Codeforces Round #386 (Div. 2)

A. Compote

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

Nikolay has a lemons, b apples and c pears. He decided to cook a compote. According to the recipe the fruits should be in the ratio $1: 2: 4$. It means that for each lemon in the compote should be exactly 2 apples and exactly 4 pears. You can't crumble up, break up or cut these fruits into pieces. These fruits — lemons, apples and pears — should be put in the compote as whole fruits.

Your task is to determine the maximum total number of lemons, apples and pears from which Nikolay can cook the compote. It is possible that Nikolay can't use any fruits, in this case print 0.

Input

The first line contains the positive integer a ($1 \le a \le 1000$) — the number of lemons Nikolay has.

The second line contains the positive integer b ($1 \le b \le 1000$) — the number of apples Nikolay has.

The third line contains the positive integer c ($1 \le c \le 1000$) — the number of pears Nikolay has.

Output

Print the maximum total number of lemons, apples and pears from which Nikolay can cook the compote.

Examples

input
2 5 7
output
7

input
4 7 13
output
21

input
2 3 2
output
0

Note

In the first example Nikolay can use 1 lemon, 2 apples and 4 pears, so the answer is $1 + 2 + 4 = 7$.

In the second example Nikolay can use 3 lemons, 6 apples and 12 pears, so the answer is $3 + 6 + 12 = 21$.

In the third example Nikolay don't have enough pears to cook any compote, so the answer is 0.

B. Decoding

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

Polycarp is mad about coding, that is why he writes Sveta encoded messages. He calls the *median letter* in a word the letter which is in the middle of the word. If the word's length is even, the median letter is the left of the two middle letters. In the following examples, the median letter is highlighted: con**t**est, i**n**fo. If the word consists of single letter, then according to above definition this letter is the median letter.

Polycarp encodes each word in the following way: he writes down the median letter of the word, then deletes it and repeats the process until there are no letters left. For example, he encodes the word volga as logva.

You are given an encoding s of some word, your task is to decode it.

Input

The first line contains a positive integer n ($1 \le n \le 2000$) — the length of the encoded word.

The second line contains the string s of length n consisting of lowercase English letters — the encoding.

Output

Print the word that Polycarp encoded.

Examples

input
5 logva
output
volga

input
2 no
output
no

input
4 abba
output
baba

Note

In the first example Polycarp encoded the word volga. At first, he wrote down the letter l from the position 3, after that his word looked like voga. After that Polycarp wrote down the letter o from the position 2, his word became vga. Then Polycarp wrote down the letter g which was at the second position, the word became va. Then he wrote down the letter v, then the letter a. Thus, the encoding looked like logva.

In the second example Polycarp encoded the word no. He wrote down the letter n, the word became o, and he wrote down the letter o. Thus, in this example, the word and its encoding are the same.

In the third example Polycarp encoded the word baba. At first, he wrote down the letter a, which was at the position 2, after that the word looked like bba. Then he wrote down the letter b, which was at the position 2, his word looked like ba. After that he wrote down the letter b, which was at the position 1, the word looked like a, and he wrote down that letter a. Thus, the encoding is abba.

C. Tram

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

The tram in Berland goes along a straight line from the point 0 to the point s and back, passing 1 meter per t_1 seconds in both directions. It means that the tram is always in the state of uniform rectilinear motion, instantly turning around at points $x = 0$ and $x = s$.

Igor is at the point x_1. He should reach the point x_2. Igor passes 1 meter per t_2 seconds.

Your task is to determine the minimum time Igor needs to get from the point x_1 to the point x_2, if it is known where the tram is and in what direction it goes at the moment Igor comes to the point x_1.

Igor can enter the tram unlimited number of times at any moment when his and the tram's positions coincide. It is **not obligatory** that points in which Igor enter and exit the tram are integers. Assume that any boarding and unboarding happens instantly. Igor can move arbitrary along the line (but not faster than 1 meter per t_2 seconds). He can also stand at some point for some time.

Input

The first line contains three integers s, x_1 and x_2 ($2 \le s \le 1000$, $0 \le x_1, x_2 \le s$, $x_1 \ne x_2$) — the maximum coordinate of the point to which the tram goes, the point Igor is at, and the point he should come to.

The second line contains two integers t_1 and t_2 ($1 \le t_1, t_2 \le 1000$) — the time in seconds in which the tram passes 1 meter and the time in seconds in which Igor passes 1 meter.

