A. Adding Digits

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

Vasya has got two number: a and b. However, Vasya finds number a too short. So he decided to repeat the operation of lengthening number a n times.

One operation of lengthening a number means adding exactly one digit to the number (in the decimal notation) to the right provided that the resulting number is divisible by Vasya's number b. If it is impossible to obtain the number which is divisible by b, then the lengthening operation cannot be performed.

Your task is to help Vasya and print the number he can get after applying the lengthening operation to number a n times.

Input

The first line contains three integers: a, b, n $(1 \le a, b, n \le 10^5)$.

Output

In a single line print the integer without leading zeros, which Vasya can get when he applies the lengthening operations to number a n times. If no such number exists, then print number -1. If there are multiple possible answers, print any of them.

Sample test(s)

input
```
5 4 5
```
output
```
524848
```

input
```
12 11 1
```
output
```
121
```

input
```
260 150 10
```
output
```
-1
```

B. Ancient Prophesy

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

A recently found Ancient Prophesy is believed to contain the exact Apocalypse date. The prophesy is a string that only consists of digits and characters "`-`".

We'll say that some date is mentioned in the Prophesy if there is a substring in the Prophesy that is the date's record in the format "`dd-mm-yyyy`". We'll say that the number of the date's occurrences is the number of such substrings in the Prophesy. For example, the Prophesy "`0012-10-2012-10-2012`" mentions date `12-10-2012` twice (first time as "`00`**`12-10-2012`**`-10-2012`", second time as "`0012-10-20`**`12-10-2012`**").

The date of the Apocalypse is such correct date that the number of times it is mentioned in the Prophesy is strictly larger than that of any other correct date.

A date is correct if the year lies in the range from 2013 to 2015, the month is from 1 to 12, and the number of the day is strictly more than a zero and doesn't exceed the number of days in the current month. Note that a date is written in the format "`dd-mm-yyyy`", that means that leading zeroes may be added to the numbers of the months or days if needed. In other words, date "`1-1-2013`" isn't recorded in the format "`dd-mm-yyyy`", and date "`01-01-2013`" is recorded in it.

Notice, that any year between 2013 and 2015 is not a leap year.

Input
The first line contains the Prophesy: a non-empty string that only consists of digits and characters "`-`". The length of the Prophesy doesn't exceed 10^5 characters.

Output
In a single line print the date of the Apocalypse. It is guaranteed that such date exists and is unique.

Sample test(s)

input
```
777-444---21-12-2013-12-2013-12-2013---444-777
```
output
```
13-12-2013
```

C. Balls and Boxes

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

Little Vasya had n boxes with balls in the room. The boxes stood in a row and were numbered with numbers from 1 to n from left to right.

Once Vasya chose one of the boxes, let's assume that its number is i, took all balls out from it (it is guaranteed that this box originally had at least one ball), and began putting balls (one at a time) to the boxes with numbers $i+1$, $i+2$, $i+3$ and so on. If Vasya puts a ball into the box number n, then the next ball goes to box 1, the next one goes to box 2 and so on. He did it until he had no balls left in his hands. It is possible that Vasya puts multiple balls to the same box, and it is also possible that one or more balls will go to the box number i. If $i = n$, Vasya puts the first ball into the box number 1, then the next ball goes to box 2 and so on.

For example, let's suppose that initially Vasya had four boxes, and the first box had 3 balls, the second one had 2, the third one had 5 and the fourth one had 4 balls. Then, if $i = 3$, then Vasya will take all five balls out of the third box and put them in the boxes with numbers: $4, 1, 2, 3, 4$. After all Vasya's actions the balls will lie in the boxes as follows: in the first box there are 4 balls, 3 in the second one, 1 in the third one and 6 in the fourth one.

At this point Vasya has completely forgotten the original arrangement of the balls in the boxes, but he knows how they are arranged now, and the number x — the number of the box, where he put the last of the taken out balls.

He asks you to help to find the initial arrangement of the balls in the boxes.

Input

The first line of the input contains two integers n and x ($2 \le n \le 10^5$, $1 \le x \le n$), that represent the number of the boxes and the index of the box that got the last ball from Vasya, correspondingly. The second line contains n space-separated integers $a_1, a_2, ..., a_n$, where integer a_i ($0 \le a_i \le 10^9$, $a_x \ne 0$) represents the number of balls in the box with index i after Vasya completes all the actions.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the cin, cout streams or the `%I64d` specifier.

Output

Print n integers, where the i-th one represents the number of balls in the box number i before Vasya starts acting. Separate the numbers in the output by spaces. If there are multiple correct solutions, you are allowed to print any of them.

Sample test(s)

input
```
4 4
4 3 1 6
```
output
```
3 2 5 4
```

input
```
5 2
3 2 0 2 7
```
output
```
2 1 4 1 6
```

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

D. Black and White Tree

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

The board has got a painted tree graph, consisting of n nodes. Let us remind you that a non-directed graph is called a tree if it is connected and doesn't contain any cycles.

Each node of the graph is painted black or white in such a manner that there aren't two nodes of the same color, connected by an edge. Each edge contains its value written on it as a non-negative integer.

A bad boy Vasya came up to the board and wrote number s_v near each node v — the sum of values of all edges that are incident to this node. Then Vasya removed the edges and their values from the board.

