# Codeforces Round #549 (Div. 2)

## A. The Doors

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

Three years have passes and nothing changed. It is still raining in London, and Mr. Black has to close all the doors in his home in order to not be flooded. Once, however, Mr. Black became so nervous that he opened one door, then another, then one more and so on until he opened all the doors in his house.

There are exactly two exits from Mr. Black's house, let's name them left and right exits. There are several doors in each of the exits, so each door in Mr. Black's house is located either in the left or in the right exit. You know where each door is located. Initially all the doors are closed. Mr. Black can exit the house if and only if all doors in at least one of the exits is open. You are given a sequence in which Mr. Black opened the doors, please find the smallest index $k$ such that Mr. Black can exit the house after opening the first $k$ doors.

We have to note that Mr. Black opened each door at most once, and in the end all doors became open.

### Input
The first line contains integer $n$ ($2 \leq n \leq 200\,000$) — the number of doors.

The next line contains $n$ integers: the sequence in which Mr. Black opened the doors. The $i$-th of these integers is equal to $0$ in case the $i$-th opened door is located in the left exit, and it is equal to $1$ in case it is in the right exit.

It is guaranteed that there is at least one door located in the left exit and there is at least one door located in the right exit.

### Output
Print the smallest integer $k$ such that after Mr. Black opened the first $k$ doors, he was able to exit the house.

#### Examples

| input |
| --- |
| 5<br>0 0 1 0 0 |
| **output** |
| 3 |

| input |
| --- |
| 4<br>1 0 0 1 |
| **output** |
| 3 |

### Note
In the first example the first two doors are from the left exit, so when Mr. Black opened both of them only, there were two more closed door in the left exit and one closed door in the right exit. So Mr. Black wasn't able to exit at that moment.

When he opened the third door, all doors from the right exit became open, so Mr. Black was able to exit the house.

In the second example when the first two doors were opened, there was open closed door in each of the exit.

With three doors opened Mr. Black was able to use the left exit.

## B. Nirvana

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

Kurt reaches nirvana when he finds the product of all the digits of some positive integer. Greater value of the product makes the nirvana deeper.

Help Kurt find the maximum possible product of digits among all integers from $1$ to $n$.

### Input
The only input line contains the integer $n$ ($1 \leq n \leq 2 \cdot 10^9$).

### Output
Print the maximum product of digits among all integers from $1$ to $n$.

#### Examples

| input |
| --- |
| 390 |
| **output** |
| 216 |

| input |
| --- |
| 7 |
| **output** |
| 7 |

| input |
| --- |
| 1000000000 |
| **output** |
| 387420489 |

### Note
In the first example the maximum product is achieved for $389$ (the product of digits is $3 \cdot 8 \cdot 9 = 216$).

In the second example the maximum product is achieved for $7$ (the product of digits is $7$).

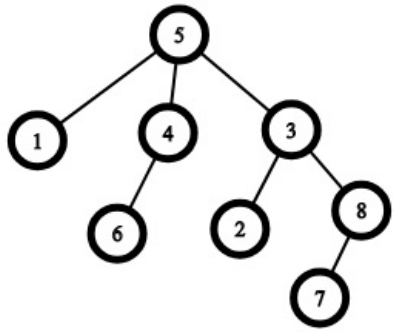
In the third example the maximum product is achieved for $999999999$ (the product of digits is $9^9 = 387420489$).

## C. Queen

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

You are given a rooted tree with vertices numerated from $1$ to $n$. A tree is a connected graph without cycles. A rooted tree has a special vertex named root.

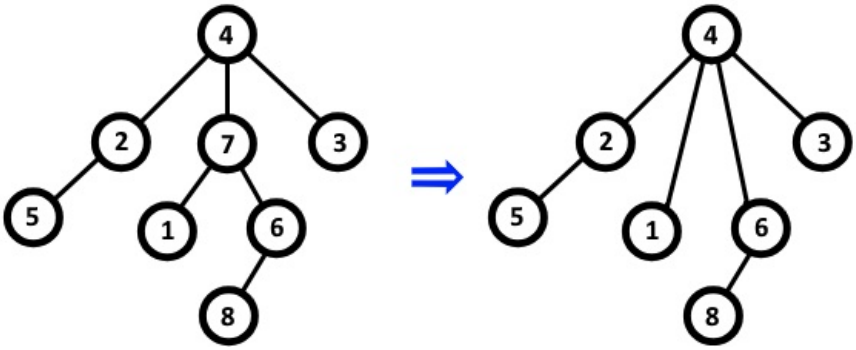
Ancestors of the vertex $i$ are all vertices on the path from the root to the vertex $i$, except the vertex $i$ itself. The parent of the vertex $i$ is the nearest to the vertex $i$ ancestor of $i$. Each vertex is a child of its parent. In the given tree the parent of the vertex $i$ is the vertex $p_i$. For the root, the value $p_i$ is $-1$.




