Codeforces Round #392 (Div. 2)

A. Holiday Of Equality

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

In Berland it is the holiday of equality. In honor of the holiday the king decided to equalize the welfare of all citizens in Berland by the expense of the state treasury.

Totally in Berland there are n citizens, the welfare of each of them is estimated as the integer in a_i burles (burle is the currency in Berland).

You are the royal treasurer, which needs to count the minimum charges of the kingdom on the king's present. The king can only give money, he hasn't a power to take away them.

Input

The first line contains the integer n $(1 \le n \le 100)$ — the number of citizens in the kingdom.

The second line contains n integers $a_1, a_2, ..., a_n$, where a_i $(0 \le a_i \le 10^6)$ — the welfare of the i-th citizen.

Output

In the only line print the integer S — the minimum number of burles which are had to spend.

Examples

input

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

output

```
10
```

input

```
5
1 1 0 1 1
```

output

```
1
```

input

```
3
1 3 1
```

output

```
4
```

input

```
1
12
```

output

```
0
```

Note

In the first example if we add to the first citizen 4 burles, to the second 3, to the third 2 and to the fourth 1, then the welfare of all citizens will equal 4.

In the second example it is enough to give one burle to the third citizen.

In the third example it is necessary to give two burles to the first and the third citizens to make the welfare of citizens equal 3.

In the fourth example it is possible to give nothing to everyone because all citizens have 12 burles.

B. Blown Garland

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

Nothing is eternal in the world, Kostya understood it on the 7-th of January when he saw partially dead four-color garland.

Now he has a goal to replace dead light bulbs, however he doesn't know how many light bulbs for each color are required. It is guaranteed that for each of four colors at least one light is working.

It is known that the garland contains light bulbs of four colors: red, blue, yellow and green. The garland is made as follows: if you take any four consecutive light bulbs then there will not be light bulbs with the same color among them. For example, the garland can look like "RYBGRYBGRY", "YBGRYBGRYBG", "BGRYB", but can not look like "BGRYG", "YBGRYBYGR" or "BGYBGY". Letters denote colors: 'R' — red, 'B' — blue, 'Y' — yellow, 'G' — green.

Using the information that for each color at least one light bulb still works count the number of dead light bulbs of each four colors.

Input

The first and the only line contains the string s ($4 \le |s| \le 100$), which describes the garland, the i-th symbol of which describes the color of the i-th light bulb in the order from the beginning of garland:

- 'R' — the light bulb is red,
- 'B' — the light bulb is blue,
- 'Y' — the light bulb is yellow,
- 'G' — the light bulb is green,
- '!' — the light bulb is dead.

The string s can not contain other symbols except those five which were described.

It is guaranteed that in the given string at least once there is each of four letters 'R', 'B', 'Y' and 'G'.

It is guaranteed that the string s is correct garland with some blown light bulbs, it means that for example the line "GRBY!!!B" can not be in the input data.

Output

In the only line print four integers k_r, k_b, k_y, k_g — the number of dead light bulbs of red, blue, yellow and green colors accordingly.

Examples

input
```
RYBGRYBGR
```
output
```
0 0 0 0
```

input
```
!RGYB
```
output
```
0 1 0 0
```

input
```
!!!!YGRB
```
output
```
1 1 1 1
```

input
```
!GB!RG!Y!
```
output
```
2 1 1 0
```

Note

In the first example there are no dead light bulbs.

In the second example it is obvious that one blue bulb is blown, because it could not be light bulbs of other colors on its place according to the statements.

C. Unfair Poll

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

On the Literature lesson Sergei noticed an awful injustice, it seems that some students are asked more often than others.

Seating in the class looks like a rectangle, where n rows with m pupils in each.

The teacher asks pupils in the following order: at first, she asks all pupils from the first row in the order of their seating, then she continues to ask pupils from the next row. If the teacher asked the last row, then the direction of the poll changes, it means that she asks the previous row. The order of asking the rows looks as follows: the 1-st row, the 2-nd row, ..., the $n - 1$-st row, the n-th row, the $n - 1$-st row, ..., the 2-nd row, the 1-st row, the 2-nd row, ...

