Problem Picasso. Zebras

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	1 second
Memory limit:	512 megabytes

Oleg writes down the history of the days he lived. For each day he decides if it was good or bad. Oleg calls a non-empty sequence of days a *zebra*, if it starts with a bad day, ends with a bad day, and good and bad days are alternating in it. Let us denote bad days as `0` and good days as `1`. Then, for example, sequences of days `0`, `010`, `01010` are zebras, while sequences `1`, `0110`, `0101` are not.

Oleg tells you the story of days he lived in chronological order in form of string consisting of `0` and `1`. Now you are interested if it is possible to divide Oleg's life history into several **subsequences**, each of which is a zebra, and the way it can be done. Each day must belong to exactly one of the subsequences. For each of the subsequences, days forming it must be ordered chronologically. Note that subsequence does not have to be a group of consecutive days.

Input

In the only line of input data there is a non-empty string s consisting of characters `0` and `1`, which describes the history of Oleg's life. Its length (denoted as $|s|$) does not exceed 200 000 characters.

Output

If there is a way to divide history into zebra subsequences, in the first line of output you should print an integer k ($1 \le k \le |s|$), the resulting number of subsequences. In the i-th of following k lines first print the integer l_i ($1 \le l_i \le |s|$), which is the length of the i-th subsequence, and then l_i indices of days forming the subsequence. Indices must follow in ascending order. Days are numbered starting from 1. Each index from 1 to n must belong to exactly one subsequence. If there is no way to divide day history into zebra subsequences, print `-1`.

Subsequences may be printed in any order. If there are several solutions, you may print any of them. You do not have to minimize nor maximize the value of k.

Examples

input	output
0010100	3 3 1 3 4 3 2 5 6 1 7
111	-1

Note

Tests for this problem are divided into three groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests of the previous groups.

Group	Points	Additional constraints	Comment
		$\lvert s\rvert$	
0	0	–	Sample tests
1	31	$\lvert s\rvert \le 10$	–
2	29	$\lvert s\rvert \le 2000$	–
3	40	–	–

Problem Kandinsky. Data Center Maintenance

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	1 second
Memory limit:	512 megabytes

BigData Inc. is a corporation that has n data centers indexed from 1 to n that are located all over the world. These data centers provide storage for client data (you can figure out that client data is really big!).

Main feature of services offered by BigData Inc. is the access availability guarantee even under the circumstances of any data center having an outage. Such a guarantee is ensured by using the *two-way replication*. Two-way replication is such an approach for data storage that any piece of data is represented by two identical copies that are stored in two different data centers.

For each of m company clients, let us denote indices of two different data centers storing this client data as $c_{i,1}$ and $c_{i,2}$.

In order to keep data centers operational and safe, the software running on data center computers is being updated regularly. Release cycle of BigData Inc. is one day meaning that the new version of software is being deployed to the data center computers each day.

Data center software update is a non-trivial long process, that is why there is a special hour-long time frame that is dedicated for data center maintenance. During the maintenance period, data center computers are installing software updates, and thus they may be unavailable. Consider the day to be exactly h hours long. For each data center there is an integer u_j ($0 \le u_j \le h - 1$) defining the index of an hour of day, such that during this hour data center j is unavailable due to maintenance.

Summing up everything above, the condition $u_{c_{i,1}} \ne u_{c_{i,2}}$ should hold for each client, or otherwise his data may be unaccessible while data centers that store it are under maintenance.

Due to occasional timezone change in different cities all over the world, the maintenance time in some of the data centers may change by one hour sometimes. Company should be prepared for such situation, that is why they decided to conduct an experiment, choosing some non-empty subset of data centers, and shifting the maintenance time for them by an hour later (i.e. if $u_j = h - 1$, then the new maintenance hour would become 0, otherwise it would become $u_j + 1$). Nonetheless, such an experiment should not break the accessibility guarantees, meaning that data of any client should be still available during any hour of a day after the data center maintenance times are changed.

Such an experiment would provide useful insights, but changing update time is quite an expensive procedure, that is why the company asked you to find out the minimum number of data centers that have to be included in an experiment in order to keep the data accessibility guarantees.

Input

The first line of input contains three integers n, m and h ($2 \le n \le 100\,000$, $1 \le m \le 100\,000$, $2 \le h \le 100\,000$), the number of company data centers, number of clients and the day length of day measured in hours.

The second line of input contains n integers $u_1, u_2, \ldots, u_n$ ($0 \le u_j < h$), j-th of these numbers is an index of a maintenance hour for data center j.

Each of the next m lines contains two integers $c_{i,1}$ and $c_{i,2}$ ($1 \le c_{i,1}, c_{i,2} \le n$, $c_{i,1} \ne c_{i,2}$), defining the data center indices containing the data of client i.

It is guaranteed that the given maintenance schedule allows each client to access at least one copy of his data at any moment of day.

Output

In the first line print the minimum possible number of data centers k ($1 \le k \le n$) that have to be included in an experiment in order to keep the data available for any client.