An example of a tree with $n = 8$, the root is vertex $5$. The parent of the vertex $2$ is vertex $3$, the parent of the vertex $1$ is vertex $5$. The ancestors of the vertex $6$ are vertices $4$ and $5$, the ancestors of the vertex $7$ are vertices $8$, $3$ and $5$

You noticed that some vertices do not respect others. In particular, if $c_i = 1$, then the vertex $i$ does not respect any of its ancestors, and if $c_i = 0$, it respects all of them.

You decided to delete vertices from the tree one by one. On each step you select such a non-root vertex that it does not respect its parent and none of its children respects it. If there are several such vertices, you select the one with the **smallest number**. When you delete this vertex $v$, all children of $v$ become connected with the parent of $v$.




An example of deletion of the vertex $7$.

Once there are no vertices matching the criteria for deletion, you stop the process. Print the order in which you will delete the vertices. Note that this order is unique.

## Input

The first line contains a single integer $n$ ($1 \le n \le 10^5$) — the number of vertices in the tree.

The next $n$ lines describe the tree: the $i$-th line contains two integers $p_i$ and $c_i$ ($1 \le p_i \le n$, $0 \le c_i \le 1$), where $p_i$ is the parent of the vertex $i$, and $c_i = 0$, if the vertex $i$ respects its parents, and $c_i = 1$, if the vertex $i$ does not respect any of its parents. The root of the tree has $-1$ instead of the parent index, also, $c_i = 0$ for the root. It is guaranteed that the values $p_i$ define a rooted tree with $n$ vertices.

## Output

In case there is at least one vertex to delete, print the only line containing the indices of the vertices you will delete in the order you delete them. Otherwise print a single integer $-1$.

### Examples

| input |
| --- |
| 5<br>3 1<br>1 1<br>-1 0<br>2 1<br>3 0 |
| **output** |
| 1 2 4 |

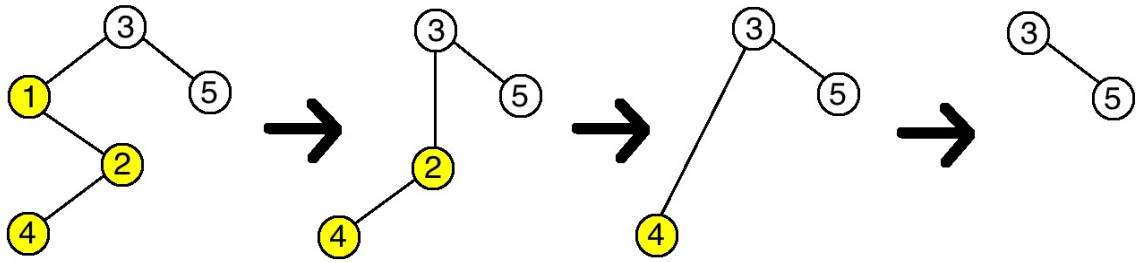
| input |
| --- |
| 5<br>-1 0<br>1 1<br>1 1<br>2 0<br>3 0 |
| **output** |
| -1 |

| input |
| --- |
| 8<br>2 1<br>-1 0<br>1 0<br>1 1<br>1 1<br>4 0<br>5 1<br>7 0 |
| **output** |
| 5 |

## Note

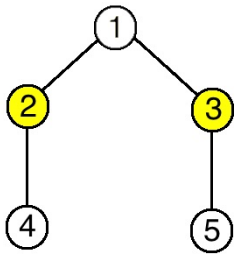
The deletion process in the first example is as follows (see the picture below, the vertices with $c_i = 1$ are in yellow):

- first you will delete the vertex $1$, because it does not respect ancestors and all its children (the vertex $2$) do not respect it, and $1$ is the smallest index among such vertices;
- the vertex $2$ will be connected with the vertex $3$ after deletion;
- then you will delete the vertex $2$, because it does not respect ancestors and all its children (the only vertex $4$) do not respect it;
- the vertex $4$ will be connected with the vertex $3$;
- then you will delete the vertex $4$, because it does not respect ancestors and all its children (there are none) do not respect it (vacuous truth);
- you will just delete the vertex $4$;
- there are no more vertices to delete.

