Codeforces Round #200 (Div. 2)

A. Magnets

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

Mad scientist Mike entertains himself by arranging rows of dominoes. He doesn't need dominoes, though: he uses rectangular magnets instead. Each magnet has two poles, positive (a "plus") and negative (a "minus"). If two magnets are put together at a close distance, then the like poles will repel each other and the opposite poles will attract each other.

Mike starts by laying one magnet horizontally on the table. During each following step Mike adds one more magnet horizontally to the right end of the row. Depending on how Mike puts the magnet on the table, it is either attracted to the previous one (forming a group of multiple magnets linked together) or repelled by it (then Mike lays this magnet at some distance to the right from the previous one). We assume that a sole magnet not linked to others forms a group of its own.

Mike arranged multiple magnets in a row. Determine the number of groups that the magnets formed.

Input

The first line of the input contains an integer n ($1 \le n \le 100000$) — the number of magnets. Then n lines follow. The i-th line ($1 \le i \le n$) contains either characters "01", if Mike put the i-th magnet in the "plus-minus" position, or characters "10", if Mike put the magnet in the "minus-plus" position.

Output

On the single line of the output print the number of groups of magnets.

Sample test(s)

input
6 10 10 10 01 10 10
output
3

input
4 01 01 10 10
output
2

Note

The first testcase corresponds to the figure. The testcase has three groups consisting of three, one and two magnets.

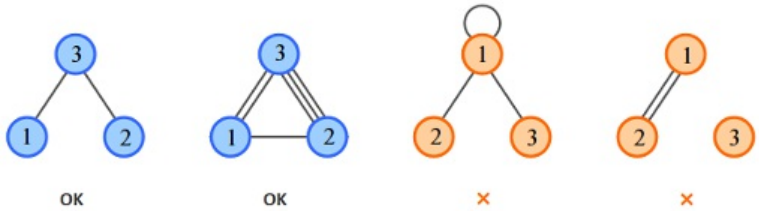
The second testcase has two groups, each consisting of two magnets.

B. Simple Molecules

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

Mad scientist Mike is busy carrying out experiments in chemistry. Today he will attempt to join three atoms into one molecule.

A molecule consists of atoms, with some pairs of atoms connected by atomic bonds. Each atom has a valence number — the number of bonds the atom must form with other atoms. An atom can form **one or multiple** bonds with any other atom, but it cannot form a bond with itself. The number of bonds of an atom in the molecule must be equal to its valence number.



Mike knows valence numbers of the three atoms. Find a molecule that can be built from these atoms according to the stated rules, or determine that it is impossible.

Input

The single line of the input contains three space-separated integers a, b and c ($1 \le a, b, c \le 10^6$) — the valence numbers of the given atoms.

Output

If such a molecule can be built, print three space-separated integers — the number of bonds between the 1-st and the 2-nd, the 2-nd and the 3-rd, the 3-rd and the 1-st atoms, correspondingly. If there are multiple solutions, output any of them. If there is no solution, print "`Impossible`" (without the quotes).

Sample test(s)

input
1 1 2
output
0 1 1

input
3 4 5
output
1 3 2

input
4 1 1
output
Impossible

Note

The first sample corresponds to the first figure. There are no bonds between atoms 1 and 2 in this case.

The second sample corresponds to the second figure. There is one or more bonds between each pair of atoms.

The third sample corresponds to the third figure. There is no solution, because an atom cannot form bonds with itself.

The configuration in the fourth figure is impossible as each atom must have at least one atomic bond.

C. Rational Resistance

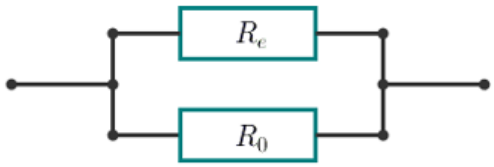
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Mad scientist Mike is building a time machine in his spare time. To finish the work, he needs a resistor with a certain resistance value.

However, all Mike has is lots of identical resistors with unit resistance $R_0 = 1$. Elements with other resistance can be constructed from these resistors. In this problem, we will consider the following as elements:

1. one resistor;
2. an element and **one** resistor plugged in sequence;
3. an element and **one** resistor plugged in parallel.

$$R = R_e + R_0$$



$$R = \frac{1}{\frac{1}{R_e} + \frac{1}{R_0}}$$

With the consecutive connection the resistance of the new element equals $R = R_e + R_0$. With the parallel connection the resistance of the new element equals $R = \frac{1}{\frac{1}{R_e} + \frac{1}{R_0}}$. In this case R_e equals the resistance of the element being connected.

Mike needs to assemble an element with a resistance equal to the fraction $\frac{a}{b}$. Determine the smallest possible number of resistors he needs to make such an element.

Input

The single input line contains two space-separated integers a and b ($1 \le a, b \le 10^{18}$). It is guaranteed that the fraction $\frac{a}{b}$ is irreducible. It is guaranteed that a solution always exists.

