Codeforces Beta Round #93 (Div. 2 Only)

A. Wasted Time

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

Mr. Scrooge, a very busy man, decided to count the time he wastes on all sorts of useless stuff to evaluate the lost profit. He has already counted the time he wastes sleeping and eating. And now Mr. Scrooge wants to count the time he has wasted signing papers.

Mr. Scrooge's signature can be represented as a polyline $A_1A_2... A_n$. Scrooge signs like that: first it places a pen at the point A_1, then draws a segment from point A_1 to point A_2, then he draws a segment from point A_2 to point A_3 and so on to point A_n, where he stops signing and takes the pen off the paper. At that the resulting line can intersect with itself and partially repeat itself but Scrooge pays no attention to it and never changes his signing style. As Scrooge makes the signature, he never takes the pen off the paper and his writing speed is constant — 50 millimeters per second.

Scrooge signed exactly k papers throughout his life and all those signatures look the same.

Find the total time Scrooge wasted signing the papers.

Input

The first line contains two integers n and k ($2 \le n \le 100$, $1 \le k \le 1000$). Each of the following n lines contains the coordinates of the polyline's endpoints. The i-th one contains coordinates of the point A_i — integers x_i and y_i, separated by a space.

All points A_i are different. The absolute value of all coordinates does not exceed 20. The coordinates are measured in millimeters.

Output

Print one real number — the total time Scrooges wastes on signing the papers in seconds. The absolute or relative error should not exceed 10^{-6}.

Examples

input
2 1 0 0 10 0
output
0.200000000

input
5 10 3 1 -5 6 -2 -1 3 2 10 0
output
6.032163204

input
6 10 5 0 4 0 6 0 3 0 7 0 2 0
output
3.000000000

B. Canvas Frames

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

Nicholas, a painter is going to paint several new canvases. Nicholas is sure that the canvases will turn out so great that each one will need framing and being hung on the wall. Frames are what Nicholas decided to begin with.

Nicholas has n sticks whose lengths equal $a_1, a_2, \ldots a_n$. Nicholas does not want to break the sticks or glue them together. To make a $h \times w$-sized frame, he needs two sticks whose lengths equal h and two sticks whose lengths equal w. Specifically, to make a square frame (when $h = w$), he needs four sticks of the same length.

Now Nicholas wants to make from the sticks that he has as many frames as possible; to be able to paint as many canvases as possible to fill the frames. Help him in this uneasy task. Note that it is not necessary to use all the sticks Nicholas has.

Input

The first line contains an integer n ($1 \le n \le 100$) — the number of sticks. The second line contains n space-separated integers. The i-th integer equals the length of the i-th stick a_i ($1 \le a_i \le 100$).

Output

Print the single number — the maximum number of frames Nicholas can make for his future canvases.

Examples

input
5 2 4 3 2 3
output
1

input
13 2 2 4 4 4 4 6 6 6 7 7 9 9
output
3

input
4 3 3 3 5
output
0

C. Hot Bath

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

Bob is about to take a hot bath.

There are two taps to fill the bath: a hot water tap and a cold water tap. The cold water's temperature is t_1, and the hot water's temperature is t_2. The cold water tap can transmit any integer number of water units per second from 0 to x_1, inclusive. Similarly, the hot water tap can transmit from 0 to x_2 water units per second.

If y_1 water units per second flow through the first tap and y_2 water units per second flow through the second tap, then the resulting bath water temperature will be:

$$t = \frac{t_1 y_1 + t_2 y_2}{y_1 + y_2}$$

Bob wants to open both taps so that the bath water temperature was not less than t_0. However, the temperature should be as close as possible to this value. If there are several optimal variants, Bob chooses the one that lets fill the bath in the quickest way possible.

Determine how much each tap should be opened so that Bob was pleased with the result in the end.

Input

You are given five integers t_1, t_2, x_1, x_2 and t_0 ($1 \le t_1 \le t_0 \le t_2 \le 10^6$, $1 \le x_1, x_2 \le 10^6$).

Output

Print two space-separated integers y_1 and y_2 ($0 \le y_1 \le x_1$, $0 \le y_2 \le x_2$).

Examples

input
10 70 100 100 25
output
99 33

input
300 500 1000 1000 300
output
1000 0

input
143 456 110 117 273
output
76 54

Note

In the second sample the hot water tap shouldn't be opened, but the cold water tap should be opened at full capacity in order to fill the bath in the quickest way possible.

D. Password

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

Asterix, Obelix and their temporary buddies Suffix and Prefix has finally found the Harmony temple. However, its doors were firmly locked and even Obelix had no luck opening them.

A little later they found a string s, carved on a rock below the temple's gates. Asterix supposed that that's the password that opens the temple and read the string aloud. However, nothing happened. Then Asterix supposed that a password is some substring t of the string s.

Prefix supposed that the substring t is the beginning of the string s; Suffix supposed that the substring t should be the end of the string s; and Obelix supposed that t should be located somewhere inside the string s, that is, t is neither its beginning, nor its end.

Asterix chose the substring t so as to please all his companions. Besides, from all acceptable variants Asterix chose the longest one (as Asterix loves long strings). When Asterix read the substring t aloud, the temple doors opened.

You know the string s. Find the substring t or determine that such substring does not exist and all that's been written above is just a nice legend.

Input

You are given the string s whose length can vary from 1 to 10^6 (inclusive), consisting of small Latin letters.

Output

Print the string t. If a suitable t string does not exist, then print "`Just a legend`" without the quotes.

Examples

input
fixprefixsuffix
output
fix

input
abcdabc
output
Just a legend

E. E-reader Display

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

After years of hard work scientists invented an absolutely new e-reader display. The new display has a larger resolution, consumes less energy and its production is cheaper. And besides, one can bend it. The only inconvenience is highly unusual management. For that very reason the developers decided to leave the e-readers' software to programmers.

The display is represented by $n \times n$ square of pixels, each of which can be either black or white. The display rows are numbered with integers from 1 to n upside down, the columns are numbered with integers from 1 to n from the left to the right. The display can perform commands like "x, y". When a traditional display fulfills such command, it simply inverts a color of (x, y), where x is the row number and y is the column number. But in our new display every pixel that belongs to at least one of the segments (x, x) - (x, y) and (y, y) - (x, y) (both ends of both segments are included) inverts a color.

For example, if initially a display 5×5 in size is absolutely white, then the sequence of commands $(1, 4)$, $(3, 5)$, $(5, 1)$, $(3, 3)$ leads to the following changes:

You are an e-reader software programmer and you should calculate minimal number of commands needed to display the picture. You can regard all display pixels as initially white.

Input

The first line contains number n ($1 \le n \le 2000$).

Next n lines contain n characters each: the description of the picture that needs to be shown. "0" represents the white color and "1" represents the black color.

Output

Print one integer z — the least number of commands needed to display the picture.

Examples

input
5 01110 10010 10001 10011 11110
output
4