A. Median Smoothing

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

A schoolboy named Vasya loves reading books on programming and mathematics. He has recently read an encyclopedia article that described the method of *median smoothing* (or median filter) and its many applications in science and engineering. Vasya liked the idea of the method very much, and he decided to try it in practice.

Applying the simplest variant of median smoothing to the sequence of numbers $a_1, a_2, ..., a_n$ will result a new sequence $b_1, b_2, ..., b_n$ obtained by the following algorithm:

- $b_1 = a_1$, $b_n = a_n$, that is, the first and the last number of the new sequence match the corresponding numbers of the original sequence.
- For $i = 2, ..., n - 1$ value b_i is equal to the *median* of three values $a_{i - 1}$, a_i and $a_{i + 1}$.

The *median* of a set of three numbers is the number that goes on the second place, when these three numbers are written in the non-decreasing order. For example, the median of the set 5, 1, 2 is number 2, and the median of set 1, 0, 1 is equal to 1.

In order to make the task easier, Vasya decided to apply the method to sequences consisting of zeros and ones only.

Having made the procedure once, Vasya looked at the resulting sequence and thought: what if I apply the algorithm to it once again, and then apply it to the next result, and so on? Vasya tried a couple of examples and found out that after some number of median smoothing algorithm applications the sequence can stop changing. We say that the sequence is *stable*, if it does not change when the median smoothing is applied to it.

Now Vasya wonders, whether the sequence always eventually becomes stable. He asks you to write a program that, given a sequence of zeros and ones, will determine whether it ever becomes stable. Moreover, if it ever becomes stable, then you should determine what will it look like and how many times one needs to apply the median smoothing algorithm to initial sequence in order to obtain a stable one.

Input

The first input line of the input contains a single integer n $(3 \le n \le 500\,000)$ — the length of the initial sequence.

The next line contains n integers $a_1, a_2, ..., a_n$ $(a_i = 0$ or $a_i = 1)$, giving the initial sequence itself.

Output

If the sequence will never become stable, print a single number -1.

Otherwise, first print a single integer — the minimum number of times one needs to apply the median smoothing algorithm to the initial sequence before it becomes is stable. In the second line print n numbers separated by a space — the resulting sequence itself.

Sample test(s)

input
```
4
0 0 1 1
```
output
```
0
0 0 1 1
```

input
```
5
0 1 0 1 0
```
output
```
2
0 0 0 0 0
```

Note

In the second sample the stabilization occurs in two steps: $01010 \rightarrow 00100 \rightarrow 00000$, and the sequence 00000 is obviously stable.

B. Chip 'n Dale Rescue Rangers

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

A team of furry rescue rangers was sitting idle in their hollow tree when suddenly they received a signal of distress. In a few moments they were ready, and the dirigible of the rescue chipmunks hit the road.

We assume that the action takes place on a Cartesian plane. The headquarters of the rescuers is located at point (x_1, y_1), and the distress signal came from the point (x_2, y_2).

Due to Gadget's engineering talent, the rescuers' dirigible can instantly change its current velocity and direction of movement at any moment and as many times as needed. The only limitation is: the speed of the aircraft relative to the air can not exceed $v_{\max}$ meters per second.

Of course, Gadget is a true rescuer and wants to reach the destination as soon as possible. The matter is complicated by the fact that the wind is blowing in the air and it affects the movement of the dirigible. According to the weather forecast, the wind will be defined by the vector (v_x, v_y) for the nearest t seconds, and then will change to (w_x, w_y). These vectors give both the direction and velocity of the wind. Formally, if a dirigible is located at the point (x, y), while its own velocity relative to the air is equal to zero and the wind (u_x, u_y) is blowing, then after T seconds the new position of the dirigible will be $(x + \tau \cdot u_x, \ y + \tau \cdot u_y)$.

Gadget is busy piloting the aircraft, so she asked Chip to calculate how long will it take them to reach the destination if they fly optimally. He coped with the task easily, but Dale is convinced that Chip has given the random value, aiming only not to lose the face in front of Gadget. Dale has asked you to find the right answer.

It is guaranteed that the speed of the wind at any moment of time is strictly less than the maximum possible speed of the airship relative to the air.

Input

The first line of the input contains four integers x_1, y_1, x_2, y_2 ($|x_1|, \ |y_1|, \ |x_2|, \ |y_2| \le 10\,000$) — the coordinates of the rescuers' headquarters and the point, where signal of the distress came from, respectively.

The second line contains two integers $v_{\max}$ and t ($0 < v, t \le 1000$), which are denoting the maximum speed of the chipmunk dirigible relative to the air and the moment of time when the wind changes according to the weather forecast, respectively.

Next follow one per line two pairs of integer (v_x, v_y) and (w_x, w_y), describing the wind for the first t seconds and the wind that will blow at all the remaining time, respectively. It is guaranteed that $v_x^2 + v_y^2 < v_{\max}^2$ and $w_x^2 + w_y^2 < v_{\max}^2$.

Output

Print a single real value — the minimum time the rescuers need to get to point (x_2, y_2). You answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct, if $\frac{|a-b|}{\max(1,b)} \le 10^{-6}$.

