Codeforces Round #592 (Div. 2)

A. Pens and Pencils

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

Tomorrow is a difficult day for Polycarp: he has to attend a lectures and b practical classes at the university! Since Polycarp is a diligent student, he is going to attend all of them.

While preparing for the university, Polycarp wonders whether he can take enough writing implements to write all of the lectures and draw everything he has to during all of the practical classes. Polycarp writes lectures using a pen (he can't use a pencil to write lectures!); he can write down c lectures using one pen, and after that it runs out of ink. During practical classes Polycarp draws blueprints with a pencil (he can't use a pen to draw blueprints!); one pencil is enough to draw all blueprints during d practical classes, after which it is unusable.

Polycarp's pencilcase can hold no more than k writing implements, so if Polycarp wants to take x pens and y pencils, they will fit in the pencilcase if and only if $x + y \leq k$.

Now Polycarp wants to know how many pens and pencils should he take. Help him to determine it, or tell that his pencilcase doesn't have enough room for all the implements he needs tomorrow!

Note that you don't have to minimize the number of writing implements (though their total number must not exceed k).

Input

The first line of the input contains one integer t ($1 \leq t \leq 100$) — the number of test cases in the input. Then the test cases follow.

Each test case is described by one line containing five integers a, b, c, d and k, separated by spaces ($1 \leq a, b, c, d, k \leq 100$) — the number of lectures Polycarp has to attend, the number of practical classes Polycarp has to attend, the number of lectures which can be written down using one pen, the number of practical classes for which one pencil is enough, and the number of writing implements that can fit into Polycarp's pencilcase, respectively.

In hacks it is allowed to use only one test case in the input, so $t = 1$ should be satisfied.

Output

For each test case, print the answer as follows:

If the pencilcase can't hold enough writing implements to use them during all lectures and practical classes, print one integer -1. Otherwise, print two non-negative integers x and y — the number of pens and pencils Polycarp should put in his pencilcase. If there are multiple answers, print any of them. Note that you don't have to minimize the number of writing implements (though their total number must not exceed k).

Example

input

```
3
7 5 4 5 8
7 5 4 5 2
20 53 45 26 4
```

output

```
7 1
-1
1 3
```

Note

There are many different answers for the first test case; $x = 7$, $y = 1$ is only one of them. For example, $x = 3$, $y = 1$ is also correct.

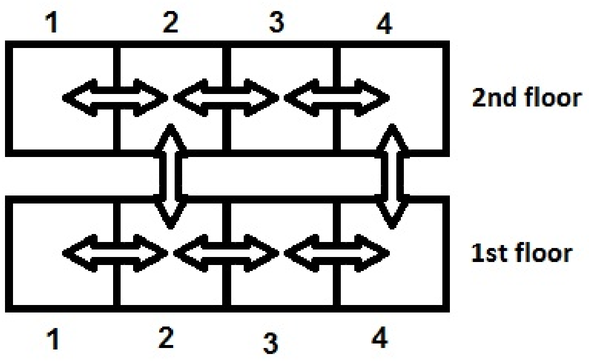
$x = 1$, $y = 3$ is the only correct answer for the third test case.

B. Rooms and Staircases

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

Nikolay lives in a two-storied house. There are n rooms on each floor, arranged in a row and numbered from one from left to right. So each room can be represented by the number of the floor and the number of the room on this floor (room number is an integer between 1 and n).

If Nikolay is currently in some room, he can move to any of the neighbouring rooms (if they exist). Rooms with numbers i and $i+1$ on each floor are neighbouring, for all $1 \leq i \leq n-1$. There may also be staircases that connect two rooms from different floors having the same numbers. If there is a staircase connecting the room x on the first floor and the room x on the second floor, then Nikolay can use it to move from one room to another.



The picture illustrates a house with $n = 4$. There is a staircase between the room 2 on the first floor and the room 2 on the second floor, and another staircase between the room 4 on the first floor and the room 4 on the second floor. The arrows denote possible directions in which Nikolay can move. The picture corresponds to the string "`0101`" in the input.

Nikolay wants to move through some rooms in his house. To do this, he firstly chooses any room where he starts. Then Nikolay moves between rooms according to the aforementioned rules. Nikolay never visits the same room twice (he won't enter a room where he has already been).

Calculate the maximum number of rooms Nikolay can visit during his tour, if:

- he can start **in any room on any floor of his choice**,
- and **he won't visit the same room twice**.

Input

The first line of the input contains one integer t $(1 \leq t \leq 100)$ — the number of test cases in the input. Then test cases follow. Each test case consists of two lines.