The order of asking of pupils on the same row is always the same: the 1-st pupil, the 2-nd pupil, ..., the m-th pupil.

During the lesson the teacher managed to ask exactly k questions from pupils in order described above. Sergei seats on the x-th row, on the y-th place in the row. Sergei decided to prove to the teacher that pupils are asked irregularly, help him count three values:

1. the maximum number of questions a particular pupil is asked,
2. the minimum number of questions a particular pupil is asked,
3. how many times the teacher asked Sergei.

If there is only one row in the class, then the teacher always asks children from this row.

Input

The first and the only line contains five integers n, m, k, x and y ($1 \leq n, m \leq 100, 1 \leq k \leq 10^{18}, 1 \leq x \leq n, 1 \leq y \leq m$).

Output

Print three integers:

1. the maximum number of questions a particular pupil is asked,
2. the minimum number of questions a particular pupil is asked,
3. how many times the teacher asked Sergei.

Examples

input
```
1 3 8 1 1
```
output
```
3 2 3
```

input
```
4 2 9 4 2
```
output
```
2 1 1
```

input
```
5 5 25 4 3
```
output
```
1 1 1
```

input
```
100 100 1000000000000000000 100 100
```
output
```
101010101010101 50505050505051 50505050505051
```

Note

The order of asking pupils in the first test:

1. the pupil from the first row who seats at the first table, it means it is Sergei;
2. the pupil from the first row who seats at the second table;
3. the pupil from the first row who seats at the third table;
4. the pupil from the first row who seats at the first table, it means it is Sergei;
5. the pupil from the first row who seats at the second table;
6. the pupil from the first row who seats at the third table;
7. the pupil from the first row who seats at the first table, it means it is Sergei;
8. the pupil from the first row who seats at the second table;

The order of asking pupils in the second test:

1. the pupil from the first row who seats at the first table;
2. the pupil from the first row who seats at the second table;
3. the pupil from the second row who seats at the first table;
4. the pupil from the second row who seats at the second table;
5. the pupil from the third row who seats at the first table;
6. the pupil from the third row who seats at the second table;
7. the pupil from the fourth row who seats at the first table;
8. the pupil from the fourth row who seats at the second table, it means it is Sergei;
9. the pupil from the third row who seats at the first table;

D. Ability To Convert

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

Alexander is learning how to convert numbers from the decimal system to any other, however, he doesn't know English letters, so he writes any number only as a decimal number, it means that instead of the letter A he will write the number 10. Thus, by converting the number 475 from decimal to hexadecimal system, he gets 11311 ($475 = 1 \cdot 16^2 + 13 \cdot 16^1 + 11 \cdot 16^0$). Alexander lived calmly until he tried to convert the number back to the decimal number system.

Alexander remembers that he worked with little numbers so he asks to find the minimum decimal number so that by converting it to the system with the base n he will get the number k.

Input

The first line contains the integer n ($2 \le n \le 10^9$). The second line contains the integer k ($0 \le k < 10^{60}$), it is guaranteed that the number k contains no more than 60 symbols. All digits in the second line are strictly less than n.

Alexander guarantees that the answer exists and does not exceed 10^{18}.

The number k doesn't contain leading zeros.

Output

Print the number x ($0 \le x \le 10^{18}$) — the answer to the problem.

Examples

input

```
13
12
```

output

```
12
```

input

```
16
11311
```

output

```
475
```

input

```
20
999
```

output

```
3789
```

input

```
17
2016
```

output

```
594
```

Note

In the first example 12 could be obtained by converting two numbers to the system with base 13: $12 = 12 \cdot 13^0$ or $15 = 1 \cdot 13^1 + 2 \cdot 13^0$.

E. Broken Tree

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

You are given a tree that has n vertices, which are numbered from 1 to n, where the vertex number one is the root. Each edge has weight w_i and strength p_i.

Botanist Innokentiy, who is the only member of the jury of the Olympiad in Informatics, doesn't like broken trees.

The tree is broken if there is such an edge the strength of which is less than the sum of weight of subtree's edges to which it leads.

