken

1.5 seconds, 256 megabytes

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 ($1 \leq T \leq 2 \cdot 10^4$). Description of the test cases follows.

The first line of each test case contains three integers r, c and k ($1 \leq r, c \leq 100, 1 \leq k \leq 62$), 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.

input

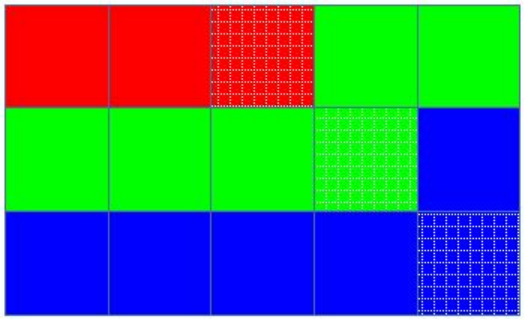
```
4
3 5 3
..R..
...R.
....R
6 4 6
R..R
R..R
RRRR
RRRR
R..R
R..R
5 5 4
RRR..
R.R..
RRR..
R..R.
R...R
2 31 62
RRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
RRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
```

output

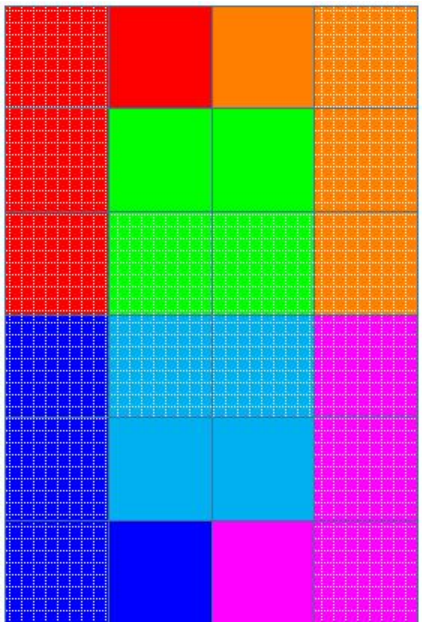
```
11122
22223
33333
aacc
aBBc
aBBc
CbbA
CbbA
CCAA
11114
22244
32444
33344
33334
abcdefghijklmnopqrstuvwxyzABCDE
FGHIJKLMNOPQRSTUVWXYZ0123456789
```

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

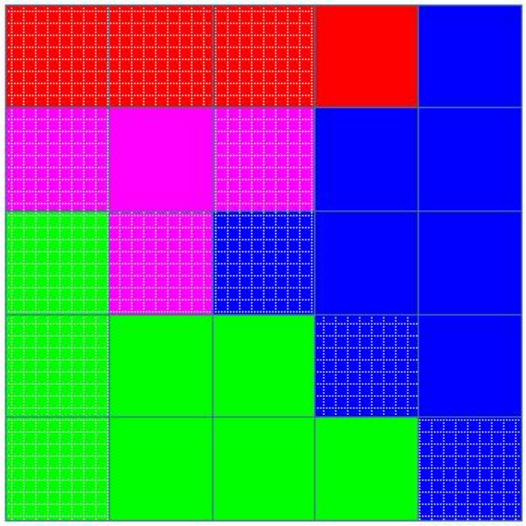
In the first test case, each chicken has one cell with rice. Hence, the difference between the maximum and the minimum number of cells with rice assigned to a chicken is 0.



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.

E1. Send Boxes to Alice (Easy Version)

1 second, 256 megabytes

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.

input

```
3
1 0 1
```

output

```
2
```

input

```
1
1
```

output
-1

E2. Send Boxes to Alice (Hard Version)

1 second, 256 megabytes

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.

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

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.

F. Point Ordering

1 second, 256 megabytes

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

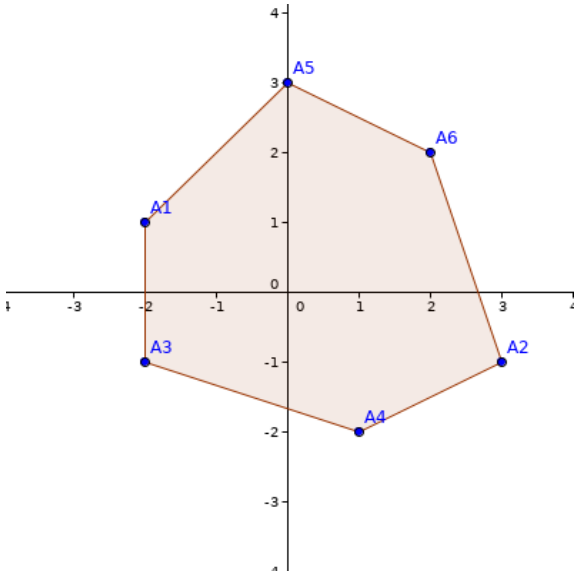
To hack, use the following format:

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.

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

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 $(1, 3, 4, 2, 6, 5)$.