A. Candies

time limit per test: 1 second
memory limit per test: 256 megabytes

After passing a test, Vasya got himself a box of n candies. He decided to eat an equal amount of candies each morning until there are no more candies. However, Petya also noticed the box and decided to get some candies for himself.

This means the process of eating candies is the following: in the beginning Vasya chooses a single integer k, same for all days. After that, in the morning he eats k candies from the box (if there are less than k candies in the box, he eats them all), then in the evening Petya eats 10% of the candies **remaining** in the box. If there are still candies left in the box, the process repeats — next day Vasya eats k candies again, and Petya — 10% of the candies left in a box, and so on.

If the amount of candies in the box is not divisible by 10, Petya rounds the amount he takes from the box down. For example, if there were 97 candies in the box, Petya would eat only 9 of them. In particular, if there are less than 10 candies in a box, Petya won't eat any at all.

Your task is to find out the minimal amount of k that can be chosen by Vasya so that he would eat at least half of the n candies he initially got. Note that the number k must be integer.

Input

The first line contains a single integer n ($1 \leq n \leq 10^{18}$) — the initial amount of candies in the box.

Output

Output a single integer — the minimal amount of k that would allow Vasya to eat at least half of candies he got.

Example

input	output
68	3

Note

In the sample, the amount of candies, with $k = 3$, would change in the following way (Vasya eats first):

$68 \to 65 \to 59 \to 56 \to 51 \to 48 \to 44 \to 41$
$\to 37 \to 34 \to 31 \to 28 \to 26 \to 23 \to 21 \to 18 \to 17 \to 14$
$\to 13 \to 10 \to 9 \to 6 \to 6 \to 3 \to 3 \to 0$.

In total, Vasya would eat 39 candies, while Petya — 29.

B. Mr. Bender and Square

time limit per test: 2 seconds
memory limit per test: 256 megabytes

Mr. Bender has a digital table of size $n \times n$, each cell can be switched on or off. He wants the field to have at least c switched on squares. When this condition is fulfilled, Mr Bender will be happy.

We'll consider the table rows numbered from top to bottom from 1 to n, and the columns — numbered from left to right from 1 to n. Initially there is exactly one switched on cell with coordinates (x, y) (x is the row number, y is the column number), and all other cells are switched off. Then each second we switch on the cells that are off but have the side-adjacent cells that are on.

For a cell with coordinates (x, y) the side-adjacent cells are cells with coordinates $(x - 1, y)$, $(x + 1, y)$, $(x, y - 1)$, $(x, y + 1)$.

In how many seconds will Mr. Bender get happy?

Input

The first line contains four space-separated integers n, x, y, c $(1 \le n, c \le 10^9; 1 \le x, y \le n; c \le n^2)$.

Output

In a single line print a single integer — the answer to the problem.

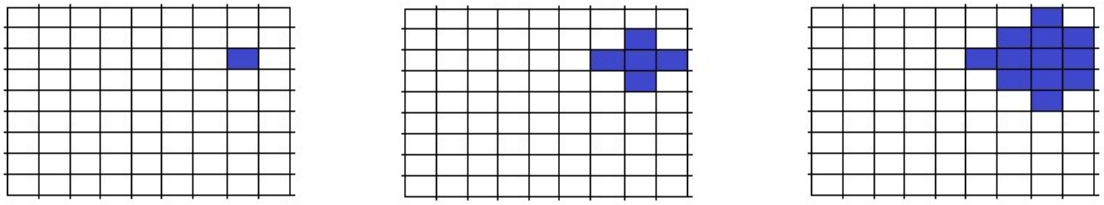
Examples

input	output
6 4 3 1	0

input	output
9 3 8 10	2

Note

Initially the first test has one painted cell, so the answer is 0. In the second test all events will go as is shown on the figure.



.

C. Degenerate Matrix

time limit per test: 1 second
memory limit per test: 256 megabytes

The *determinant* of a matrix 2×2 is defined as follows:

$$\det \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$

A matrix is called *degenerate* if its determinant is equal to zero.

The *norm* $||A||$ of a matrix A is defined as a maximum of absolute values of its elements.

You are given a matrix $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Consider any degenerate matrix B such that norm $||A - B||$ is minimum possible. Determine $||A - B||$.

Input

The first line contains two integers a and b ($|a|, |b| \le 10^9$), the elements of the first row of matrix A.

The second line contains two integers c and d ($|c|, |d| \le 10^9$) the elements of the second row of matrix A.