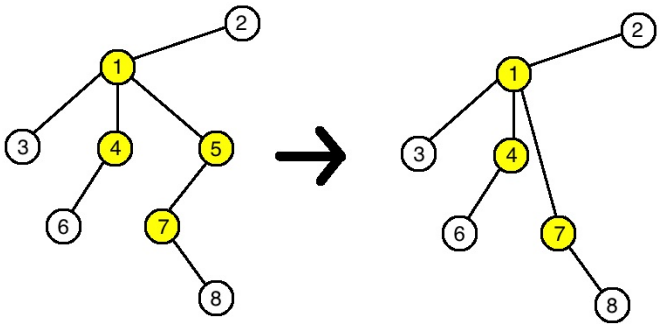



In the second example you don't need to delete any vertex:

- vertices $2$ and $3$ have children that respect them;
- vertices $4$ and $5$ respect ancestors.




In the third example the tree will change this way:




## D. The Beatles

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

Recently a Golden Circle of Beetlovers was found in Byteland. It is a circle route going through $n \cdot k$ cities. The cities are numerated from $1$ to $n \cdot k$, the distance between the neighboring cities is exactly $1$ km.

Sergey does not like beetles, he loves burgers. Fortunately for him, there are $n$ fast food restaurants on the circle, they are located in the $1$-st, the $(k+1)$-st, the $(2k+1)$-st, and so on, the $((n-1)k+1)$-st cities, i.e. the distance between the neighboring cities with fast food restaurants is $k$ km.

Sergey began his journey at some city $s$ and traveled along the circle, making stops at cities each $l$ km ($l > 0$), until he stopped in $s$ once again. Sergey then forgot numbers $s$ and $l$, but he remembers that the distance from the city $s$ to the nearest fast food restaurant was $a$ km, and the distance from the city he stopped at after traveling the first $l$ km from $s$ to the nearest fast food restaurant was $b$ km. Sergey always traveled in the same direction along the circle, but when he calculated distances to the restaurants, he considered both directions.

Now Sergey is interested in two integers. The first integer $x$ is the minimum number of stops (excluding the first) Sergey could have done before returning to $s$. The second integer $y$ is the maximum number of stops (excluding the first) Sergey could have done before returning to $s$.

### Input

The first line contains two integers $n$ and $k$ ($1 \le n, k \le 100\,000$) — the number of fast food restaurants on the circle and the distance between the neighboring restaurants, respectively.

The second line contains two integers $a$ and $b$ ($0 \le a, b \le \frac{k}{2}$) — the distances to the nearest fast food restaurants from the initial city and from the city Sergey made the first stop at, respectively.

### Output

Print the two integers $x$ and $y$.

**Examples**

| input |
| --- |
| 2 3<br>1 1 |
| **output** |
| 1 6 |

| input |
| --- |
| 3 2<br>0 0 |
| **output** |
| 1 3 |

| input |
| --- |
| 1 10<br>5 3 |
| **output** |
| 5 5 |

### Note

In the first example the restaurants are located in the cities $1$ and $4$, the initial city $s$ could be $2$, $3$, $5$, or $6$. The next city Sergey stopped at could also be at cities $2, 3, 5, 6$. Let's loop through all possible combinations of these cities. If both $s$ and the city of the first stop are at the city $2$ (for example, $l = 6$), then Sergey is at $s$ after the first stop already, so $x = 1$. In other pairs Sergey needs $1, 2, 3$, or $6$ stops to return to $s$, so $y = 6$.

In the second example Sergey was at cities with fast food restaurant both initially and after the first stop, so $l$ is $2$, $4$, or $6$. Thus $x = 1$, $y = 3$.

In the third example there is only one restaurant, so the possible locations of $s$ and the first stop are: $(6, 8)$ and $(6, 4)$. For the first option $l = 2$, for the second $l = 8$. In both cases Sergey needs $x = y = 5$ stops to go to $s$.