The first line contains one integer n $(1 \leq n \leq 1\,000)$ — the number of rooms on each floor.

The second line contains one string consisting of n characters, each character is either a '`0`' or a '`1`'. If the i-th character is a '`1`', then there is a staircase between the room i on the first floor and the room i on the second floor. If the i-th character is a '`0`', then there is no staircase between the room i on the first floor and the room i on the second floor.

In hacks it is allowed to use only one test case in the input, so $t = 1$ should be satisfied.

Output

For each test case print one integer — the maximum number of rooms Nikolay can visit during his tour, if he can start in any room on any floor, and he won't visit the same room twice.

Example

input

```
4
5
00100
8
00000000
5
11111
3
110
```

output

```
6
8
10
6
```

Note

In the first test case Nikolay may start in the first room of the first floor. Then he moves to the second room on the first floor, and then — to the third room on the first floor. Then he uses a staircase to get to the third room on the second floor. Then he goes to the fourth room on the second floor, and then — to the fifth room on the second floor. So, Nikolay visits 6 rooms.

There are no staircases in the second test case, so Nikolay can only visit all rooms on the same floor (if he starts in the leftmost or in the rightmost room).

In the third test case it is possible to visit all rooms: first floor, first room $\rightarrow$ second floor, first room $\rightarrow$ second floor, second room $\rightarrow$ first floor, second room $\rightarrow$ first floor, third room $\rightarrow$ second floor, third room $\rightarrow$ second floor, fourth room $\rightarrow$ first floor, fourth room $\rightarrow$ first floor, fifth room $\rightarrow$ second floor, fifth room.

In the fourth test case it is also possible to visit all rooms: second floor, third room $\rightarrow$ second floor, second room $\rightarrow$ second floor, first room $\longrightarrow$ first floor, first room $\longrightarrow$ first floor, second room $\longrightarrow$ first floor, third room.

C. The Football Season

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

The football season has just ended in Berland. According to the rules of Berland football, each match is played between two teams. The result of each match is either a draw, or a victory of one of the playing teams. If a team wins the match, it gets w points, and the opposing team gets 0 points. If the game results in a draw, both teams get d points.

The manager of the Berland capital team wants to summarize the results of the season, but, unfortunately, all information about the results of each match is lost. The manager only knows that the team has played n games and got p points for them.

You have to determine three integers x, y and z — the number of wins, draws and loses of the team. If there are multiple answers, print any of them. If there is no suitable triple (x, y, z), report about it.

Input

The first line contains four integers n, p, w and d $(1 \le n \le 10^{12}, 0 \le p \le 10^{17}, 1 \le d < w \le 10^5)$ — the number of games, the number of points the team got, the number of points awarded for winning a match, and the number of points awarded for a draw, respectively. Note that $w > d$, so the number of points awarded for winning is strictly greater than the number of points awarded for draw.

Output

If there is no answer, print -1.

Otherwise print three non-negative integers x, y and z — the number of wins, draws and losses of the team. If there are multiple possible triples (x, y, z), print any of them. The numbers should meet the following conditions:

- $x \cdot w + y \cdot d = p$,
- $x + y + z = n$.

Examples

input
```
30 60 3 1
```
output
```
17 9 4
```

input
```
10 51 5 4
```
output
```
-1
```

input
```
20 0 15 5
```
output
```
0 0 20
```

Note

One of the possible answers in the first example — 17 wins, 9 draws and 4 losses. Then the team got $17 \cdot 3 + 9 \cdot 1 = 60$ points in $17 + 9 + 4 = 30$ games.

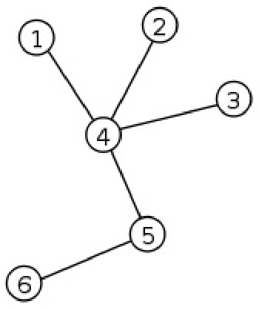
In the second example the maximum possible score is $10 \cdot 5 = 50$. Since $p = 51$, there is no answer.

In the third example the team got 0 points, so all 20 games were lost.

D. Paint the Tree

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a tree consisting of n vertices. A tree is an undirected connected acyclic graph.



Example of a tree.

You have to paint each vertex into one of three colors. For each vertex, you know the cost of painting it in every color.

You have to paint the vertices so that any path consisting of exactly three distinct vertices does not contain any vertices with equal colors. In other words, let's consider all triples (x, y, z) such that $x \neq y, y \neq z, x \neq z$, x is connected by an edge with y, and y is connected by an edge with z. The colours of x, y and z should be pairwise distinct. Let's call a painting which meets this condition *good*.

