A. Repeating Cipher

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

Polycarp loves ciphers. He has invented his own cipher called *repeating*.

Repeating cipher is used for strings. To encrypt the string $s = s_1 s_2 \dots s_m$ $(1 \le m \le 10)$, Polycarp uses the following algorithm:

- he writes down s_1 ones,
- he writes down s_2 twice,
- he writes down s_3 three times,
- ...
- he writes down s_m m times.

For example, if s="bab" the process is: "b" $\rightarrow$ "baa" $\rightarrow$ "baabbb". So the encrypted s="bab" is "baabbb".

Given string t — the result of encryption of some string s. Your task is to decrypt it, i. e. find the string s.

Input

The first line contains integer n $(1 \le n \le 55)$ — the length of the encrypted string. The second line of the input contains t — the result of encryption of some string s. It contains only lowercase Latin letters. The length of t is exactly n.

It is guaranteed that the answer to the test exists.

Output

Print such string s that after encryption it equals t.

Examples

input	Copy
6 baabbb	
output	Copy
bab	

input	Copy
10 ooopppssss	
output	Copy
oops	

input	Copy
1 z	
output	Copy
z	

B. Tanya and Stairways

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

Little girl Tanya climbs the stairs inside a multi-storey building. Every time Tanya climbs a stairway, she starts counting steps from 1 to the number of steps in this stairway. She speaks every number aloud. For example, if she climbs two stairways, the first of which contains 3 steps, and the second contains 4 steps, she will pronounce the numbers $1, 2, 3, 1, 2, 3, 4$.

You are given all the numbers pronounced by Tanya. How many stairways did she climb? Also, output the number of steps in each stairway.

The given sequence will be a valid sequence that Tanya could have pronounced when climbing one or more stairways.

Input

The first line contains n ($1 \le n \le 1000$) — the total number of numbers pronounced by Tanya.

The second line contains integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 1000$) — all the numbers Tanya pronounced while climbing the stairs, in order from the first to the last pronounced number. Passing a stairway with x steps, she will pronounce the numbers $1, 2, \ldots, x$ in that order.

The given sequence will be a valid sequence that Tanya could have pronounced when climbing one or more stairways.

Output

In the first line, output t — the number of stairways that Tanya climbed. In the second line, output t numbers — the number of steps in each stairway she climbed. Write the numbers in the correct order of passage of the stairways.

Examples

input Copy

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

output Copy

```
2
3 4
```

input Copy

```
4
1 1 1 1
```

output Copy

```
4
1 1 1 1
```

input Copy

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

output Copy

```
1
5
```

input Copy

```
5
1 2 1 2 1
```

output Copy

```
3
2 2 1
```

C. Array Stabilization

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

You are given an array a consisting of n integer numbers.

Let *instability* of the array be the following value: $\max\limits_{i=1}^{n} a_i - \min\limits_{i=1}^{n} a_i$.

You have to remove **exactly one** element from this array to minimize *instability* of the resulting $(n-1)$-elements array. Your task is to calculate the minimum possible *instability*.

Input

The first line of the input contains one integer n ($2 \le n \le 10^5$) — the number of elements in the array a.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$) — elements of the array a.

Output

Print one integer — the minimum possible *instability* of the array if you have to remove **exactly one** element from the array a.

Examples

input Copy

```
4
1 3 3 7
```

output Copy

```
2
```

input Copy

```
2
1 100000
```

output Copy

```
0
```

Note

In the first example you can remove 7 then *instability* of the remaining array will be $3 - 1 = 2$.

In the second example you can remove either 1 or 100000 then *instability* of the remaining array will be $100000 - 100000 = 0$ and $1 - 1 = 0$ correspondingly.

D. Delete from the Left

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

You are given two strings s and t. In a single move, you can choose any of two strings and delete the first (that is, the leftmost) character. After a move, the length of the string decreases by 1. You can't choose a string if it is empty.

For example:

- by applying a move to the string "where", the result is the string "here",
- by applying a move to the string "a", the result is an empty string "".

You are required to make two given strings equal using the fewest number of moves. It is possible that, in the end, both strings will be equal to the empty string, and so, are equal to each other. In this case, the answer is obviously the sum of the lengths of the initial strings.

Write a program that finds the minimum number of moves to make two given strings s and t equal.

Input

The first line of the input contains s. In the second line of the input contains t. Both strings consist only of lowercase Latin letters. The number of letters in each string is between 1 and $2 \cdot 10^5$, inclusive.

Output

Output the fewest number of moves required. It is possible that, in the end, both strings will be equal to the empty string, and so, are equal to each other. In this case, the answer is obviously the sum of the lengths of the given strings.

Examples

input	Copy
test west	
output	Copy
2	

input	Copy
codeforces yes	
output	Copy
9	

input	Copy
test yes	
output	Copy
7	

input	Copy
b ab	
output	Copy
1	

Note

In the first example, you should apply the move once to the first string and apply the move once to the second string. As a result, both strings will be equal to "est".

