Codeforces Round #377 (Div. 2)

A. Buy a Shovel

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

Polycarp urgently needs a shovel! He comes to the shop and chooses an appropriate one. The shovel that Policarp chooses is sold for k burles. Assume that there is an unlimited number of such shovels in the shop.

In his pocket Polycarp has an unlimited number of "10-burle coins" and exactly one coin of r burles ($1 \le r \le 9$).

What is the minimum number of shovels Polycarp has to buy so that he can pay for the purchase without any change? It is obvious that he can pay for 10 shovels without any change (by paying the requied amount of 10-burle coins and not using the coin of r burles). But perhaps he can buy fewer shovels and pay without any change. Note that Polycarp should buy at least one shovel.

Input

The single line of input contains two integers k and r ($1 \le k \le 1000$, $1 \le r \le 9$) — the price of one shovel and the denomination of the coin in Polycarp's pocket that is different from "10-burle coins".

Remember that he has an unlimited number of coins in the denomination of 10, that is, Polycarp has enough money to buy any number of shovels.

Output

Print the required minimum number of shovels Polycarp has to buy so that he can pay for them without any change.

Examples

input
117 3
output
9

input
237 7
output
1

input
15 2
output
2

Note

In the first example Polycarp can buy 9 shovels and pay $9 \cdot 117 = 1053$ burles. Indeed, he can pay this sum by using 10-burle coins and one 3-burle coin. He can't buy fewer shovels without any change.

In the second example it is enough for Polycarp to buy one shovel.

In the third example Polycarp should buy two shovels and pay $2 \cdot 15 = 30$ burles. It is obvious that he can pay this sum without any change.

B. Cormen — The Best Friend Of a Man

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

Recently a dog was bought for Polycarp. The dog's name is Cormen. Now Polycarp has a lot of troubles. For example, Cormen likes going for a walk.

Empirically Polycarp learned that the dog needs at least k walks for any two consecutive days in order to feel good. For example, if $k = 5$ and yesterday Polycarp went for a walk with Cormen 2 times, today he has to go for a walk at least 3 times.

Polycarp analysed all his affairs over the next n days and made a sequence of n integers $a_1, a_2, ..., a_n$, where a_i is the number of times Polycarp will walk with the dog on the i-th day while doing all his affairs (for example, he has to go to a shop, throw out the trash, etc.).

Help Polycarp determine the minimum number of walks he needs to do additionaly in the next n days so that Cormen will feel good during all the n days. You can assume that on the day before the first day and on the day after the n-th day Polycarp will go for a walk with Cormen exactly k times.

Write a program that will find the minumum number of additional walks and the appropriate schedule — the sequence of integers $b_1, b_2, ..., b_n$ ($b_i \geq a_i$), where b_i means the total number of walks with the dog on the i-th day.

Input

The first line contains two integers n and k ($1 \leq n, k \leq 500$) — the number of days and the minimum number of walks with Cormen for any two consecutive days.

The second line contains integers $a_1, a_2, ..., a_n$ ($0 \leq a_i \leq 500$) — the number of walks with Cormen on the i-th day which Polycarp has already planned.

Output

In the first line print the smallest number of additional walks that Polycarp should do during the next n days so that Cormen will feel good during all days.

In the second line print n integers $b_1, b_2, ..., b_n$, where b_i — the total number of walks on the i-th day according to the found solutions ($a_i \leq b_i$ for all i from 1 to n). If there are multiple solutions, print any of them.

Examples

input
3 5 2 0 1
output
4 2 3 2

input
3 1 0 0 0
output
1 0 1 0

input
4 6 2 4 3 5
output
0 2 4 3 5

C. Sanatorium

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

Vasiliy spent his vacation in a sanatorium, came back and found that he completely forgot details of his vacation!

Every day there was a breakfast, a dinner and a supper in a dining room of the sanatorium (of course, in this order). The only thing that Vasiliy has now is a card from the dining room contaning notes how many times he had a breakfast, a dinner and a supper (thus, the card contains three integers). Vasiliy could sometimes have missed some meal, for example, he could have had a breakfast and a supper, but a dinner, or, probably, at some days he haven't been at the dining room at all.