The third line contains two integers p and d ($1 \le p \le s$ - 1, d is either 1 or) — the position of the tram in the moment Igor came to the point x_1 and the direction of the tram at this moment. If , the tram goes in the direction from the point s to the point 0. If $d = 1$, the tram goes in the direction from the point 0 to the point s.

Output

Print the minimum time in seconds which Igor needs to get from the point x_1 to the point x_2.

Examples

input
4 2 4 3 4 1 1

output
8

input
5 4 0 1 2 3 1

output
7

Note

In the first example it is profitable for Igor to go by foot and not to wait the tram. Thus, he has to pass 2 meters and it takes 8 seconds in total, because he passes 1 meter per 4 seconds.

In the second example Igor can, for example, go towards the point x_2 and get to the point 1 in 6 seconds (because he has to pass 3 meters, but he passes 1 meters per 2 seconds). At that moment the tram will be at the point 1, so Igor can enter the tram and pass 1 meter in 1 second. Thus, Igor will reach the point x_2 in 7 seconds in total.

D. Green and Black Tea

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

Innokentiy likes tea very much and today he wants to drink exactly n cups of tea. He would be happy to drink more but he had exactly n tea bags, a of them are green and b are black.

Innokentiy doesn't like to drink the same tea (green or black) more than k times in a row. Your task is to determine the order of brewing tea bags so that Innokentiy will be able to drink n cups of tea, without drinking the same tea more than k times in a row, or to inform that it is impossible. Each tea bag has to be used exactly once.

Input

The first line contains four integers n, k, a and b ($1 \le k \le n \le 10^5$, $0 \le a, b \le n$) — the number of cups of tea Innokentiy wants to drink, the maximum number of cups of same tea he can drink in a row, the number of tea bags of green and black tea. It is guaranteed that $a + b = n$.

Output

If it is impossible to drink n cups of tea, print "NO" (without quotes).

Otherwise, print the string of the length n, which consists of characters 'G' and 'B'. If some character equals 'G', then the corresponding cup of tea should be green. If some character equals 'B', then the corresponding cup of tea should be black.

If there are multiple answers, print any of them.

Examples

input
5 1 3 2
output
GBGBG

input
7 2 2 5
output
BBGBGBB

input
4 3 4 0
output
NO

E. Numbers Exchange

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

Eugeny has n cards, each of them has exactly one integer written on it. Eugeny wants to exchange some cards with Nikolay so that the number of even integers on his cards would equal the number of odd integers, and that all these numbers would be **distinct**.

Nikolay has m cards, distinct numbers from 1 to m are written on them, one per card. It means that Nikolay has exactly one card with number 1, exactly one card with number 2 and so on.

A single exchange is a process in which Eugeny gives one card to Nikolay and takes another one from those Nikolay has. Your task is to find the minimum number of card exchanges and determine which cards Eugeny should exchange.

Input

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 10^9$) — the number of cards Eugeny has and the number of cards Nikolay has. It is guaranteed that n is even.

The second line contains a sequence of n positive integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$) — the numbers on Eugeny's cards.

Output

If there is no answer, print `-1`.

Otherwise, in the first line print the minimum number of exchanges. In the second line print n integers — Eugeny's cards after all the exchanges with Nikolay. The order of cards should coincide with the card's order in the input data. If the i-th card wasn't exchanged then the i-th number should coincide with the number from the input data. Otherwise, it is considered that this card was exchanged, and the i-th number should be equal to the number on the card it was exchanged to.

If there are multiple answers, it is allowed to print any of them.

Examples

input
6 2 5 6 7 9 4 5
output
1 5 6 7 9 4 2

input
8 6 7 7 7 7 8 8 8 8
output
6 7 2 4 6 8 1 3 5

input
4 1 4 2 1 10
output
-1

F. Music in Car

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

Sasha reaches the work by car. It takes exactly k minutes. On his way he listens to music. All songs in his playlist go one by one, after listening to the i-th song Sasha gets a pleasure which equals a_i. The i-th song lasts for t_i minutes.

Before the beginning of his way Sasha turns on some song x and then he listens to the songs one by one: at first, the song x, then the song $(x + 1)$, then the song number $(x + 2)$, and so on. He listens to songs until he reaches the work or until he listens to the last song in his playlist.

Sasha can listen to each song to the end or *partly*.