In the second example, the move should be applied to the string "codeforces" 8 times. As a result, the string becomes "~~codeforces~~" $\rightarrow$ "es". The move should be applied to the string "yes" once. The result is the same string "~~y~~es" $\rightarrow$ "es".

In the third example, you can make the strings equal only by completely deleting them. That is, in the end, both strings will be empty.

In the fourth example, the first character of the second string should be deleted.

E. Integer Sequence Dividing

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

You are given an integer sequence $1, 2, \ldots, n$. You have to divide it into two sets A and B in such a way that each element belongs to **exactly one** set and $|sum(A) - sum(B)|$ is minimum possible.

The value $|x|$ is the absolute value of x and $sum(S)$ is the sum of elements of the set S.

Input

The first line of the input contains one integer n ($1 \le n \le 2 \cdot 10^9$).

Output

Print one integer — the minimum possible value of $|sum(A) - sum(B)|$ if you divide the initial sequence $1, 2, \ldots, n$ into two sets A and B.

Examples

input	Copy
3	
output	Copy
0	

input	Copy
5	
output	Copy
1	

input	Copy
6	
output	Copy
1	

Note

Some (not all) possible answers to examples:

In the first example you can divide the initial sequence into sets $A = \{1, 2\}$ and $B = \{3\}$ so the answer is 0.

In the second example you can divide the initial sequence into sets $A = \{1, 3, 4\}$ and $B = \{2, 5\}$ so the answer is 1.

In the third example you can divide the initial sequence into sets $A = \{1, 4, 5\}$ and $B = \{2, 3, 6\}$ so the answer is 1.

F. Asterisk Visibility (Easy Edition)

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

You are given a rectangular field of characters is given. It consists of n lines with m characters in each line. Each character is either a period '`.`' or an asterisk '`*`'.

Let's call *the beauty* of a cell (i, j) the number 0 if there is a period in this cell and the number of asterisks that are visible from it if there is an asterisk in (i, j). From one asterisk you can see the other if and only if they are in the same line (row or column) and there are no periods between them.

Find the greatest beauty among the beauties of all cells and the number of cells that have such beauty.

Input

The first line contains two integers n, m $(1 \le n, m \le 400)$ — the number of lines and columns respectively.

The following n lines contain m characters each. Each character is either a period '`.`', either an asterisk '`*`'.

Output

Print two numbers — the maximum beauty among the beauties of all cells and the number of cells with such beauty.

Examples

input Copy

```
4 4
..**
****
..*.
*.**
```

output Copy

```
7 1
```

input Copy

```
1 3
*.*
```

output Copy

```
1 2
```

input Copy

```
3 1
*

.
*
```

output Copy

```
1 2
```

input Copy

```
5 5
*****
*.***
*****
*****
*****
```

output Copy

```
9 16
```

H. Songs Compression

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

Ivan has n songs on his phone. The size of the i-th song is a_i bytes. Ivan also has a flash drive which can hold at most m bytes in total. Initially, his flash drive is empty.

Ivan wants to copy all n songs to the flash drive. He can compress the songs. If he compresses the i-th song, the size of the i-th song reduces from a_i to b_i bytes ($b_i < a_i$).

Ivan can compress any subset of the songs (possibly empty) and copy all the songs to his flash drive if the sum of their sizes is at most m. He can compress *any* subset of the songs (not necessarily contiguous).

Ivan wants to find the minimum number of songs he needs to compress in such a way that all his songs fit on the drive (i.e. the sum of their sizes is less than or equal to m).

If it is impossible to copy all the songs (even if Ivan compresses all the songs), print "`-1`". Otherwise print the minimum number of songs Ivan needs to compress.

Input

The first line of the input contains two integers n and m ($1 \le n \le 10^5, 1 \le m \le 10^9$) — the number of the songs on Ivan's phone and the capacity of Ivan's flash drive.

The next n lines contain two integers each: the i-th line contains two integers a_i and b_i ($1 \le a_i, b_i \le 10^9$, $a_i > b_i$) — the initial size of the i-th song and the size of the i-th song after compression.

Output

If it is impossible to compress a subset of the songs in such a way that all songs fit on the flash drive, print "`-1`". Otherwise print the minimum number of the songs to compress.

Examples

input Copy

```
4 21
10 8
7 4
3 1
5 4
```

output Copy

```
2
```

input Copy

```
4 16
10 8
7 4
3 1
5 4
```

output Copy

```
-1
```

Note

In the first example Ivan can compress the first and the third songs so after these moves the sum of sizes will be equal to $8 + 7 + 1 + 5 = 21 \le 21$. Also Ivan can compress the first and the second songs, then the sum of sizes will be equal $8 + 4 + 3 + 5 = 20 \le 21$. Note that compressing any single song is not sufficient to copy all the songs on the flash drive (for example, after compressing the second song the sum of sizes will be equal to $10 + 4 + 3 + 5 = 22 > 21$).

In the second example even if Ivan compresses all the songs the sum of sizes will be equal $8 + 4 + 1 + 4 = 17 > 16$.