## E. Lynyrd Skynyrd

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

Recently Lynyrd and Skynyrd went to a shop where Lynyrd bought a permutation $p$ of length $n$, and Skynyrd bought an array $a$ of length $m$, consisting of integers from $1$ to $n$.

Lynyrd and Skynyrd became bored, so they asked you $q$ queries, each of which has the following form: "does the subsegment of $a$ from the $l$-th to the $r$-th positions, inclusive, have a subsequence that is a cyclic shift of $p$?" Please answer the queries.

A *permutation* of length $n$ is a sequence of $n$ integers such that each integer from $1$ to $n$ appears exactly once in it.

A *cyclic shift* of a permutation $(p_1, p_2, \ldots, p_n)$ is a permutation $(p_i, p_{i+1}, \ldots, p_n, p_1, p_2, \ldots, p_{i-1})$ for some $i$ from $1$ to $n$. For example, a permutation $(2, 1, 3)$ has three distinct cyclic shifts: $(2, 1, 3)$, $(1, 3, 2)$, $(3, 2, 1)$.

A *subsequence* of a subsegment of array $a$ from the $l$-th to the $r$-th positions, inclusive, is a sequence $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ for some $i_1, i_2, \ldots, i_k$ such that $l \leq i_1 < i_2 < \ldots < i_k \leq r$.

### Input

The first line contains three integers $n$, $m$, $q$ ($1 \leq n, m, q \leq 2 \cdot 10^5$) — the length of the permutation $p$, the length of the array $a$ and the number of queries.

The next line contains $n$ integers from $1$ to $n$, where the $i$-th of them is the $i$-th element of the permutation. Each integer from $1$ to $n$ appears exactly once.

The next line contains $m$ integers from $1$ to $n$, the $i$-th of them is the $i$-th element of the array $a$.

The next $q$ lines describe queries. The $i$-th of these lines contains two integers $l_i$ and $r_i$ ($1 \leq l_i \leq r_i \leq m$), meaning that the $i$-th query is about the subsegment of the array from the $l_i$-th to the $r_i$-th positions, inclusive.

### Output

Print a single string of length $q$, consisting of $0$ and $1$, the digit on the $i$-th positions should be $1$, if the subsegment of array $a$ from the $l_i$-th to the $r_i$-th positions, inclusive, contains a subsequence that is a cyclic shift of $p$, and $0$ otherwise.

**Examples**

| input |
| --- |
| 3 6 3 |

| |
|---|
| 2 1 3<br>1 2 3 1 2 3<br>1 5<br>2 6<br>3 5 |

| **output** |
|---|
| 110 |

| **input** |
|---|
| 2 4 3<br>2 1<br>1 1 2 2<br>1 2<br>2 3<br>3 4 |

| **output** |
|---|
| 010 |

### Note

In the first example the segment from the $1$-st to the $5$-th positions is $1, 2, 3, 1, 2$. There is a subsequence $1, 3, 2$ that is a cyclic shift of the permutation. The subsegment from the $2$-nd to the $6$-th positions also contains a subsequence $2, 1, 3$ that is equal to the permutation. The subsegment from the $3$-rd to the $5$-th positions is $3, 1, 2$, there is only one subsequence of length $3$ ($3, 1, 2$), but it is not a cyclic shift of the permutation.

In the second example the possible cyclic shifts are $1, 2$ and $2, 1$. The subsegment from the $1$-st to the $2$-nd positions is $1, 1$, its subsequences are not cyclic shifts of the permutation. The subsegment from the $2$-nd to the $3$-rd positions is $1, 2$, it coincides with the permutation. The subsegment from the $3$ to the $4$ positions is $2, 2$, its subsequences are not cyclic shifts of the permutation.

## F. U2

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

Recently Vasya learned that, given two points with different $x$ coordinates, you can draw through them exactly one parabola with equation of type $y = x^2 + bx + c$, where $b$ and $c$ are reals. Let's call such a parabola an $U$-shaped one.

Vasya drew several distinct points with integer coordinates on a plane and then drew an $U$-shaped parabola through each pair of the points that have different $x$ coordinates. The picture became somewhat messy, but Vasya still wants to count how many of the parabolas drawn don't have any drawn point inside their internal area. Help Vasya.

The internal area of an $U$-shaped parabola is the part of the plane that lies strictly above the parabola when the $y$ axis is directed upwards.