Output

Print a single number — the answer to the problem.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is recommended to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
1 1
output
1

input
3 2
output
3

input
199 200
output
200

Note

In the first sample, one resistor is enough.

In the second sample one can connect the resistors in parallel, take the resulting element and connect it to a third resistor consecutively. Then, we get an element with resistance $\frac{1}{\frac{1}{1}+\frac{1}{1}} + 1 = \frac{3}{2}$. We cannot make this element using two resistors.

D. Alternating Current

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

Mad scientist Mike has just finished constructing a new device to search for extraterrestrial intelligence! He was in such a hurry to launch it for the first time that he plugged in the power wires without giving it a proper glance and started experimenting right away. After a while Mike observed that the wires ended up entangled and now have to be untangled again.

The device is powered by two wires "plus" and "minus". The wires run along the floor from the wall (on the left) to the device (on the right). Both the wall and the device have two contacts in them on the same level, into which the wires are plugged in some order. The wires are considered entangled if there are one or more places where one wire runs above the other one. For example, the picture below has four such places (top view):

Mike knows the sequence in which the wires run above each other. Mike also noticed that on the left side, the "plus" wire is always plugged into the top contact (as seen on the picture). He would like to untangle the wires without unplugging them and **without moving** the device. Determine if it is possible to do that. A wire can be freely moved and stretched on the floor, but cannot be cut.

To understand the problem better please read the notes to the test samples.

Input

The single line of the input contains a sequence of characters "+" and "−" of length n ($1 \le n \le 100000$). The i-th ($1 \le i \le n$) position of the sequence contains the character "+", if on the i-th step from the wall the "plus" wire runs above the "minus" wire, and the character "−" otherwise.

Output

Print either "Yes" (without the quotes) if the wires can be untangled or "No" (without the quotes) if the wires cannot be untangled.

Sample test(s)

input
-++-
output
Yes

input
+-
output
No

input
++
output
Yes

input
-
output
No

Note

The first testcase corresponds to the picture in the statement. To untangle the wires, one can first move the "plus" wire lower, thus eliminating the two crosses in the middle, and then draw it under the "minus" wire, eliminating also the remaining two crosses.

In the second testcase the "plus" wire makes one full revolution around the "minus" wire. Thus the wires cannot be untangled:

In the third testcase the "plus" wire simply runs above the "minus" wire twice in sequence. The wires can be untangled by lifting "plus" and moving it higher:

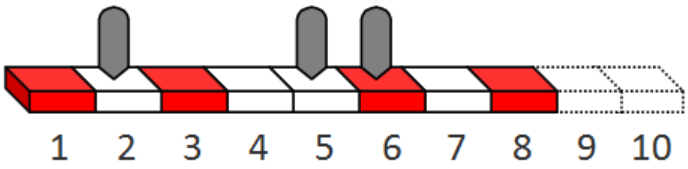
In the fourth testcase the "minus" wire runs above the "plus" wire once. The wires cannot be untangled without moving the device itself:

E. Read Time

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

Mad scientist Mike does not use slow hard disks. His modification of a hard drive has not one, but n different heads that can read data in parallel.

When viewed from the side, Mike's hard drive is an endless array of tracks. The tracks of the array are numbered from left to right with integers, starting with 1. In the initial state the i-th reading head is above the track number h_i. For each of the reading heads, the hard drive's firmware can move the head exactly one track to the right or to the left, or leave it on the current track. During the operation each head's movement does not affect the movement of the other heads: the heads can change their relative order; there can be multiple reading heads above any of the tracks. A track is considered *read* if at least one head has visited this track. In particular, all of the tracks numbered $h_1, h_2, ..., h_n$ have been read at the beginning of the operation.



Mike needs to read the data on m distinct tracks with numbers $p_1, p_2, ..., p_m$. Determine the minimum time the hard drive firmware needs to move the heads and read all the given tracks. Note that an arbitrary number of other tracks can also be read.

Input

The first line of the input contains two space-separated integers n, m $(1 \le n, m \le 10^5)$ — the number of disk heads and the number of tracks to read, accordingly. The second line contains n distinct integers h_i in ascending order $(1 \le h_i \le 10^{10}, h_i < h_{i+1})$ — the initial positions of the heads. The third line contains m distinct integers p_i in ascending order $(1 \le p_i \le 10^{10}, p_i < p_{i+1})$ - the numbers of tracks to read.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is recommended to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

Print a single number — the minimum time required, in seconds, to read all the needed tracks.

Sample test(s)

input
3 4 2 5 6 1 3 6 8
output
2

input
3 3 1 2 3 1 2 3
output
0

input
1 2 165 142 200
output
81

Note

The first test coincides with the figure. In this case the given tracks can be read in 2 seconds in the following way:

1. during the first second move the 1-st head to the left and let it stay there;
2. move the second head to the left twice;
3. move the third head to the right twice (note that the 6-th track has already been read at the beginning).

One cannot read the tracks in 1 second as the 3-rd head is at distance 2 from the 8-th track.