Output

Output a single real number, the minimum possible value of $||A - B||$. Your answer is considered to be correct if its absolute or relative error does not exceed 10^{-9}.

Examples

input	output
1 2 3 4	0.2000000000

input	output
1 0 0 1	0.5000000000

Note

In the first sample matrix B is $\begin{pmatrix} 1.2 & 1.8 \\ 2.8 & 4.2 \end{pmatrix}$

In the second sample matrix B is $\begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$

D. Buoys

time limit per test: 1 second
memory limit per test: 256 megabytes

The swimming area of Berhattan's city beach is marked out with n buoys. The buoys form a straight line. When the buoys were being put into the water, nobody cared to observe the same distance between each pair of adjacent buoys.

Now the beach keeper wants the distance between any two adjacent buoys to be the same. He plans to shift some or all of the buoys without changing their respective order. To facilitate the task, he wants the total length of all shifts to be as small as possible.

Given coordinates of the buoys, you should find the minimum possible length of all shifts, as well as new coordinates of the buoys.

Input

The first line of input contains a single integer n ($2 \le n \le 400$), n — the number of buoys. The second line contains buoys' integer coordinates $x_1, x_2, ..., x_n$ ($-10000 \le x_i \le 10000$). No two given buoys will share the same place. The coordinates are given in strictly increasing order.

Output

To the first line print a real number t — the minimum possible total length of required shifts. Output this value with at least 4 digits after the decimal point.

To the second line print n numbers — new coordinates of the buoys. The new coordinates should be printed in strictly increasing order with at least 7 digits after the decimal point. If there are several optimal ways to shift the buoys, you may output any of them.

Examples

input	output
4 -2 2 6 9	1.0000 -2.0000000000 1.6666666667 5.3333333333 9.0000000000

Note

All buoys are located on the Ox axis. You may move buoys only along the Ox axis.

E. Hack it!

time limit per test: 1 second
memory limit per test: 256 megabytes

Little X has met the following problem recently.

Let's define $f(x)$ as the sum of digits in decimal representation of number x (for example, $f(1234) = 1 + 2 + 3 + 4$). You are to calculate $\sum_{i=l}^{r} f(i) \bmod a$.

Of course Little X has solved this problem quickly, has locked it, and then has tried to hack others. He has seen the following C++ code:

```
ans = solve(l, r) % a;
if (ans <= 0)
  ans += a;
```

This code will fail only on the test with $\sum_{i=l}^{r} f(i) \equiv 0 \pmod{a}$. You are given number a, help Little X to find a proper test for hack.

Input

The first line contains a single integer a ($1 \le a \le 10^{18}$).

Output

Print two integers: l, r ($1 \le l \le r < 10^{200}$) — the required test data. Leading zeros aren't allowed. It's guaranteed that the solution exists.

Examples

input	output
46	1 10

input	output
126444381000032	2333333 2333333333333

F. Packmen Strike Back

time limit per test: 3 seconds
memory limit per test: 256 megabytes

Game field is represented by a line of n square cells. In some cells there are packmen, in some cells there are asterisks and the rest of the cells are empty. Packmen eat asterisks.

Before the game starts you can choose a movement direction, left or right, for each packman. Once the game begins all the packmen simultaneously start moving according their directions. A packman can't change the given direction.

Once a packman enters a cell containing an asterisk, packman immediately eats the asterisk. Once the packman leaves the cell it becomes empty. Each packman moves at speed 1 cell per second. If a packman enters a border cell, the packman stops. Packmen do not interfere with the movement of other packmen; in one cell there can be any number of packmen moving in any directions.

Your task is to assign a direction to each packman so that they eat the maximal number of asterisks. If there are multiple ways to assign directions to eat the maximal number of asterisks, you should choose the way which minimizes the time to do that.

Input

The first line contains integer number n ($2 \le n \le 1\,000\,000$) — the number of cells in the game field.

The second line contains n characters. If the i-th character is '`.`', the i-th cell is empty. If the i-th character is '`*`', the i-th cell contains an asterisk. If the i-th character is '`P`', the i-th cell contains a packman.

The field contains at least one asterisk and at least one packman.

Output

Print two integer numbers — the maximal number of asterisks packmen can eat and the minimal time to do it.

Examples

input	output
6 *.P*P*	3 4

input	output
8 *...P..*	1 3

Note

In the first example the leftmost packman should move to the right, the rightmost packman should move to the left. All the asterisks will be eaten, the last asterisk will be eaten after 4 seconds.