You have to calculate the minimum cost of a *good* painting and find one of the optimal paintings. If there is no *good* painting, report about it.

Input

The first line contains one integer n $(3 \le n \le 100\,000)$ — the number of vertices.

The second line contains a sequence of integers $c_{1,1}, c_{1,2}, \dots, c_{1,n}$ $(1 \le c_{1,i} \le 10^9)$, where $c_{1,i}$ is the cost of painting the i-th vertex into the first color.

The third line contains a sequence of integers $c_{2,1}, c_{2,2}, \dots, c_{2,n}$ $(1 \le c_{2,i} \le 10^9)$, where $c_{2,i}$ is the cost of painting the i-th vertex into the second color.

The fourth line contains a sequence of integers $c_{3,1}, c_{3,2}, \dots, c_{3,n}$ $(1 \le c_{3,i} \le 10^9)$, where $c_{3,i}$ is the cost of painting the i-th vertex into the third color.

Then $(n-1)$ lines follow, each containing two integers u_j and v_j $(1 \le u_j, v_j \le n, u_j \neq v_j)$ — the numbers of vertices connected by the j-th undirected edge. It is guaranteed that these edges denote a tree.

Output

If there is no *good* painting, print -1.

Otherwise, print the minimum cost of a *good* painting in the first line. In the second line print n integers $b_1, b_2, \dots, b_n$ $(1 \le b_i \le 3)$, where the i-th integer should denote the color of the i-th vertex. If there are multiple good paintings with minimum cost, print any of them.

Examples

input
```
3
3 2 3
4 3 2
3 1 3
1 2
2 3
```
output
```
6
1 3 2
```

input
```
5
3 4 2 1 2
4 2 1 5 4
5 3 2 1 1
1 2
3 2
4 3
5 3
```
output
```
-1
```

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

output

```
9
1 3 2 1 3
```

Note

All vertices should be painted in different colors in the first example. The optimal way to do it is to paint the first vertex into color 1, the second vertex — into color 3, and the third vertex — into color 2. The cost of this painting is $3+2+1=6$.

E. Minimizing Difference

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a sequence $a_1, a_2, \dots, a_n$ consisting of n integers.

You may perform the following operation on this sequence: choose any element and either increase or decrease it by one.

Calculate the minimum possible difference between the maximum element and the minimum element in the sequence, if you can perform the aforementioned operation **no more than** k times.

Input

The first line contains two integers n and k $(2 \le n \le 10^5, 1 \le k \le 10^{14})$ — the number of elements in the sequence and the maximum number of times you can perform the operation, respectively.

The second line contains a sequence of integers $a_1, a_2, \dots, a_n$ $(1 \le a_i \le 10^9)$.

Output

Print the minimum possible difference between the maximum element and the minimum element in the sequence, if you can perform the aforementioned operation **no more than** k times.

Examples

input

```
4 5
3 1 7 5
```

output

```
2
```

input

```
3 10
100 100 100
```

output

```
0
```

input

```
10 9
4 5 5 7 5 4 5 2 4 3
```

output

```
1
```

Note

In the first example you can increase the first element twice and decrease the third element twice, so the sequence becomes $[3,3,5,5]$, and the difference between maximum and minimum is 2. You still can perform one operation after that, but it's useless since you can't make the answer less than 2.

In the second example all elements are already equal, so you may get 0 as the answer even without applying any operations.

F. Chips

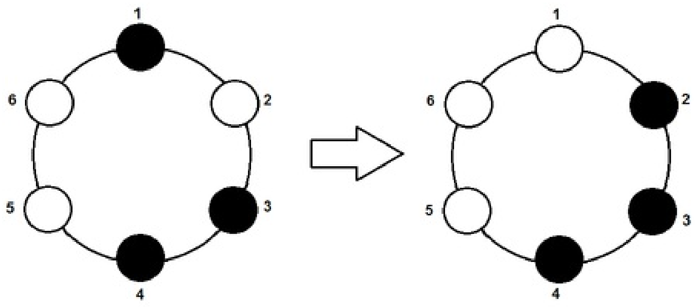
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n chips arranged in a circle, numbered from 1 to n.

Initially each chip has black or white color. Then k iterations occur. During each iteration the chips change their colors according to the following rules. For each chip i, three chips are considered: chip i itself and two its neighbours. If the number of white chips among these three is greater than the number of black chips among these three chips, then the chip i becomes white. Otherwise, the chip i becomes black.

