Codeforces Round #539 (Div. 2)

A. Sasha and His Trip

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

Sasha is a very happy guy, that's why he is always on the move. There are n cities in the country where Sasha lives. They are all located on one straight line, and for convenience, they are numbered from 1 to n in increasing order. The distance between any two adjacent cities is equal to 1 kilometer. Since all roads in the country are directed, it's possible to reach the city y from the city x only if $x < y$.

Once Sasha decided to go on a trip around the country and to visit all n cities. He will move with the help of his car, *Cheetah-2677*. The tank capacity of this model is v liters, and it spends exactly 1 liter of fuel for 1 kilometer of the way. At the beginning of the journey, the tank is empty. Sasha is located in the city with the number 1 and wants to get to the city with the number n. There is a gas station in each city. In the i-th city, the price of 1 liter of fuel is i dollars. It is obvious that at any moment of time, the tank can contain at most v liters of fuel.

Sasha doesn't like to waste money, that's why he wants to know what is the minimum amount of money is needed to finish the trip if he can buy fuel in any city he wants. Help him to figure it out!

Input

The first line contains two integers n and v ($2 \le n \le 100$, $1 \le v \le 100$) — the number of cities in the country and the capacity of the tank.

Output

Print one integer — the minimum amount of money that is needed to finish the trip.

Examples

input

```
4 2
```

output

```
4
```

input

```
7 6
```

output

```
6
```

Note

In the first example, Sasha can buy 2 liters for 2 dollars (1 dollar per liter) in the first city, drive to the second city, spend 1 liter of fuel on it, then buy 1 liter for 2 dollars in the second city and then drive to the 4-th city. Therefore, the answer is $1+1+2=4$.

In the second example, the capacity of the tank allows to fill the tank completely in the first city, and drive to the last city without stops in other cities.

B. Sasha and Magnetic Machines

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

One day Sasha visited the farmer 2D and his famous magnetic farm. On this farm, the crop grows due to the influence of a special magnetic field. Maintaining of the magnetic field is provided by n machines, and the power of the i-th machine is a_i.

This year 2D decided to cultivate a new culture, but what exactly he didn't say. For the successful growth of the new culture, it is necessary to slightly change the powers of the machines. 2D can **at most once** choose an arbitrary integer x, then choose one machine and reduce the power of its machine by x times, and at the same time increase the power of one another machine by x times (powers of all the machines must stay **positive integers**). Note that he may not do that if he wants. More formally, 2D can choose two such indices i and j, and one integer x such that x is a divisor of a_i, and change powers as following: $a_i = \frac{a_i}{x}$, $a_j = a_j \cdot x$

Sasha is very curious, that's why he wants to calculate the **minimum** total power the farmer can reach. There are too many machines, and Sasha can't cope with computations, help him!

Input

The first line contains one integer n ($2 \le n \le 5 \cdot 10^4$) — the number of machines.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100$) — the powers of the machines.

Output

Print one integer — minimum total power.

Examples

input

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

output

```
14
```

input

```
4
4 2 4 4
```

output

```
14
```

input

```
5
2 4 2 3 7
```

output

```
18
```

Note

In the first example, the farmer can reduce the power of the 4-th machine by 2 times, and increase the power of the 1-st machine by 2 times, then the powers will be: $[2, 2, 3, 2, 5]$.

In the second example, the farmer can reduce the power of the 3-rd machine by 2 times, and increase the power of the 2-nd machine by 2 times. At the same time, the farmer can leave is be as it is and the total power won't change.

In the third example, it is optimal to leave it be as it is.

C. Sasha and a Bit of Relax

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

Sasha likes programming. Once, during a very long contest, Sasha decided that he was a bit tired and needed to relax. So he did. But since Sasha isn't an ordinary guy, he prefers to relax unusually. During leisure time Sasha likes to upsolve unsolved problems because upsolving is very useful.

