A. Equator

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

Polycarp has created his own training plan to prepare for the programming contests. He will train for n days, all days are numbered from 1 to n, beginning from the first.

On the i-th day Polycarp will necessarily solve a_i problems. One evening Polycarp plans to celebrate the *equator*. He will celebrate it on the first evening of such a day that from the beginning of the training and to this day inclusive he will solve half or more of all the problems.

Determine the index of day when Polycarp will celebrate the equator.

Input

The first line contains a single integer n ($1 \le n \le 200\,000$) — the number of days to prepare for the programming contests.

The second line contains a sequence $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10\,000$), where a_i equals to the number of problems, which Polycarp will solve on the i-th day.

Output

Print the index of the day when Polycarp will celebrate the equator.

Examples

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

input
```
6
2 2 2 2 2 2
```
output
```
3
```

Note

In the first example Polycarp will celebrate the equator on the evening of the second day, because up to this day (inclusive) he will solve 4 out of 7 scheduled problems on four days of the training.

In the second example Polycarp will celebrate the equator on the evening of the third day, because up to this day (inclusive) he will solve 6 out of 12 scheduled problems on six days of the training.

B. Students in Railway Carriage

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

There are n consecutive seat places in a railway carriage. Each place is either empty or occupied by a passenger.

The university team for the Olympiad consists of a student-programmers and b student-athletes. Determine the largest number of students from all $a+b$ students, which you can put in the railway carriage so that:

- no student-programmer is sitting next to the student-programmer;
- and no student-athlete is sitting next to the student-athlete.

In the other words, there should not be two consecutive (adjacent) places where two student-athletes or two student-programmers are sitting.

Consider that initially occupied seat places are occupied by jury members (who obviously are not students at all).

Input

The first line contain three integers n, a and b ($1 \le n \le 2\cdot10^{5}$, $0 \le a, b \le 2\cdot10^{5}$, $a + b > 0$) — total number of seat places in the railway carriage, the number of student-programmers and the number of student-athletes.

The second line contains a string with length n, consisting of characters "." and "*". The dot means that the corresponding place is empty. The asterisk means that the corresponding place is occupied by the jury member.

Output

Print the largest number of students, which you can put in the railway carriage so that no student-programmer is sitting next to a student-programmer and no student-athlete is sitting next to a student-athlete.

Examples

input

```
5 1 1
*...*
```

output

```
2
```

input

```
6 2 3
*...*.
```

output

```
4
```

input

```
11 3 10
.*....**.*.
```

output

```
7
```

input

```
3 2 3
***
```

output

```
0
```

Note

In the first example you can put all student, for example, in the following way: `*.AB*`

In the second example you can put four students, for example, in the following way: `*BAB*B`

In the third example you can put seven students, for example, in the following way: `B*ABAB**A*B`

The letter `A` means a student-programmer, and the letter `B` — student-athlete.

C. Make a Square

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given a positive integer n, written without leading zeroes (for example, the number `04` is incorrect).

In one operation you can delete any digit of the given integer so that the result remains a positive integer without leading zeros.

Determine the minimum number of operations that you need to consistently apply to the given integer n to make from it the square of some positive integer or report that it is impossible.

An integer x is the square of some positive integer if and only if $x=y^2$ for some positive integer y.

Input

The first line contains a single integer n ($1 \le n \le 2 \cdot 10^{9}$). The number is given without leading zeroes.

Output

If it is impossible to make the square of some positive integer from n, print `-1`. In the other case, print the minimal number of operations required to do it.

Examples

input

```
8314
```

output

```
2
```

input
625
output
0

input
333
output
-1

Note

In the first example we should delete from $$$8314$$$ the digits $$$3$$$ and $$$4$$$. After that $$$8314$$$ become equals to $$$81$$$, which is the square of the integer $$$9$$$.

In the second example the given $$$625$$$ is the square of the integer $$$25$$$, so you should not delete anything.

In the third example it is impossible to make the square from $$$333$$$, so the answer is `-1`.

