Codeforces Beta Round 5

A. Chat Server's Outgoing Traffic

1 second, 64 megabytes

Polycarp is working on a new project called "Polychat". Following modern tendencies in IT, he decided, that this project should contain chat as well. To achieve this goal, Polycarp has spent several hours in front of his laptop and implemented a chat server that can process three types of commands:

- Include a person to the chat (`'Add'` command).
- Remove a person from the chat (`'Remove'` command).
- Send a message from a person to all people, who are currently in the chat, including the one, who sends the message (`'Send'` command).

Now Polycarp wants to find out the amount of outgoing traffic that the server will produce while processing a particular set of commands.

Polycarp knows that chat server sends no traffic for 'Add' and 'Remove' commands. When 'Send' command is processed, server sends l bytes to each participant of the chat, where l is the length of the message.

As Polycarp has no time, he is asking for your help in solving this problem.

Input

Input file will contain not more than 100 commands, each in its own line. No line will exceed 100 characters. Formats of the commands will be the following:

- `+<name>` for `'Add'` command.
- `-<name>` for `'Remove'` command.
- `<sender_name>:<message_text>` for `'Send'` command.

`<name>` and `<sender_name>` is a non-empty sequence of Latin letters and digits. `<message_text>` can contain letters, digits and spaces, but can't start or end with a space. `<message_text>` can be an empty line.

It is guaranteed, that input data are correct, i.e. there will be no `'Add'` command if person with such a name is already in the chat, there will be no `'Remove'` command if there is no person with such a name in the chat etc.

All names are case-sensitive.

Output

Print a single number — answer to the problem.

input

```
+Mike
Mike:hello
+Kate
+Dmitry
-Dmitry
Kate:hi
-Kate
```

output

```
9
```

input

```
+Mike
-Mike
+Mike
Mike:Hi   I am here
-Mike
+Kate
-Kate
```

output

```
14
```

B. Center Alignment

1 second, 64 megabytes

Almost every text editor has a built-in function of center text alignment. The developers of the popular in Berland text editor «Textpad» decided to introduce this functionality into the fourth release of the product.

You are to implement the alignment in the shortest possible time. Good luck!

Input

The input file consists of one or more lines, each of the lines contains Latin letters, digits and/or spaces. The lines cannot start or end with a space. It is guaranteed that at least one of the lines has positive length. The length of each line and the total amount of the lines do not exceed 1000.

Output

Format the given text, aligning it center. Frame the whole text with characters «*» of the minimum size. If a line cannot be aligned perfectly (for example, the line has even length, while the width of the block is uneven), you should place such lines rounding down the distance to the left or to the right edge and bringing them closer left or right alternatively (you should start with bringing left). Study the sample tests carefully to understand the output format better.

input

```
This  is

Codeforces
Beta
Round
5
```

output

```
************
* This  is *
*          *
*Codeforces*
*   Beta   *
*  Round   *
*     5    *
************
```

input

```
welcome to the
Codeforces
Beta
Round 5

and
good luck
```

output

```
****************
*welcome to the*
*  Codeforces  *
*     Beta     *
*   Round 5    *
*              *
*      and     *
*  good luck   *
****************
```

C. Longest Regular Bracket Sequence

2 seconds, 256 megabytes

This is yet another problem dealing with regular bracket sequences.

We should remind you that a bracket sequence is called regular, if by inserting «+» and «1» into it we can get a correct mathematical expression. For example, sequences «(())()», «()» and «(()(()))» are regular, while «)(», «(()» and «(())))(» are not.

You are given a string of «(» and «)» characters. You are to find its longest substring that is a regular bracket sequence. You are to find the number of such substrings as well.

Input
The first line of the input file contains a non-empty string, consisting of «(» and «)» characters. Its length does not exceed 10^6.

Output
Print the length of the longest substring that is a regular bracket sequence, and the number of such substrings. If there are no such substrings, write the only line containing "0 1".

input

```
)(((((())(()))
```

output

```
6 2
```

input

```
)((
```

output

```
0 1
```

D. Follow Traffic Rules

1 second, 64 megabytes

Everybody knows that the capital of Berland is connected to Bercouver (the Olympic capital) by a direct road. To improve the road's traffic capacity, there was placed just one traffic sign, limiting the maximum speed. Traffic signs in Berland are a bit peculiar, because they limit the speed only at that point on the road where they are placed. Right after passing the sign it is allowed to drive at any speed.

It is known that the car of an average Berland citizen has the acceleration (deceleration) speed of a km/h^2, and has maximum speed of v km/h. The road has the length of l km, and the speed sign, limiting the speed to w km/h, is placed d km ($1 \le d < l$) away from the capital of Berland. The car has a zero speed at the beginning of the journey. Find the minimum time that an average Berland citizen will need to get from the capital to Bercouver, if he drives at the optimal speed.

The car can enter Bercouver at any speed.

Input
The first line of the input file contains two integer numbers a and v ($1 \le a, v \le 10000$). The second line contains three integer numbers l, d and w ($2 \le l \le 10000$; $1 \le d < l$; $1 \le w \le 10000$).

Output
Print the answer with at least five digits after the decimal point.

input

```
1 1
2 1 3
```

output

```
2.500000000000
```

input

```
5 70
200 170 40
```

output

```
8.965874696353
```

E. Bindian Signalizing

4 seconds, 256 megabytes

Everyone knows that long ago on the territory of present-day Berland there lived Bindian tribes. Their capital was surrounded by n hills, forming a circle. On each hill there was a watchman, who watched the neighbourhood day and night.

In case of any danger the watchman could make a fire on the hill. One watchman could see the signal of another watchman, if on the circle arc connecting the two hills there was no hill higher than any of the two. As for any two hills there are two different circle arcs connecting them, the signal was seen if the above mentioned condition was satisfied on at least one of the arcs. For example, for any two neighbouring watchmen it is true that the signal of one will be seen by the other.

An important characteristics of this watch system was the amount of pairs of watchmen able to see each other's signals. You are to find this amount by the given heights of the hills.

Input
The first line of the input data contains an integer number n ($3 \le n \le 10^6$), n — the amount of hills around the capital. The second line contains n numbers — heights of the hills in clockwise order. All height numbers are integer and lie between 1 and 10^9.

Output
Print the required amount of pairs.

input

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

output

```
7
```