It is allowed to reduce weight of any edge by arbitrary integer value, but then the strength of its edge is reduced by the same value. It means if the weight of the edge is 10, and the strength is 12, then by the reducing the weight by 7 its weight will equal 3, and the strength will equal 5.

It is not allowed to increase the weight of the edge.

Your task is to get the tree, which is not broken, by reducing the weight of edges of the given tree, and also all edged should have the positive weight, moreover, the total weight of all edges should be as large as possible.

It is obvious that the strength of edges can not be negative, however it can equal zero if the weight of the subtree equals zero.

Input

The first line contains the integer n ($1 \le n \le 2 \cdot 10^5$) — the number of vertices in the tree. The next n - 1 lines contains the description of edges. Each line contains four integers x, y, w, p ($1 \le x, y \le n$, $1 \le w \le 10^9$, $0 \le p \le 10^9$), where x and y — vertices which connect the edge (the vertex number x is the parent of the vertex number y), w and p are the weight and the strength of the edge, accordingly. It is guaranteed that the edges describe the tree with the root in the vertex 1.

Output

If it is impossible to get unbroken tree from the given tree, print `-1` in the only line.

Otherwise, the output data should contain n lines:

In the first line print the number n — the number of vertices on the tree.

In the next n - 1 lines print the description of edges of the resulting tree. Each line should contain four integers x, y, w, p ($1 \le x, y \le n$, $1 \le w \le 10^9$, $0 \le p \le 10^9$), where x and y — vertices, which the edge connects (the vertex number x is the parent of the vertex number y), w and p are the new weight and the strength of the edge, accordingly.

Print edges in the same order as they are given in input data: the first two integers of each line should not be changed.

Examples

input

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

output

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

input

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

output

```
-1
```

input

```
5
1 2 2 4
2 4 1 9
4 5 5 6
4 3 4 8
```

output

```
5
1 2 2 4
2 4 1 9
4 5 1 2
4 3 2 6
```

input

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

output

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

F. Geometrical Progression

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

For given n, l and r find the number of distinct geometrical progression, each of which contains n distinct integers not less than l and not greater than r. In other words, for each progression the following must hold: $l \le a_i \le r$ and $a_i \ne a_j$, where $a_1, a_2, ..., a_n$ is the geometrical progression, $1 \le i, j \le n$ and $i \ne j$.

Geometrical progression is a sequence of numbers $a_1, a_2, ..., a_n$ where each term after first is found by multiplying the previous one by a fixed non-zero number d called the common ratio. Note that in our task d may be non-integer. For example in progression $4, 6, 9$, common ratio is .

Two progressions $a_1, a_2, ..., a_n$ and $b_1, b_2, ..., b_n$ are considered different, if there is such i ($1 \le i \le n$) that $a_i \ne b_i$.

Input

The first and the only line cotains three integers n, l and r ($1 \le n \le 10^7$, $1 \le l \le r \le 10^7$).

Output

Print the integer K — is the answer to the problem.

Examples

input
```
1 1 10
```
output
```
10
```

input
```
2 6 9
```
output
```
12
```

input
```
3 1 10
```
output
```
8
```

input
```
3 3 10
```
output
```
2
```

Note

These are possible progressions for the first test of examples:

- 1;
- 2;
- 3;
- 4;
- 5;
- 6;
- 7;
- 8;
- 9;
- 10.

These are possible progressions for the second test of examples:

- 6, 7;
- 6, 8;
- 6, 9;
- 7, 6;
- 7, 8;
- 7, 9;
- 8, 6;
- 8, 7;

- $8, 9$;
- $9, 6$;
- $9, 7$;
- $9, 8$.

These are possible progressions for the third test of examples:

- $1, 2, 4$;
- $1, 3, 9$;
- $2, 4, 8$;
- $4, 2, 1$;
- $4, 6, 9$;
- $8, 4, 2$;
- $9, 3, 1$;
- $9, 6, 4$.

These are possible progressions for the fourth test of examples:

- $4, 6, 9$;
- $9, 6, 4$.