AIM Tech Round (Div. 2)

A. Save Luke

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

Luke Skywalker got locked up in a rubbish shredder between two presses. R2D2 is already working on his rescue, but Luke needs to stay alive as long as possible. For simplicity we will assume that everything happens on a straight line, the presses are initially at coordinates 0 and L, and they move towards each other with speed v_1 and v_2, respectively. Luke has width d and is able to choose any position between the presses. Luke dies as soon as the distance between the presses is less than his width. Your task is to determine for how long Luke can stay alive.

Input

The first line of the input contains four integers d, L, v_1, v_2 ($1 \le d, L, v_1, v_2 \le 10\,000$, $d < L$) — Luke's width, the initial position of the second press and the speed of the first and second presses, respectively.

Output

Print a single real value — the maximum period of time Luke can stay alive for. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct, if .

Examples

input
```
2 6 2 2
```
output
```
1.00000000000000000000
```

input
```
1 9 1 2
```
output
```
2.66666666666666650000
```

Note

In the first sample Luke should stay exactly in the middle of the segment, that is at coordinates $[2;4]$, as the presses move with the same speed.

In the second sample he needs to occupy the position . In this case both presses move to his edges at the same time.

B. Making a String

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

You are given an alphabet consisting of n letters, your task is to make a string of the maximum possible length so that the following conditions are satisfied:

- the i-th letter occurs in the string no more than a_i times;
- the number of occurrences of each letter in the string must be **distinct** for all the letters that occurred in the string at least once.

Input

The first line of the input contains a single integer n ($2 \leq n \leq 26$) — the number of letters in the alphabet.

The next line contains n integers a_i ($1 \leq a_i \leq 10^9$) — i-th of these integers gives the limitation on the number of occurrences of the i-th character in the string.

Output

Print a single integer — the maximum length of the string that meets all the requirements.

Examples

input

```
3
2 5 5
```

output

```
11
```

input

```
3
1 1 2
```

output

```
3
```

Note

For convenience let's consider an alphabet consisting of three letters: "`a`", "`b`", "`c`". In the first sample, some of the optimal strings are: "`cccaabbccbb`", "`aabcbcbcbcb`". In the second sample some of the optimal strings are: "`acc`", "`cbc`".

C. Graph and String

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

One day student Vasya was sitting on a lecture and mentioned a string $s_1 s_2 \dots s_n$, consisting of letters "a", "b" and "c" that was written on his desk. As the lecture was boring, Vasya decided to complete the picture by composing a graph G with the following properties:

- G has exactly n vertices, numbered from 1 to n.
- For all pairs of vertices i and j, where $i \neq j$, there is an edge connecting them **if and only if** characters s_i and s_j are either equal or neighbouring in the alphabet. That is, letters in pairs "a"-"b" and "b"-"c" are neighbouring, while letters "a"-"c" are not.

Vasya painted the resulting graph near the string and then erased the string. Next day Vasya's friend Petya came to a lecture and found some graph at his desk. He had heard of Vasya's adventure and now he wants to find out whether it could be the original graph G, painted by Vasya. In order to verify this, Petya needs to know whether there exists a string s, such that if Vasya used this s he would produce the given graph G.

Input

The first line of the input contains two integers n and m — the number of vertices and edges in the graph found by Petya, respectively.

Each of the next m lines contains two integers u_i and v_i $(1 \le u_i, v_i \le n, u_i \neq v_i)$ — the edges of the graph G. It is guaranteed, that there are no multiple edges, that is any pair of vertexes appear in this list no more than once.

Output

In the first line print "Yes" (without the quotes), if the string s Petya is interested in really exists and "No" (without the quotes) otherwise.

If the string s exists, then print it on the second line of the output. The length of s must be exactly n, it must consist of only letters "a", "b" and "c" only, and the graph built using this string must coincide with G. If there are multiple possible answers, you may print any of them.

Examples

input
```
2 1
1 2
```
output
```
Yes
aa
```

input
```
4 3
1 2
1 3
1 4
```
output
```
No
```

Note

In the first sample you are given a graph made of two vertices with an edge between them. So, these vertices can correspond to both the same and adjacent letters. Any of the following strings "aa", "ab", "ba", "bb", "bc", "cb", "cc" meets the graph's conditions.