Therefore, Sasha decided to upsolve the following problem:

You have an array a with n integers. You need to count the number of *funny* pairs (l, r) $(l \le r)$. To check if a pair (l, r) is a *funny* pair, take $mid = \frac{l+r-1}{2}$, then if $r - l + 1$ is an **even** number and $a_l \oplus a_{l+1} \oplus \ldots \oplus a_{mid} = a_{mid+1} \oplus a_{mid+2} \oplus \ldots \oplus a_r$, then the pair is *funny*. In other words, $\oplus$ of elements of the left half of the subarray from l to r should be equal to $\oplus$ of elements of the right half. Note that $\oplus$ denotes the bitwise XOR operation.

It is time to continue solving the contest, so Sasha asked you to solve this task.

Input

The first line contains one integer n ($2 \le n \le 3 \cdot 10^5$) — the size of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < 2^{20}$) — array itself.

Output

Print one integer — the number of *funny* pairs. You should consider only pairs where $r - l + 1$ is even number.

Examples

input

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

output

```
1
```

input
```
6
3 2 2 3 7 6
```
output
```
3
```

input
```
3
42 4 2
```
output
```
0
```

Note

Be as cool as Sasha, upsolve problems!

In the first example, the only *funny* pair is $(2, 5)$, as $2 \oplus 3 = 4 \oplus 5 = 1$.

In the second example, funny pairs are $(2, 3)$, $(1, 4)$, and $(3, 6)$.

In the third example, there are no *funny* pairs.

D. Sasha and One More Name

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

Reading books is one of Sasha's passions. Once while he was reading one book, he became acquainted with an unusual character. The character told about himself like that: "Many are my names in many countries. Mithrandir among the Elves, Tharkûn to the Dwarves, Olórin I was in my youth in the West that is forgotten, in the South Incánus, in the North Gandalf; to the East I go not."

And at that moment Sasha thought, how would that character be called in the East? In the East all names are palindromes. A string is a palindrome if it reads the same backward as forward. For example, such strings as "kazak", "oo" and "r" are palindromes, but strings "abb" and "ij" are not.

Sasha believed that the hero would be named after one of the gods of the East. As long as there couldn't be two equal names, so in the East people did the following: they wrote the original name as a string on a piece of paper, then cut the paper minimum number of times k, so they got $k + 1$ pieces of paper with substrings of the initial string, and then unite those pieces together to get a new string. Pieces **couldn't be turned over**, they could be shuffled.

In this way, it's possible to achive a string abcdefg from the string f|de|abc|g using 3 cuts (by swapping papers with substrings f and abc). The string cbadefg can't be received using the same cuts.

More formally, Sasha wants for the given **palindrome** s find such minimum k, that you can cut this string into $k + 1$ parts, and then unite them in such a way that the final string will be a palindrome and it won't be equal to the initial string s. It there is no answer, then print "Impossible" (without quotes).

Input

The first line contains one string s ($1 \le |s| \le 5\,000$) — the initial name, which consists only of lowercase Latin letters. It is guaranteed that s is a palindrome.

Output

Print one integer k — the minimum number of cuts needed to get a new name, or "Impossible" (without quotes).

Examples

input
```
nolon
```
output
```
2
```

input
```
otto
```
output
```
1
```

input
```
qqqq
```
output
```
Impossible
```

input

```
kinnikkinnik
```

output

```
1
```

Note

In the first example, you can cut the string in those positions: `no|l|on`, and then unite them as follows `on|l|no`. It can be shown that there is no solution with one cut.

In the second example, you can cut the string right in the middle, and swap peaces, so you get `toot`.

In the third example, you can't make a string, that won't be equal to the initial one.

In the fourth example, you can cut the suffix `nik` and add it to the beginning, so you get `nikkinnikkin`.

E. Sasha and a Patient Friend

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Fedya and Sasha are friends, that's why Sasha knows everything about Fedya.