Your task is to restore the original tree by the node colors and numbers s_v.

Input

The first line of the input contains a single integer n $(2 \le n \le 10^5)$ — the number of nodes in the tree. Next n lines contain pairs of space-separated integers c_i, s_i $(0 \le c_i \le 1, 0 \le s_i \le 10^9)$, where c_i stands for the color of the i-th vertex (0 is for white, 1 is for black), and s_i represents the sum of values of the edges that are incident to the i-th vertex of the tree that is painted on the board.

Output

Print the description of $n - 1$ edges of the tree graph. Each description is a group of three integers v_i, u_i, w_i $(1 \le v_i, u_i \le n, v_i \ne u_i, 0 \le w_i \le 10^9)$, where v_i and u_i — are the numbers of the nodes that are connected by the i-th edge, and w_i is its value. Note that the following condition must fulfill $c_{v_i} \ne c_{u_i}$.

It is guaranteed that for any input data there exists at least one graph that meets these data. If there are multiple solutions, print any of them. You are allowed to print the edges in any order. As you print the numbers, separate them with spaces.

Sample test(s)

input

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

output

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

input

```
6
1 0
0 3
1 8
0 2
0 3
0 0
```

output

```
2 3 3
5 3 3
4 3 2
1 6 0
2 1 0
```

E. Dividing Kingdom

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

A country called Flatland is an infinite two-dimensional plane. Flatland has n cities, each of them is a point on the plane.

Flatland is ruled by king Circle IV. Circle IV has 9 sons. He wants to give each of his sons part of Flatland to rule. For that, he wants to draw four **distinct** straight lines, such that two of them are parallel to the Ox axis, and two others are parallel to the Oy axis. At that, no straight line can go through any city. Thus, Flatland will be divided into 9 parts, and each son will be given exactly one of these parts. Circle IV thought a little, evaluated his sons' obedience and decided that the i-th son should get the part of Flatland that has exactly a_i cities.

Help Circle find such four straight lines that if we divide Flatland into 9 parts by these lines, the resulting parts can be given to the sons so that son number i got the part of Flatland which contains a_i cities.

Input

The first line contains integer n $(9 \le n \le 10^5)$ — the number of cities in Flatland. Next n lines each contain two space-separated integers: x_i, y_i $(-10^9 \le x_i, y_i \le 10^9)$ — the coordinates of the i-th city. No two cities are located at the same point. The last line contains nine space-separated integers: $a_1, a_2, \cdots, a_9$ $(1 \le a_i \le 10^5)$, $\sum_{i=1}^{9} a_i = n$.

Output

If there is no solution, print a single integer -1.

Otherwise, print in the first line two distinct real space-separated numbers: x_1, x_2 — the abscissas of the straight lines that are parallel to the Oy axis. And in the second line print two distinct real space-separated numbers: y_1, y_2 — the ordinates of the straight lines, parallel to the Ox. If there are multiple solutions, print any of them.

When the answer is being checked, a city is considered to lie on a straight line, if the distance between the city and the line doesn't exceed 10^{-6}. Two straight lines are considered the same if the distance between them doesn't exceed 10^{-6}.

Sample test(s)

input

```
9
1 1
1 2
1 3
2 1
2 2
2 3
3 1
3 2
3 3
1 1 1 1 1 1 1 1 1
```

output

```
1.5000000000 2.5000000000
1.5000000000 2.5000000000
```

input

```
15
4 4
-1 -3
1 5
3 -4
-4 4
-1 1
3 -3
-4 -5
-3 3
3 2
4 1
-4 2
-2 -5
-3 4
-1 4
2 1 2 1 2 1 3 2 1
```

output

```
-3.5000000000 2.0000000000
3.5000000000 -1.0000000000
```

input

```
10
-2 10
6 0
-16 -6
```

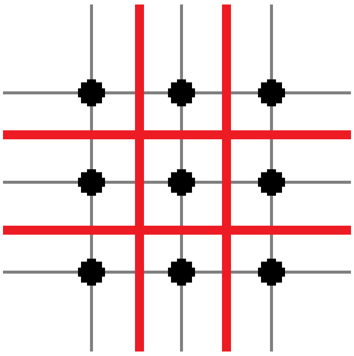
```
-4 13
-4 -2
-17 -10
9 15
18 16
-5 2
10 -5
2 1 1 1 1 1 1 1 1
```

output

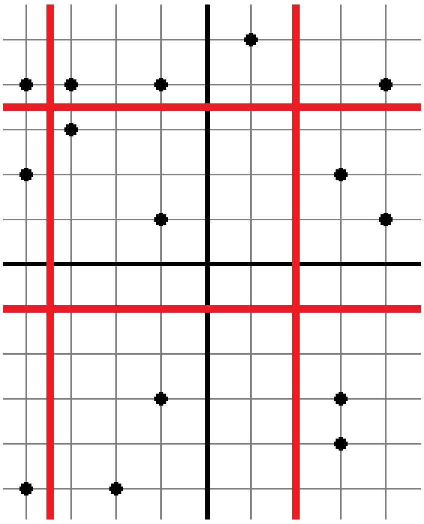
```
-1
```

Note

The solution for the first sample test is shown below:



The solution for the second sample test is shown below:



There is no solution for the third sample test.