In the second sample the first vertex is connected to all three other vertices, but these three vertices are not connected with each other. That means that they must correspond to distinct letters that are not adjacent, but that is impossible as there are only two such letters: a and c.

D. Array GCD

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given array a_i of length n. You may consecutively apply two operations to this array:

- remove some subsegment (continuous subsequence) of length $m < n$ and pay for it $m \cdot a$ coins;
- change some elements of the array by at most 1, and pay b coins for each change.

Please note that each of operations may be applied at most once (and may be not applied at all) so you can remove only one segment and each number may be changed (increased or decreased) by at most 1. Also note, that you are not allowed to delete the whole array.

Your goal is to calculate the minimum number of coins that you need to spend in order to make the greatest common divisor of the elements of the resulting array be greater than 1.

Input

The first line of the input contains integers n, a and b ($1 \le n \le 1\,000\,000$, $0 \le a, b \le 10^9$) — the length of the array, the cost of removing a single element in the first operation and the cost of changing an element, respectively.

The second line contains n integers a_i ($2 \le a_i \le 10^9$) — elements of the array.

Output

Print a single number — the minimum cost of changes needed to obtain an array, such that the greatest common divisor of all its elements is greater than 1.

Examples

input

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

output

```
1
```

input

```
5 3 2
5 17 13 5 6
```

output

```
8
```

input

```
8 3 4
3 7 5 4 3 12 9 4
```

output

```
13
```

Note

In the first sample the optimal way is to remove number 3 and pay 1 coin for it.

In the second sample you need to remove a segment $[17, 13]$ and then decrease number 6. The cost of these changes is equal to $2 \cdot 3 + 2 = 8$ coins.

E. Electric Charges

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

Programmer Sasha is a student at MIPT (Moscow Institute of Physics and Technology) and he needs to make a laboratory work to pass his finals.

A laboratory unit is a plane with standard coordinate axes marked on it. Physicists from Moscow Institute of Physics and Technology charged the axes by large electric charges: axis X is positive and axis Y is negative.

Experienced laboratory worker marked n points with integer coordinates (x_i, y_i) on the plane and stopped the time. Sasha should use "atomic tweezers" to place elementary particles in these points. He has an unlimited number of electrons (negatively charged elementary particles) and protons (positively charged elementary particles). He can put either an electron or a proton at each marked point. As soon as all marked points are filled with particles, laboratory worker will turn on the time again and the particles will come in motion and after some time they will stabilize in equilibrium. The objective of the laboratory work is to arrange the particles in such a way, that the diameter of the resulting state (the maximum distance between the pairs of points of the set) is as small as possible.

Since Sasha is a programmer, he naively thinks that all the particles will simply "fall" into their projections on the corresponding axes: electrons will fall on axis X, while protons will fall on axis Y. As we are programmers too, we will consider the same model as Sasha. That is, a **particle gets from point (x, y) to point $(x, 0)$ if it is an electron and to point $(0, y)$ if it is a proton.**

As the laboratory has high background radiation and Sasha takes care of his laptop, he did not take it with him, and now he can't write a program that computes the minimum possible diameter of the resulting set. Therefore, you will have to do it for him.

Print a **square** of the minimum possible diameter of the set.

Input

The first line of the input contains a single integer n $(1 \le n \le 100\,000)$ — the number of points marked on the plane.

Each of the next n lines contains two integers x_i and y_i $(-10^8 \le x_i, y_i \le 10^8)$ — the coordinates of the i-th point. It is guaranteed that no two points coincide.

Output

Print a single integer — the square of the minimum possible diameter of the set.

Examples

input

```
3
1 10
1 20
1 30
```

output

```
0
```

input

```
2
1 10
10 1
```

output

```
2
```

Note

In the first sample Sasha puts electrons at all points, all particles eventually fall at a single point $(1, 0)$.

In the second sample Sasha puts an electron at point $(1, 10)$, and a proton at point $(10, 1)$. The result is a set of two points $(1, 0)$ and $(0, 1)$, which has a diameter of .