A. ACM ICPC

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

In a small but very proud high school it was decided to win ACM ICPC. This goal requires to compose as many teams of three as possible, but since there were only 6 students who wished to participate, the decision was to build exactly two teams.

After practice competition, participant number i got a *score* of a_i. *Team score* is defined as sum of scores of its participants. High school management is interested if it's possible to build two teams with equal scores. Your task is to answer that question.

Input

The single line contains six integers $a_1, \ldots, a_6$ $(0 \le a_i \le 1000)$ — scores of the participants

Output

Print "YES" (quotes for clarity), if it is possible to build teams with equal score, and "NO" otherwise.

You can print each character either upper- or lowercase ("YeS" and "yes" are valid when the answer is "YES").

Examples

input
```
1 3 2 1 2 1
```
output
```
YES
```

input
```
1 1 1 1 1 99
```
output
```
NO
```

Note

In the first sample, first team can be composed of 1st, 2nd and 6th participant, second — of 3rd, 4th and 5th: team scores are $1 + 3 + 1 = 2 + 1 + 2 = 5$.

In the second sample, score of participant number 6 is too high: his team score will be definitely greater.

B. Vlad and Cafes

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

Vlad likes to eat in cafes very much. During his life, he has visited cafes n times. Unfortunately, Vlad started to feel that his last visits are not any different from each other. To fix that Vlad had a small research.

First of all, Vlad assigned individual indices to all cafes. Then, he wrote down indices of cafes he visited in a row, in order of visiting them. Now, Vlad wants to find such a cafe that his last visit to that cafe was before his last visits to every other cafe. In other words, he wants to find such a cafe that he hasn't been there for as long as possible. Help Vlad to find that cafe.

Input

In first line there is one integer n ($1 \le n \le 2 \cdot 10^5$) — number of cafes indices written by Vlad.

In second line, n numbers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 2 \cdot 10^5$) are written — indices of cafes in order of being visited by Vlad. Vlad could visit some cafes more than once. Note that in numeration, some indices could be omitted.

Output

Print one integer — index of the cafe that Vlad hasn't visited for as long as possible.

Examples

input

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

output

```
3
```

input

```
6
2 1 2 2 4 1
```

output

```
2
```

Note

In first test, there are three cafes, and the last visits to cafes with indices 1 and 2 were after the last visit to cafe with index 3; so this cafe is the answer.

In second test case, there are also three cafes, but with indices 1, 2 and 4. Cafes with indices 1 and 4 were visited after the last visit of cafe with index 2, so the answer is 2. Note that Vlad could omit some numbers while numerating the cafes.

C. Petya and Catacombs

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

A very brave explorer Petya once decided to explore Paris catacombs. Since Petya is not really experienced, his exploration is just walking through the catacombs.

Catacombs consist of several rooms and bidirectional passages between some pairs of them. Some passages can connect a room to itself and since the passages are built on different depths they do not intersect each other. Every minute Petya arbitrary chooses a passage from the room he is currently in and then reaches the room on the other end of the passage in exactly one minute. When he enters a room at minute i, he makes a note in his logbook with number t_i:

- If Petya has visited this room before, he writes down the minute he was in this room last time;
- Otherwise, Petya writes down an arbitrary non-negative integer strictly less than current minute i.

Initially, Petya was in one of the rooms at minute 0, he didn't write down number t_0.

At some point during his wandering Petya got tired, threw out his logbook and went home. Vasya found his logbook and now he is curious: what is the minimum possible number of rooms in Paris catacombs according to Petya's logbook?

Input

The first line contains a single integer n $(1 \le n \le 2 \cdot 10^5)$ — then number of notes in Petya's logbook.

The second line contains n non-negative integers $t_1, t_2, \ldots, t_n$ $(0 \le t_i < i)$ — notes in the logbook.

Output

In the only line print a single integer — the minimum possible number of rooms in Paris catacombs.

Examples

input

```
2
0 0
```

output

```
2
```

input

```
5
0 1 0 1 3
```

output

```
3
```

Note

In the first sample, sequence of rooms Petya visited could be, for example $1 \to 1 \to 2$, $1 \to 2 \to 1$ or $1 \to 2 \to 3$. The minimum possible number of rooms is 2.

In the second sample, the sequence could be $1 \to 2 \to 3 \to 1 \to 2 \to 1$.

D. Restoration of string

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

A substring of some string is called the most frequent, if the number of its occurrences is not less than number of occurrences of any other substring.

You are given a set of strings. A string (not necessarily from this set) is called good if all elements of the set are the most frequent substrings of this string. Restore the non-empty good string with minimum length. If several such strings exist, restore lexicographically minimum string. If there are no good strings, print "`NO`" (without quotes).

A substring of a string is a contiguous subsequence of letters in the string. For example, "`ab`", "`c`", "`abc`" are substrings of string "`abc`", while "`ac`" is not a substring of that string.

The number of occurrences of a substring in a string is the number of starting positions in the string where the substring occurs. These occurrences could overlap.