D. Merge Equals

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given an array of positive integers. While there are at least two equal elements, we will perform the following operation. We choose the smallest value $$$x$$$ that occurs in the array $$$2$$$ or more times. Take the first two occurrences of $$$x$$$ in this array (the two leftmost occurrences). Remove the left of these two occurrences, and the right one is replaced by the sum of this two values (that is, $$$2 \cdot x$$$).

Determine how the array will look after described operations are performed.

For example, consider the given array looks like $$$[3, 4, 1, 2, 2, 1, 1]$$$. It will be changed in the following way: $$$[3, 4, 1, 2, 2, 1, 1]~\rightarrow~[3, 4, 2, 2, 2, 1]~\rightarrow~[3, 4, 4, 2, 1]~\rightarrow~[3, 8, 2, 1]$$$.

If the given array is look like $$$[1, 1, 3, 1, 1]$$$ it will be changed in the following way: $$$[1, 1, 3, 1, 1]~\rightarrow~[2, 3, 1, 1]~\rightarrow~[2, 3, 2]~\rightarrow~[3, 4]$$$.

Input

The first line contains a single integer $$$n$$$ ($$$2 \le n \le 150\,000$$$) — the number of elements in the array.

The second line contains a sequence from $$$n$$$ elements $$$a_1, a_2, \dots, a_n$$$ ($$$1 \le a_i \le 10^{9}$$$) — the elements of the array.

Output

In the first line print an integer $$$k$$$ — the number of elements in the array after all the performed operations. In the second line print $$$k$$$ integers — the elements of the array after all the performed operations.

Examples

input
```
7
3 4 1 2 2 1 1
```
output
```
4
3 8 2 1
```

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

input
```
5
10 40 20 50 30
```
output
```
5
10 40 20 50 30
```

Note

The first two examples were considered in the statement.

In the third example all integers in the given array are distinct, so it will not change.

E. Byteland, Berland and Disputed Cities

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

The cities of Byteland and Berland are located on the axis Ox. In addition, on this axis there are also disputed cities, which belong to each of the countries in their opinion. Thus, on the line Ox there are three types of cities:

- the cities of Byteland,
- the cities of Berland,
- disputed cities.

Recently, the project BNET has been launched — a computer network of a new generation. Now the task of the both countries is to connect the cities so that the network of this country is *connected*.

The countries agreed to connect *the pairs* of cities with BNET cables in such a way that:

- If you look at the *only* cities of Byteland and the disputed cities, then in the resulting set of cities, any city should be reachable from any other one by one or more cables,
- If you look at the *only* cities of Berland and the disputed cities, then in the resulting set of cities, any city should be reachable from any other one by one or more cables.

Thus, it is necessary to choose a set of pairs of cities to connect by cables in such a way that both conditions are satisfied simultaneously. Cables allow bi-directional data transfer. Each cable connects exactly two distinct cities.

The cost of laying a cable from one city to another is equal to the distance between them. Find the minimum total cost of laying a set of cables so that two subsets of cities (Byteland and disputed cities, Berland and disputed cities) are connected.

Each city is a point on the line Ox. It is technically possible to connect the cities a and b with a cable so that the city c ($a < c < b$) is not connected to this cable, where a, b and c are simultaneously coordinates of the cities a, b and c.

Input

The first line contains a single integer n ($2 \le n \le 2 \cdot 10^{5}$) — the number of cities.

The following n lines contains an integer x_i and the letter c_i ($-10^{9} \le x_i \le 10^{9}$) — the coordinate of the city and its type. If the city belongs to Byteland, c_i equals to '`B`'. If the city belongs to Berland, c_i equals to «`R`». If the city is disputed, c_i equals to '`P`'.

All cities have distinct coordinates. Guaranteed, that the cities are given in the increasing order of their coordinates.

Output

Print the minimal total length of such set of cables, that if we delete all Berland cities (c_i='`R`'), it will be possible to find a way from any remaining city to any other remaining city, moving only by cables. Similarly, if we delete all Byteland cities (c_i='`B`'), it will be possible to find a way from any remaining city to any other remaining city, moving only by cables.