In the second line print k distinct integers $x_1, x_2, \ldots, x_k$ ($1 \le x_i \le n$), the indices of data centers whose maintenance time will be shifted by one hour later. Data center indices may be printed in any order.

If there are several possible answers, it is allowed to print any of them. It is guaranteed that at there is at least one valid choice of data centers.

Examples

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

Note

Consider the first sample test. The given answer is the only way to conduct an experiment involving the only data center. In such a scenario the third data center has a maintenance during the hour 1, and no two data centers storing the information of the same client have maintenance at the same hour.

On the other hand, for example, if we shift the maintenance time on hour later for the first data center, then the data of clients 1 and 3 will be unavailable during the hour 0.

Scoring

Tests for this problem are divided into six groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests in some of the previous groups. **Offline evaluation** means that your submission will be evaluated on the tests of the group only after the end of the contest.

Group	Points	Additional constraints		Comment
		n	m	
0	0	–	–	Sample tests
1	30	$n \le 10$	$m \le 100$	–
2	30	$n \le 500$	$m \le 1000$	–
3	40	–	–	**Offline evaluation**

Problem Michelangelo. Curfew

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	2 seconds
Memory limit:	512 megabytes

Instructors of Some Informatics School make students go to bed.

The house contains n rooms, in each room exactly b students were supposed to sleep. However, at the time of curfew it happened that many students are not located in their assigned rooms. The rooms are arranged in a row and numbered from 1 to n. Initially, in i-th room there are a_i students. All students are currently somewhere in the house, therefore $a_1 + a_2 + \ldots + a_n = nb$. Also p instructors live in this house ($p \le 2$).

The process of curfew enforcement is the following. One instructor starts near room 1 and moves toward room n. If there is a second instructor, she starts near room n and moves toward room 1. After processing current room, each instructor moves on to the next one. If there are two instructors they enter rooms and move simultaneously. If n is odd and there are two instructors, then only the first instructor processes the middle room. When all rooms are processed, the process ends.

When an instructor processes a room, she counts the number of students in the room, then turns off the light, and locks the room. Also, if the number of students inside the processed room is not equal to b, the instructors writes down the number of this room into her notebook (and turns off the light, and locks the room). Instructors are in a hurry (to prepare the study plan for the next day), so they don't care about who is in the room, but only about the number of students.

While instructors are inside the rooms, students can run between rooms that are not locked and not being processed. A student can run by at most d rooms, that is she can move to a room with number that differs my at most d. Also, after (or instead of) running each student can hide under a bed in a room she is in. In this case the instructor will not count her during the processing. In each room any number of students can hide simultaneously.

Formally, here is what's happening:

- A curfew is announced, at this point in room i there are a_i students.
- Each student can run to another room but not further than d rooms away from her initial room, or stay in place. After that each student can optionally hide under a bed.
- Instructors enter room 1 (and also room n if $p = 2$), counts students there, and locks the room (after it no one can enter or leave this room).
- Each student from rooms with numbers from 2 to n (or to $n - 1$ if $p = 2$) can run to another room but not further than d rooms away from her **current** room, or stay in place. Each student can optionally hide under a bed.
- Instructor(s) move from room 1 to room 2 (and from room n to room $n - 1$ if $p = 2$).
- This process continues until all rooms are processed.

Let x_i denote the number of rooms in which i-th instructor counted the number of students different from b. Students know that the principal will only listen to one complaint, therefore they want to minimize the maximum of numbers x_i. Help them find this value if they use the optimal strategy.

Input

The first line contains four integers p, n, d and b ($1 \le p \le 2$, $2 \le n \le 100\,000$, $1 \le d \le n - 1$, $1 \le b \le 10\,000$), the number of instructors, number of rooms in the house, running distance of a student, official number of students in a room.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$), i-th of which stands for the number of students in the i-th room before curfew announcement.

It is guaranteed that $a_1 + a_2 + \ldots + a_n = nb$.

Output

Output one integer, the minimal possible value of the maximum of x_i.

Examples

input	output
1 5 3 1 0 0 0 5 0	0
1 5 3 10 5 1 1 1 42	1
2 5 1 1 1 0 0 0 4	1
2 6 1 2 3 8 0 1 0 0	2

Note

In the first sample students can run fast enough to reach their own rooms before the instructor enters room 1, thus the answer is 0.

In the second sample the instructor writes down at least one room into her notebook. One of the optimal strategies is the following: before the instructor enters room 1, 10 students run from room 5 to room 2, 10 students from room 5 to room 3, and 10 students from room 5 to room 4. Then in rooms 2, 3, and 4 one student hides under a bed, in room 5 two students hide, and then students do nothing. This way only room 1 gets written down.

In the third sample the first three rooms are processed by the first instructor, and the last two are processed by the second instructor. One of the optimal strategies is the following: firstly three students run from room 5 to room 4, on the next stage two of them run to room 3, and one of those two hides under a bed. This way, the first instructor writes down room 2, and the second writes down nothing.