In the second case he listens to the song for integer number of minutes, at least half of the song's length. Formally, if the length of the song equals d minutes, Sasha listens to it for no less than minutes, then he immediately switches it to the next song (if there is such). For example, if the length of the song which Sasha wants to *partly* listen to, equals 5 minutes, then he should listen to it for at least 3 minutes, if the length of the song equals 8 minutes, then he should listen to it for at least 4 minutes.

It takes no time to switch a song.

Sasha wants to listen *partly* no more than w songs. If the last listened song plays for less than half of its length, then Sasha doesn't get pleasure from it and that song is not included to the list of *partly* listened songs. It is not allowed to skip songs. A pleasure from a song does not depend on the listening mode, for the i-th song this value equals a_i.

Help Sasha to choose such x and no more than w songs for *partial* listening to get the maximum pleasure. Write a program to find the maximum pleasure Sasha can get from the listening to the songs on his way to the work.

Input

The first line contains three integers n, w and k ($1 \le w \le n \le 2 \cdot 10^5$, $1 \le k \le 2 \cdot 10^9$) — the number of songs in the playlist, the number of songs Sasha can listen to *partly* and time in minutes which Sasha needs to reach work.

The second line contains n positive integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^4$), where a_i equals the pleasure Sasha gets after listening to the i-th song.

The third line contains n positive integers $t_1, t_2, ..., t_n$ ($2 \le t_i \le 10^4$), where t_i equals the length of the i-th song in minutes.

Output

Print the maximum pleasure Sasha can get after listening to the songs on the way to work.

Examples

input
7 2 11 3 4 3 5 1 4 6 7 7 3 6 5 3 9
output
12

input
8 4 20 5 6 4 3 7 5 4 1 10 12 5 12 14 8 5 8
output
19

input
1 1 5 6 9
output
6

input
1 1 3 4 7
output
0

Note

In the first example Sasha needs to start listening from the song number 2. He should listen to it *partly* (for 4 minutes), then listen to the song number

3 to the end (for 3 minutes) and then *partly* listen to the song number 4 (for 3 minutes). After listening to these songs Sasha will get pleasure which equals $4 + 3 + 5 = 12$. Sasha will not have time to listen to the song number 5 because he will spend $4 + 3 + 3 = 10$ minutes listening to songs number 2, 3 and 4 and only 1 minute is left after that.

G. New Roads

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

There are n cities in Berland, each of them has a unique id — an integer from 1 to n, the capital is the one with id 1. Now there is a serious problem in Berland with roads — there are no roads.

That is why there was a decision to build n - 1 roads so that there will be exactly one simple path between each pair of cities.

In the construction plan t integers $a_1, a_2, ..., a_t$ were stated, where t equals to the distance from the capital to the most distant city, concerning new roads. a_i equals the number of cities which should be at the distance i from the capital. The distance between two cities is the number of roads one has to pass on the way from one city to another.

Also, it was decided that among all the cities except the capital there should be exactly k cities with exactly one road going from each of them. Such cities are dead-ends and can't be economically attractive. In calculation of these cities the capital is not taken into consideration regardless of the number of roads from it.

Your task is to offer a plan of road's construction which satisfies all the described conditions or to inform that it is impossible.

Input

The first line contains three positive numbers n, t and k ($2 \le n \le 2 \cdot 10^5$, $1 \le t, k < n$) — the distance to the most distant city from the capital and the number of cities which should be dead-ends (the capital in this number is not taken into consideration).

The second line contains a sequence of t integers $a_1, a_2, ..., a_t$ ($1 \le a_i < n$), the i-th number is the number of cities which should be at the distance i from the capital. It is guaranteed that the sum of all the values a_i equals n - 1.

Output

If it is impossible to built roads which satisfy all conditions, print `-1`.

Otherwise, in the first line print one integer n — the number of cities in Berland. In the each of the next n - 1 line print two integers — the ids of cities that are connected by a road. Each road should be printed exactly once. You can print the roads and the cities connected by a road in any order.

If there are multiple answers, print any of them. Remember that the capital has id 1.

Examples

input
7 3 3 2 3 1
output
7 1 3 2 1 2 6 2 4 7 4 3 5

input
14 5 6 4 4 2 2 1
output
14 3 1 1 4 11 6 1 2 10 13 6 10 10 12 14 12 8 4 5 1 3 7 2 6 5 9

input
3 1 1 2
output
-1