Examples

input

```
4
-5 R
0 P
3 P
7 B
```

output

```
12
```

input

```
5
10 R
14 B
16 B
21 R
32 R
```

output

```
24
```

Note

In the first example, you should connect the first city with the second, the second with the third, and the third with the fourth. The total length of the cables will be $5 + 3 + 4 = 12$.

In the second example there are no disputed cities, so you need to connect all the neighboring cities of Byteland and all the neighboring cities of

Berland. The cities of Berland have coordinates $$$10, 21, 32$$$, so to connect them you need two cables of length $$$11$$$ and $$$11$$$. The cities of Byteland have coordinates $$$14$$$ and $$$16$$$, so to connect them you need one cable of length $$$2$$$. Thus, the total length of all cables is $$$11 + 11 + 2 = 24$$$.

F. Simple Cycles Edges

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

You are given an undirected graph, consisting of $$$n$$$ vertices and $$$m$$$ edges. The graph does not necessarily connected. Guaranteed, that the graph does not contain multiple edges (more than one edges between a pair of vertices) or loops (edges from a vertex to itself).

A cycle in a graph is called a simple, if it contains each own vertex exactly once. So simple cycle doesn't allow to visit a vertex more than once in a cycle.

Determine the edges, which belong to *exactly on one* simple cycle.

Input

The first line contain two integers $$$n$$$ and $$$m$$$ $$$(1 \le n \le 100\,000$$$, $$$0 \le m \le \min(n \cdot (n - 1) / 2, 100\,000))$$$ — the number of vertices and the number of edges.

Each of the following $$$m$$$ lines contain two integers $$$u$$$ and $$$v$$$ ($$$1 \le u, v \le n$$$, $$$u \neq v$$$) — the description of the edges.

Output

In the first line print the number of edges, which belong to exactly one simple cycle.

In the second line print the indices of edges, which belong to exactly one simple cycle, in **increasing** order. The edges are numbered from one in the same order as they are given in the input.

Examples

input

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

output

```
3
1 2 3
```

input

```
6 7
2 3
3 4
4 2
1 2
1 5
5 6
6 1
```

output

```
6
1 2 3 5 6 7
```

input

```
5 6
1 2
2 3
2 4
4 3
2 5
5 3
```

output

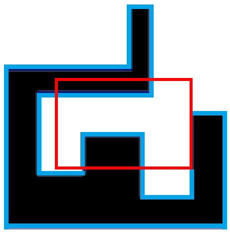
```
0
```

G. Visible Black Areas

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

Petya has a polygon consisting of n vertices. All sides of the Petya's polygon are parallel to the coordinate axes, and each two adjacent sides of the Petya's polygon are perpendicular. It is guaranteed that the polygon is simple, that is, it doesn't have self-intersections and self-touches. All internal area of the polygon (borders are not included) was painted in black color by Petya.

Also, Petya has a rectangular window, defined by its coordinates, through which he looks at the polygon. A rectangular window can not be moved. The sides of the rectangular window are parallel to the coordinate axes.



Blue color represents the border of a polygon, red color is the Petya's window. The answer in this case is 2.

Determine the number of black connected areas of Petya's polygon, which can be seen through the rectangular window.

Input

The first line contain four integers x_1, y_1, x_2, y_2 ($x_1 < x_2$, $y_2 < y_1$) — the coordinates of top-left and bottom-right corners of the rectangular window.

The second line contains a single integer n ($4 \le n \le 15\,000$) — the number of vertices in Petya's polygon.

Each of the following n lines contains two integers — the coordinates of vertices of the Petya's polygon in counterclockwise order. Guaranteed, that the given polygon satisfies the conditions described in the statement.

All coordinates of the rectangular window and all coordinates of the vertices of the polygon are non-negative and do not exceed $15\,000$.

Output

Print the number of black connected areas of Petya's polygon, which can be seen through the rectangular window.

Example

input

```
5 7 16 3
16
0 0
18 0
18 6
16 6
16 1
10 1
10 4
7 4
7 2
2 2
2 6
12 6
12 12
10 12
10 8
0 8
```

output

```
2
```

Note

The example corresponds to the picture above.