Vasiliy doesn't remember what was the time of the day when he arrived to sanatorium (before breakfast, before dinner, before supper or after supper), and the time when he left it (before breakfast, before dinner, before supper or after supper). So he considers any of these options. After Vasiliy arrived to the sanatorium, he was there all the time until he left. Please note, that it's possible that Vasiliy left the sanatorium on the same day he arrived.

According to the notes in the card, help Vasiliy determine the minimum number of meals in the dining room that he could have missed. We shouldn't count as missed meals on the arrival day before Vasiliy's arrival and meals on the departure day after he left.

Input

The only line contains three integers b, d and s ($0 \le b, d, s \le 10^{18},\ b + d + s \ge 1$) — the number of breakfasts, dinners and suppers which Vasiliy had during his vacation in the sanatorium.

Output

Print single integer — the minimum possible number of meals which Vasiliy could have missed during his vacation.

Examples

input
3 2 1
output
1

input
1 0 0
output
0

input
1 1 1
output
0

input
1000000000000000000 0 1000000000000000000
output
999999999999999999

Note

In the first sample, Vasiliy could have missed one supper, for example, in case he have arrived before breakfast, have been in the sanatorium for two days (including the day of arrival) and then have left after breakfast on the third day.

In the second sample, Vasiliy could have arrived before breakfast, have had it, and immediately have left the sanatorium, not missing any meal.

In the third sample, Vasiliy could have been in the sanatorium for one day, not missing any meal.

D. Exams

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

Vasiliy has an exam period which will continue for n days. He has to pass exams on m subjects. Subjects are numbered from 1 to m.

About every day we know exam for which one of m subjects can be passed on that day. Perhaps, some day you can't pass any exam. It is not allowed to pass more than one exam on any day.

On each day Vasiliy can either pass the exam of that day (it takes the whole day) or prepare all day for some exam or have a rest.

About each subject Vasiliy know a number a_i — the number of days he should prepare to pass the exam number i. Vasiliy can switch subjects while preparing for exams, it is not necessary to prepare continuously during a_i days for the exam number i. He can mix the order of preparation for exams in any way.

Your task is to determine the minimum number of days in which Vasiliy can pass all exams, or determine that it is impossible. Each exam should be passed exactly one time.

Input

The first line contains two integers n and m ($1 \le n, m \le 10^5$) — the number of days in the exam period and the number of subjects.

The second line contains n integers $d_1, d_2, ..., d_n$ ($0 \le d_i \le m$), where d_i is the number of subject, the exam of which can be passed on the day number i. If d_i equals 0, it is not allowed to pass any exams on the day number i.

The third line contains m positive integers $a_1, a_2, ..., a_m$ ($1 \le a_i \le 10^5$), where a_i is the number of days that are needed to prepare before passing the exam on the subject i.

Output

Print one integer — the minimum number of days in which Vasiliy can pass all exams. If it is impossible, print `-1`.

Examples

input
7 2 0 1 0 2 1 0 2 2 1
output
5

input
10 3 0 0 1 2 3 0 2 0 1 2 1 1 4
output
9

input
5 1 1 1 1 1 1 5
output
-1

Note

In the first example Vasiliy can behave as follows. On the first and the second day he can prepare for the exam number 1 and pass it on the fifth day, prepare for the exam number 2 on the third day and pass it on the fourth day.

In the second example Vasiliy should prepare for the exam number 3 during the first four days and pass it on the fifth day. Then on the sixth day he should prepare for the exam number 2 and then pass it on the seventh day. After that he needs to prepare for the exam number 1 on the eighth day and pass it on the ninth day.

In the third example Vasiliy can't pass the only exam because he hasn't anough time to prepare for it.

E. Sockets

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

The ICM ACPC World Finals is coming! Unfortunately, the organizers of the competition were so busy preparing tasks that totally missed an important technical point — the organization of electricity supplement for all the participants workstations.