Note that for each i from 2 to $(n-1)$ two neighbouring chips have numbers $(i-1)$ and $(i+1)$. The neighbours for the chip $i=1$ are n and 2. The neighbours of $i=n$ are $(n-1)$ and 1.

The following picture describes one iteration with $n=6$. The chips 1, 3 and 4 are initially black, and the chips 2, 5 and 6 are white. After the iteration 2, 3 and 4 become black, and 1, 5 and 6 become white.



Your task is to determine the color of each chip after k iterations.

Input

The first line contains two integers n and k $(3 \le n \le 200\,000, 1 \le k \le 10^9)$ — the number of chips and the number of iterations, respectively.

The second line contains a string consisting of n characters "W" and "B". If the i-th character is "W", then the i-th chip is white initially. If the i-th character is "B", then the i-th chip is black initially.

Output

Print a string consisting of n characters "W" and "B". If after k iterations the i-th chip is white, then the i-th character should be "W". Otherwise the i-th character should be "B".

Examples

input

```
6 1
BWBBWW
```

output

```
WBBBWW
```

input

```
7 3
WBWBWBW
```

output

```
WWWWWWW
```

input

```
6 4
BWBWBW
```

output

```
BWBWBW
```

Note

The first example is described in the statement.

The second example: "WBWBWBW" $\rightarrow$ "WWBWBWW" $\rightarrow$ "WWWBWWW" $\rightarrow$ "WWWWWWW". So all chips become white.

The third example: "BWBWBW" $\rightarrow$ "WBWBWB" $\rightarrow$ "BWBWBW" $\rightarrow$ "WBWBWB" $\rightarrow$ "BWBWBW".

G. Running in Pairs

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

Demonstrative competitions will be held in the run-up to the $20NN$ Berlatov Olympic Games. Today is the day for the running competition!

Berlatov team consists of $2n$ runners which are placed on two running tracks; n runners are placed on each track. The runners are numbered from 1 to n on each track. The runner with number i runs through the entire track in i seconds.

The competition is held as follows: first runners on both tracks start running at the same time; when the slower of them arrives at the end of the track, second runners on both tracks start running, and everyone waits until the slower of them finishes running, and so on, until all n pairs run through the track.

The organizers want the run to be as long as possible, but if it lasts for more than k seconds, the crowd will get bored. As the coach of the team, you may choose any order in which the runners are arranged on each track (but you can't change the number of runners on each track or swap runners between different tracks).

You have to choose the order of runners on each track so that the duration of the competition is as long as possible, but does not exceed k seconds.

Formally, you want to find two permutations p and q (both consisting of n elements) such that $sum = \sum\limits_{i=1}^{n} max(p_i, q_i)$ is maximum possible, but does not exceed k. If there is no such pair, report about it.

Input

The first line contains two integers n and k ($1 \le n \le 10^6, 1 \le k \le n^2$) — the number of runners on each track and the maximum possible duration of the competition, respectively.

Output

If it is impossible to reorder the runners so that the duration of the competition does not exceed k seconds, print -1.

Otherwise, print three lines. The first line should contain one integer sum — the maximum possible duration of the competition not exceeding k. The second line should contain a permutation of n integers $p_1, p_2, \dots, p_n$ ($1 \le p_i \le n$, all p_i should be pairwise distinct) — the numbers of runners on the first track in the order they participate in the competition. The third line should contain a permutation of n integers $q_1, q_2, \dots, q_n$ ($1 \le q_i \le n$, all q_i should be pairwise distinct) — the numbers of runners on the second track in the order they participate in the competition. The value of $sum = \sum\limits_{i=1}^{n} max(p_i, q_i)$ should be maximum possible, but should not exceed k. If there are multiple answers, print any of them.

Examples

input

```
5 20
```

output

```
20
1 2 3 4 5
5 2 4 3 1
```

input

```
3 9
```

output

```
8
1 2 3
3 2 1
```

input

```
10 54
```

output

```
-1
```

Note

In the first example the order of runners on the first track should be $[5, 3, 2, 1, 4]$, and the order of runners on the second track should be $[1, 4, 2, 5, 3]$. Then the duration of the competition is $max(5, 1) + max(3, 4) + max(2, 2) + max(1, 5) + max(4, 3) = 5 + 4 + 2 + 5 + 4 = 20$, so it is equal to the maximum allowed duration.

In the first example the order of runners on the first track should be $[2, 3, 1]$, and the order of runners on the second track should be $[2, 1, 3]$. Then the duration of the competition is 8, and it is the maximum possible duration for $n = 3$.