Sample test(s)

input

```
0 0 5 5
3 2
-1 -1
-1 0
```

output

```
3.729935587093555327
```

input

```
0 0 0 1000
100 1000
-50 0
50 0
```

output

```
11.547005383792516398
```

C. Three States

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

The famous global economic crisis is approaching rapidly, so the states of Berman, Berance and Bertaly formed an alliance and allowed the residents of all member states to freely pass through the territory of any of them. In addition, it was decided that a road between the states should be built to guarantee so that one could any point of any country can be reached from any point of any other State.

Since roads are always expensive, the governments of the states of the newly formed alliance asked you to help them assess the costs. To do this, you have been issued a map that can be represented as a rectangle table consisting of n rows and m columns. Any cell of the map either belongs to one of three states, or is an area where it is allowed to build a road, or is an area where the construction of the road is not allowed. A cell is called *passable*, if it belongs to one of the states, or the road was built in this cell. From any passable cells you can move up, down, right and left, if the cell that corresponds to the movement exists and is passable.

Your task is to construct a road inside a minimum number of cells, so that it would be possible to get from any cell of any state to any cell of any other state using only passable cells.

It is guaranteed that initially it is possible to reach any cell of any state from any cell of this state, moving only along its cells. It is also guaranteed that for any state there is at least one cell that belongs to it.

Input

The first line of the input contains the dimensions of the map n and m ($1 \le n, m \le 1000$) — the number of rows and columns respectively.

Each of the next n lines contain m characters, describing the rows of the map. Digits from 1 to 3 represent the accessory to the corresponding state. The character '`.`' corresponds to the cell where it is allowed to build a road and the character '`#`' means no construction is allowed in this cell.

Output

Print a single integer — the minimum number of cells you need to build a road inside in order to connect all the cells of all states. If such a goal is unachievable, print `-1`.

Sample test(s)

input

```
4 5
11..2
#..22
#.323
.#333
```

output

```
2
```

input

```
1 5
1#2#3
```

output

```
-1
```

D. Top Secret Task

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

A top-secret military base under the command of Colonel Zuev is expecting an inspection from the Ministry of Defence. According to the charter, each top-secret military base must include a top-secret troop that should... well, we cannot tell you exactly what it should do, it is a top secret troop at the end. The problem is that Zuev's base is missing this top-secret troop for some reasons.

The colonel decided to deal with the problem immediately and ordered to line up in a single line all n soldiers of the base entrusted to him. Zuev knows that the *loquacity* of the i-th soldier from the left is equal to q_i. Zuev wants to form the top-secret troop using k leftmost soldiers in the line, thus he wants their total loquacity to be as small as possible (as the troop should remain top-secret). To achieve this, he is going to choose a pair of **consecutive** soldiers and swap them. He intends to do so no more than s times. Note that any soldier can be a participant of such swaps for any number of times. The problem turned out to be unusual, and colonel Zuev asked you to help.

Determine, what is the minimum total loquacity of the first k soldiers in the line, that can be achieved by performing no more than s swaps of two consecutive soldiers.

Input

The first line of the input contains three positive integers n, k, s ($1 \le k \le n \le 150$, $1 \le s \le 10^9$) — the number of soldiers in the line, the size of the top-secret troop to be formed and the maximum possible number of swap operations of the consecutive pair of soldiers, respectively.

The second line of the input contains n integer q_i ($1 \le q_i \le 1\,000\,000$) — the values of loquacity of soldiers in order they follow in line from left to right.

Output

Print a single integer — the minimum possible total loquacity of the top-secret troop.

Sample test(s)

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

input
```
5 4 2
10 1 6 2 5
```
output
```
18
```

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

Note

In the first sample Colonel has to swap second and third soldiers, he doesn't really need the remaining swap. The resulting soldiers order is: $(2, 1, 4)$. Minimum possible summary loquacity of the secret troop is 3. In the second sample Colonel will perform swaps in the following order:

1. (10, 1, **6** — **2**, 5)
2. (10, 1, 2, **6** — **5**)

The resulting soldiers order is (10, 1, 2, 5, 6).

Minimum possible summary loquacity is equal to 18.

E. Birthday

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Today is birthday of a Little Dasha — she is now 8 years old! On this occasion, each of her n friends and relatives gave her a ribbon with a greeting written on it, and, as it turned out, all the greetings are different. Dasha gathered all the ribbons and decided to throw away some of them in order to make the remaining set *stylish*. The birthday girl considers a set of ribbons stylish if no greeting written on some ribbon is a substring of another greeting written on some other ribbon. Let us recall that the substring of the string s is a continuous segment of s.

Help Dasha to keep as many ribbons as possible, so that she could brag about them to all of her friends. Dasha cannot rotate or flip ribbons, that is, each greeting can be read in a single way given in the input.

Input

The first line of the input contains integer n $(1 \le n \le 750)$ — the number of Dasha's relatives and friends.

Each of the next n lines contains exactly one greeting. Each greeting consists of characters 'a' and 'b' only.

The total length of all greetings won't exceed $10\ 000\ 000$ characters.

Output

In the first line print the maximum size of the stylish set. In the second line print the numbers of ribbons involved in it, assuming that they are numbered from 1 to n in the order they appear in the input. If there are several stylish sets of the maximum size, print any of them.

Sample test(s)

input

```
5
abab
aba
aabab
ababb
bab
```

output

```
2
2 5
```

Note

In the sample, the answer that keeps ribbons 3 and 4 is also considered correct.