In the fourth sample one of the optimal strategies is the following: firstly all students in room 1 hide, all students from room 2 run to room 3. On the next stage one student runs from room 3 to room 4, and 5 students hide. This way, the first instructor writes down rooms 1 and 2, the second instructor writes down rooms 5 and 6.

Scoring

Tests for this problem are divided into seven groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests in some of the previous groups. Note that your solution may **not pass** the sample tests, but it will still be accepted for evaluation. **Offline evaluation** means that your submission will be evaluated on the tests of the group only after the end of the contest.

Group	Points	Additional constrinats			Required groups	Comment
		p	n	b		
0	0	–	–	–	–	Sample tests
1	10	$p = 1$	$n \le 1000$	$b = 1$	–	–
2	10	$p = 1$	$n \le 1000$	—	1	–
3	10	—	$n \le 100$	$b = 1$	–	–
4	15	—	$n \le 100$	$b \le 30$	0, 3	–
5	20	—	$n \le 1000$	$b \le 30$	0, 1, 3, 4	–
6	10	$p = 1$	—	—	1, 2	**Offline evaluation**
7	25	—	—	—	0–6	**Offline evaluation**

Problem Munch. Binary Cards

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	1 second
Memory limit:	512 megabytes

It is never too late to play the fancy "Binary Cards" game!

There is an infinite amount of cards of positive and negative ranks that are used in the game. The absolute value of any card rank is a power of two, i.e. each card has a rank of either 2^k or -2^k for some integer $k \geq 0$. There is an infinite amount of cards of any valid rank.

At the beginning of the game player forms his deck that is some multiset (possibly empty) of cards. It is allowed to pick any number of cards of any rank but the small deck is considered to be a skill indicator. Game consists of n rounds. In the i-th round jury tells the player an integer a_i. After that the player is obligated to draw such a subset of his deck that the sum of ranks of the chosen cards is equal to a_i (it is allowed to not draw any cards, in which case the sum is considered to be equal to zero). If player fails to do so, he loses and the game is over. Otherwise, player takes back all of his cards into his deck and the game proceeds to the next round. Player is considered a winner if he is able to draw the suitable set of cards in each of the rounds.

Somebody told you which numbers a_i the jury is going to tell you in each round. Now you want to pick a deck consisting of the minimum number of cards that allows you to win the "Binary Cards" game.

Input

The first line of input contains an integer n ($1 \leq n \leq 100\,000$), the number of rounds in the game.

The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($-100\,000 \leq a_i \leq 100\,000$), the numbers that jury is going to tell in each round.

Output

In the first line print the integer k ($0 \leq k \leq 100\,000$), the minimum number of cards you have to pick in your deck in ordered to win the "Binary Cards".

In the second line print k integers $b_1, b_2, \ldots, b_k$ ($-2^{20} \leq b_i \leq 2^{20}$, $|b_i|$ is a power of two), the ranks of the cards in your deck. You may output ranks in any order. If there are several optimum decks, you are allowed to print any of them.

It is guaranteed that there exists a deck of minimum size satisfying all the requirements above.

Examples

input	output
1 9	2 1 8
5 -1 3 0 4 7	3 4 -1 4
4 2 -2 14 18	3 -2 2 16

Note

In the first sample there is the only round in the game, in which you may simply draw both your cards. Note that this sample test is the only one satisfying the first test group constraints.

In the second sample you may draw the only card -1 in the first round, cards 4 and -1 in the second round, nothing in the third round, the only card 4 in the fourth round and the whole deck in the fifth round.

Scoring

Tests for this problem are divided into fourteen groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests in some of the previous groups. Note that your solution may **not pass** the sample tests, but it will still be accepted for evaluation. **Offline evaluation** means that your submission will be evaluated on the tests of the group only after the end of the contest.

Group	Points	Additional constrinats		Comment
		n	$\lvert a_i\rvert$	
0	0	–	–	Sample tests
1	8	$n = 1$	$\lvert a_i\rvert \le 10$	–
2	7	$n \le 10$	$\lvert a_i\rvert \le 10$	–
3	7	$n \le 30$	$\lvert a_i\rvert \le 30$	–
4	7	$n \le 50$	$\lvert a_i\rvert \le 50$	–
5	7	$n \le 100$	$\lvert a_i\rvert \le 100$	–
6	7	$n \le 300$	$\lvert a_i\rvert \le 300$	–
7	8	$n \le 500$	$\lvert a_i\rvert \le 500$	–
8	7	$n \le 1000$	$\lvert a_i\rvert \le 1000$	**Offline evaluation**
9	6	$n \le 3000$	$\lvert a_i\rvert \le 3000$	**Offline evaluation**
10	7	$n \le 5000$	$\lvert a_i\rvert \le 5000$	**Offline evaluation**
11	6	$n \le 10\,000$	$\lvert a_i\rvert \le 10\,000$	**Offline evaluation**
12	7	$n \le 30\,000$	$\lvert a_i\rvert \le 30\,000$	**Offline evaluation**
13	7	$n \le 50\,000$	$\lvert a_i\rvert \le 50\,000$	**Offline evaluation**
14	9	–	–	**Offline evaluation**