### Input

The first line contains a single integer $n$ ($1 \le n \le 100\,000$) — the number of points.

The next $n$ lines describe the points, the $i$-th of them contains two integers $x_i$ and $y_i$ — the coordinates of the $i$-th point. It is guaranteed that all points are distinct and that the coordinates do not exceed $10^6$ by absolute value.

### Output

In the only line print a single integer — the number of $U$-shaped parabolas that pass through at least two of the given points and do not contain any of the given points inside their internal area (excluding the parabola itself).

#### Examples

| **input** |
|---|
| 3<br>-1 0<br>0 2<br>1 0 |

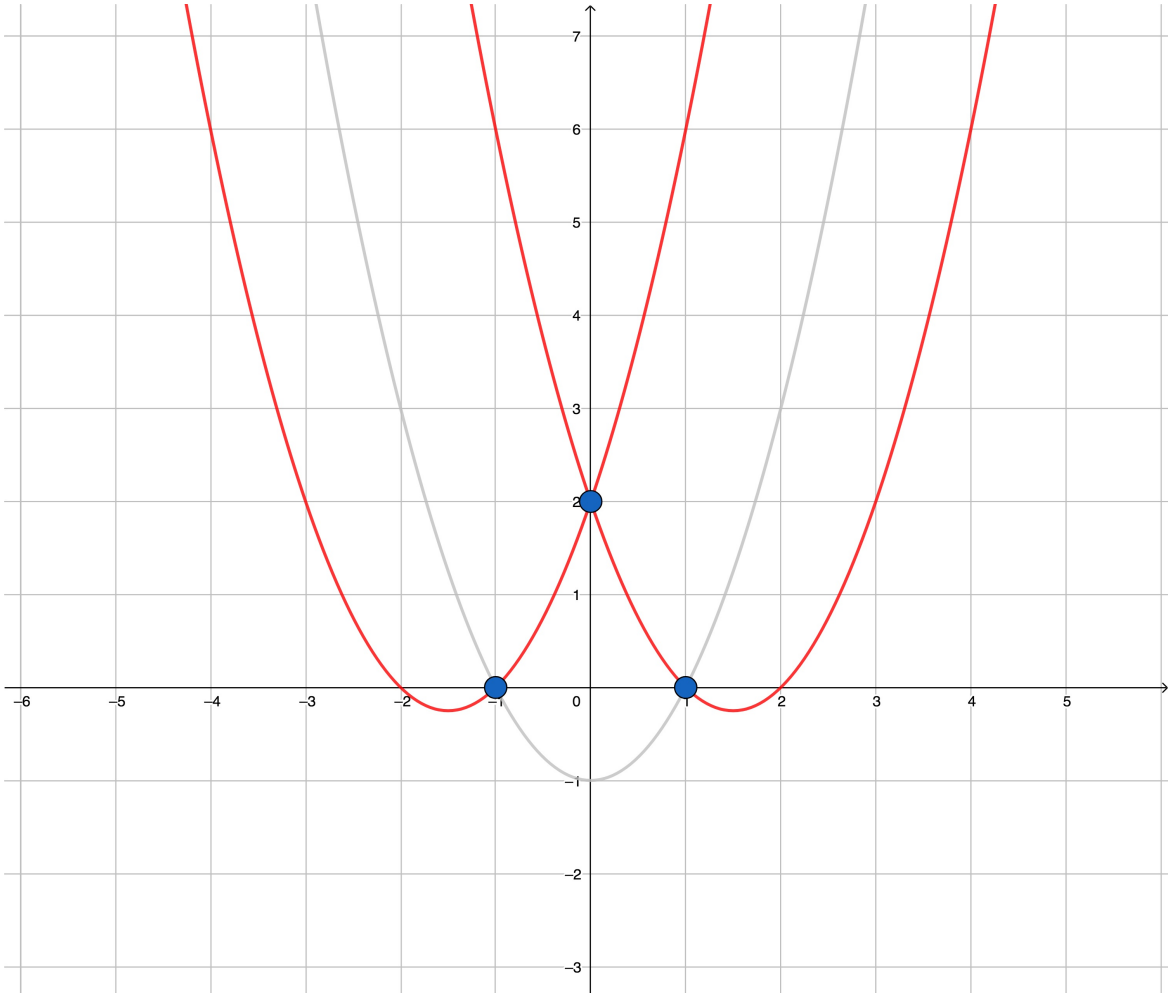
| **output** |
|---|
| 2 |

| **input** |
|---|
| 5<br>1 0<br>1 -1<br>0 -1<br>-1 0<br>-1 -1 |

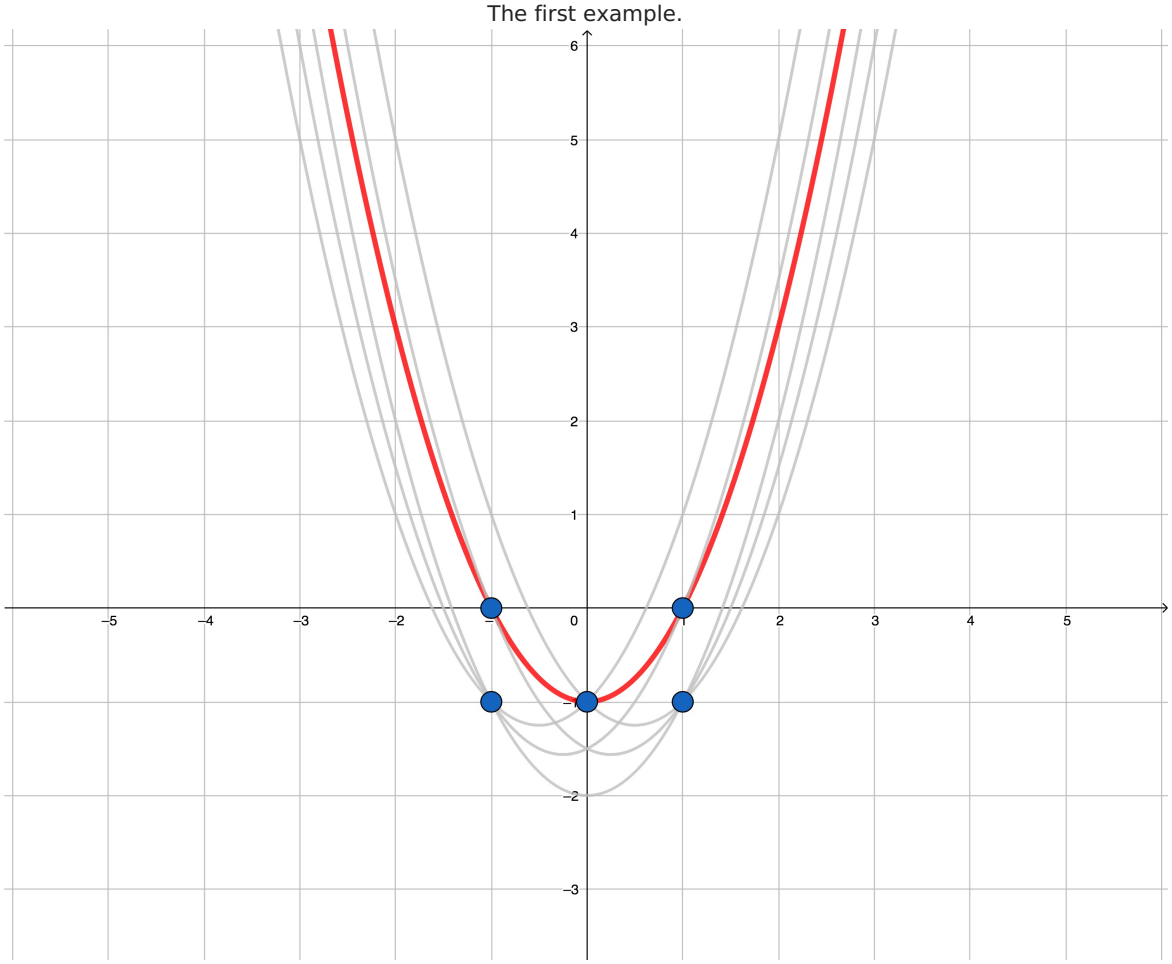
| **output** |
|---|
| 1 |

**Note**

On the pictures below all $U$-shaped parabolas that pass through at least two given points are drawn for each of the examples. The $U$-shaped parabolas that do not have any given point inside their internal area are drawn in red.




The first example.




The second example.

The only programming contests Web 2.0 platform