Fedya keeps his patience in an infinitely large bowl. But, unlike the bowl, Fedya's patience isn't infinite, that is why let v be the number of liters of Fedya's patience, and, as soon as v becomes **equal** to 0, the bowl will burst immediately. There is one tap in the bowl which pumps s liters of patience per second. Notice that s can be negative, in that case, the tap pumps out the patience. Sasha can do different things, so he is able to change the tap's speed. All actions that Sasha does can be represented as q queries. There are three types of queries:

1. "`1 t s`" — add a new event, means that starting from the t-th second the tap's speed will be equal to s.
2. "`2 t`" — delete the event which happens at the t-th second. It is guaranteed that such event exists.
3. "`3 l r v`" — Sasha wonders: if you take all the events for which $l \le t \le r$ and simulate changes of Fedya's patience from the very beginning of the l-th second till the very beginning of the r-th second inclusive (the initial volume of patience, at the beginning of the l-th second, equals to v liters) then when will be the moment when the bowl will burst. If that does not happen, then the answer will be -1.

Since Sasha does not want to check what will happen when Fedya's patience ends, and he has already come up with the queries, he is asking you to help him and find the answer for each query of the 3-rd type.

It is guaranteed that at any moment of time, there won't be two events which happen at the same second.

Input

The first line contans one integer q $(1 \le q \le 10^5)$ — the number of queries.

Each of the next q lines have one of the following formats:

- `1 t s` $(1 \le t \le 10^9, -10^9 \le s \le 10^9)$, means that a new event is added, which means that starting from the t-th second the tap's speed will be equal to s.
- `2 t` $(1 \le t \le 10^9)$, means that the event which happens at the t-th second must be deleted. Guaranteed that such exists.
- `3 l r v` $(1 \le l \le r \le 10^9, 0 \le v \le 10^9)$, means that you should simulate the process from the very beginning of the l-th second till the very beginning of the r-th second inclusive, and to say when will the bowl burst.

It is guaranteed that t, s, l, r, v in all the queries are integers.

Also, it is guaranteed that there is at least one query of the 3-rd type, and there won't be a query of the 1-st type with such t, that there already exists an event which happens at that second t.

Output

For each query of the 3-rd type, print in a new line the moment when the bowl will burst or print -1 if it won't happen.

Your answer will be considered correct if it's absolute or relative error does not exceed 10^{-6}.

Formally, let your answer be a, and the jury's answer be b. Your answer is accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$.

Examples

input

```
6
1 2 1
1 4 -3
3 1 6 1
3 1 6 3
3 1 6 4
3 1 6 5
```

output

```
5
5.666667
6
-1
```

input

```
10
1 2 2
1 4 4
1 7 -10
3 2 4 1
3 5 6 0
3 1 15 1
2 4
3 1 15 1
1 8 1
3 1 15 1
```

output

```
-1
5
8.7
8.1
-1
```

input

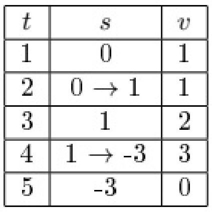
```
5
1 1000 9999999
1 2000 -9999
3 1000 2000 0
2 1000
3 1000 2002 1
```

output

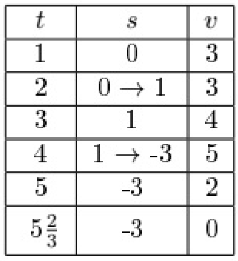
```
1000
2000.0001
```

Note

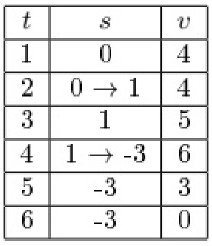
In the first example all the queries of the 3-rd type cover all the events, it's simulation is following:



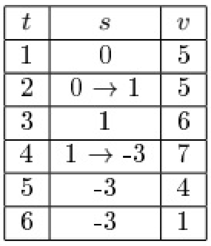
t	s	v
1	0	1
2	$0 \rightarrow 1$	1
3	1	2
4	$1 \rightarrow -3$	3
5	-3	0



t	s	v
1	0	3
2	$0 \rightarrow 1$	3
3	1	4
4	$1 \rightarrow -3$	5
5	-3	2
$5\frac{2}{3}$	-3	0



t	s	v
1	0	4
2	$0 \rightarrow 1$	4
3	1	5
4	$1 \rightarrow -3$	6
5	-3	3
6	-3	0



t	s	v
1	0	5
2	$0 \rightarrow 1$	5
3	1	6
4	$1 \rightarrow -3$	7
5	-3	4
6	-3	1

F. Sasha and Interesting Fact from Graph Theory

time limit per test: 2.5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Once, during a lesson, Sasha got bored and decided to talk with his friends. Suddenly, he saw Kefa. Since we can talk endlessly about Kefa, we won't even start doing that. The conversation turned to graphs. Kefa promised Sasha to tell him about one interesting fact from graph theory if Sasha helps Kefa to count the number of *beautiful trees*.

In this task, a *tree* is a weighted connected graph, consisting of n vertices and $n-1$ edges, and weights of edges are integers from 1 to m. Kefa determines the beauty of a tree as follows: he finds in the tree his two favorite vertices — vertices with numbers a and b, and counts the distance between them. The distance between two vertices x and y is the sum of weights of edges on the simple path from x to y. If the distance between two vertices a and b is **equal** to m, then the tree is *beautiful*.

Sasha likes graph theory, and even more, Sasha likes interesting facts, that's why he agreed to help Kefa. Luckily, Sasha is familiar with you `the best programmer in Byteland`. Help Sasha to count the number of *beautiful* trees for Kefa. Two trees are considered to be distinct if there is an edge that occurs in one of them and doesn't occur in the other one. Edge's **weight matters**.

Kefa warned Sasha, that there can be too many beautiful trees, so it will be enough to count the number modulo 10^9+7.

Input

The first line contains four integers n, m, a, b ($2 \le n \le 10^6$, $1 \le m \le 10^6$, $1 \le a, b \le n$, $a \ne b$) — the number of vertices in the tree, the maximum weight of an edge and two Kefa's favorite vertices.

Output

Print one integer — the number of *beautiful trees* modulo 10^9+7.

Examples

input

```
3 2 1 3
```

output

```
5
```

input

```
3 1 1 2
```

output

```
2
```

input

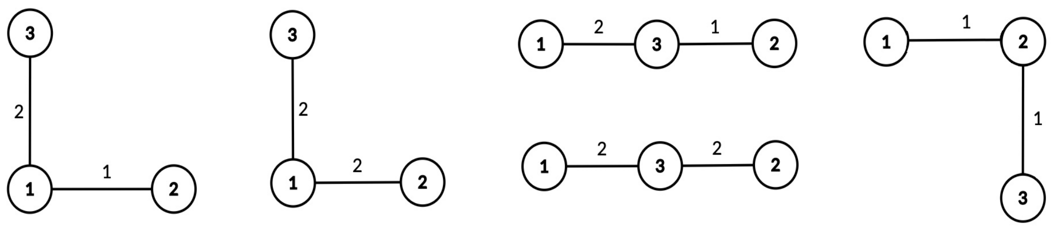
```
5 15 1 5
```

output

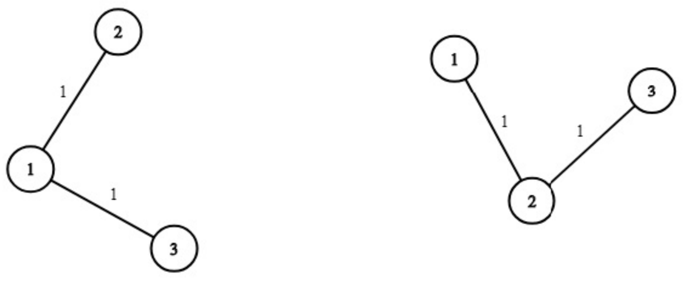
```
345444
```

Note

There are 5 *beautiful trees* in the first example:



In the second example the following trees are *beautiful*:


2
1
1
1
3
1
1
1
3
2