There are n computers for participants, the i-th of which has power equal to positive integer p_i. At the same time there are m sockets available, the j-th of which has power euqal to positive integer s_j. It is possible to connect the i-th computer to the j-th socket if and only if their powers are the same: $p_i = s_j$. It is allowed to connect no more than one computer to one socket. Thus, if the powers of all computers and sockets are distinct, then no computer can be connected to any of the sockets.

In order to fix the situation professor Puch Williams urgently ordered a wagon of adapters — power splitters. Each adapter has one plug and one socket with a voltage divider between them. After plugging an adapter to a socket with power x, the power on the adapter's socket becomes equal to , it means that it is equal to the socket's power divided by two with rounding up, for example and .

Each adapter can be used only once. It is possible to connect several adapters in a chain plugging the first to a socket. For example, if two adapters are plugged one after enother to a socket with power 10, it becomes possible to connect one computer with power 3 to this socket.

The organizers should install adapters so that it will be possible to supply with electricity the maximum number of computers c at the same time. If there are several possible connection configurations, they want to find the one that uses the minimum number of adapters u to connect c computers.

Help organizers calculate the maximum number of connected computers c and the minimum number of adapters u needed for this.

The wagon of adapters contains enough of them to do the task. It is guaranteed that it's possible to connect at least one computer.

Input

The first line contains two integers n and m ($1 \le n, m \le 200\,000$) — the number of computers and the number of sockets.

The second line contains n integers $p_1, p_2, ..., p_n$ ($1 \le p_i \le 10^9$) — the powers of the computers.

The third line contains m integers $s_1, s_2, ..., s_m$ ($1 \le s_i \le 10^9$) — the power of the sockets.

Output

In the first line print two numbers c and u — the maximum number of computers which can at the same time be connected to electricity and the minimum number of adapters needed to connect c computers.

In the second line print m integers $a_1, a_2, ..., a_m$ ($0 \le a_i \le 10^9$), where a_i equals the number of adapters orginizers need to plug into the i-th socket. The sum of all a_i should be equal to u.

In third line print n integers $b_1, b_2, ..., b_n$ ($0 \le b_i \le m$), where the b_j-th equals the number of the socket which the j-th computer should be connected to. $b_j = 0$ means that the j-th computer should not be connected to any socket. All b_j that are different from 0 should be distinct. The power of the j-th computer should be equal to the power of the socket b_j after plugging in a_{b_j} adapters. The number of non-zero b_j should be equal to c.

If there are multiple answers, print any of them.

Examples

input
2 2 1 1 2 2
output
2 2 1 1 1 2

input
2 1 2 100 99
output
1 6 6 1 0

F. Tourist Reform

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Berland is a tourist country! At least, it can become such — the government of Berland is confident about this.

There are n cities in Berland, some pairs of which are connected by two-ways roads. Each road connects two different cities. In Berland there are no roads which connect the same pair of cities. It is possible to get from any city to any other city using given two-ways roads.

According to the reform each road will become one-way. It will be oriented to one of two directions.

To maximize the tourist attraction of Berland, after the reform for each city i the value r_i will be calculated. It will equal to the number of cities x for which there is an oriented path from the city i to the city x. In other words, r_i will equal the number of cities which can be reached from the city i by roads.

The government is sure that tourist's attention will be focused on the minimum value of r_i.

Help the government of Berland make the reform to maximize the minimum of r_i.

Input

The first line contains two integers n, m ($2 \le n \le 400\,000$, $1 \le m \le 400\,000$) — the number of cities and the number of roads.

The next m lines describe roads in Berland: the j-th of them contains two integers u_j and v_j ($1 \le u_j, v_j \le n$, $u_j \ne v_j$), where u_j and v_j are the numbers of cities which are connected by the j-th road.

The cities are numbered from 1 to n. It is guaranteed that it is possible to get from any city to any other by following two-ways roads. In Berland there are no roads which connect the same pair of cities.

Output

In the first line print single integer — the maximum possible value $min_{1 \le i \le n}\{r_i\}$ after the orientation of roads.

The next m lines must contain the description of roads after the orientation: the j-th of them must contain two integers u_j, v_j, it means that the j-th road will be directed from the city u_j to the city v_j. Print roads in the same order as they are given in the input data.

Example

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