String a is lexicographically smaller than string b, if a is a prefix of b, or a has a smaller letter at the first position where a and b differ.

Input

The first line contains integer n ($1 \leq n \leq 10^5$) — the number of strings in the set.

Each of the next n lines contains a non-empty string consisting of lowercase English letters. It is guaranteed that the strings are distinct.

The total length of the strings doesn't exceed 10^5.

Output

Print the non-empty good string with minimum length. If several good strings exist, print lexicographically minimum among them. Print "`NO`" (without quotes) if there are no good strings.

Examples

input

```
4
mail
ai
lru
cf
```

output

```
cfmailru
```

input

```
3
kek
preceq
cheburek
```

output

```
NO
```

Note

One can show that in the first sample only two good strings with minimum length exist: "`cfmailru`" and "`mailrucf`". The first string is lexicographically minimum.

E. Maximum Element

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

One day Petya was solving a very interesting problem. But although he used many optimization techniques, his solution still got Time limit exceeded verdict. Petya conducted a thorough analysis of his program and found out that his function for finding maximum element in an array of n positive integers was too slow. Desperate, Petya decided to use a somewhat unexpected optimization using parameter k, so now his function contains the following code:

```
int fast_max(int n, int a[]) {
    int ans = 0;
    int offset = 0;
    for (int i = 0; i < n; ++i)
        if (ans < a[i]) {
            ans = a[i];
            offset = 0;
        } else {
            offset = offset + 1;
            if (offset == k)
                return ans;
        }
    return ans;
}
```

That way the function iteratively checks array elements, storing the intermediate maximum, and if after k consecutive iterations that maximum has not changed, it is returned as the answer.

Now Petya is interested in fault rate of his function. He asked you to find the number of permutations of integers from 1 to n such that the return value of his function on those permutations is not equal to n. Since this number could be very big, output the answer modulo $10^9 + 7$.

Input

The only line contains two integers n and k ($1 \le n, k \le 10^6$), separated by a space — the length of the permutations and the parameter k.

Output

Output the answer to the problem modulo $10^9 + 7$.

Examples

input

```
5 2
```

output

```
22
```

input

```
5 3
```

output

```
6
```

input

```
6 3
```

output

```
84
```

Note

Permutations from second example:

$[4, 1, 2, 3, 5]$, $[4, 1, 3, 2, 5]$, $[4, 2, 1, 3, 5]$, $[4, 2, 3, 1, 5]$, $[4, 3, 1, 2, 5]$, $[4, 3, 2, 1, 5]$.

F. Symmetric Projections

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

You are given a set of n points on the plane. A line containing the origin is called good, if projection of the given set to this line forms a symmetric multiset of points. Find the total number of good lines.

Multiset is a set where equal elements are allowed.

Multiset is called symmetric, if there is a point P on the plane such that the multiset is centrally symmetric in respect of point P.

Input

The first line contains a single integer n $(1 \le n \le 2000)$ — the number of points in the set.

Each of the next n lines contains two integers x_i and y_i ($-10^6 \le x_i, y_i \le 10^6$) — the coordinates of the points. It is guaranteed that no two points coincide.

Output

If there are infinitely many good lines, print `-1`.

Otherwise, print single integer — the number of good lines.

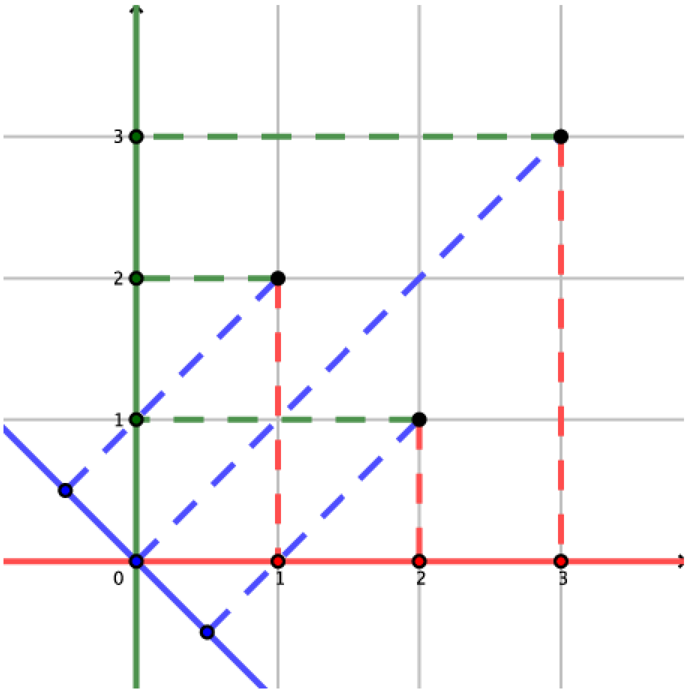
Examples

input
```
3
1 2
2 1
3 3
```
output
```
3
```

input
```
2
4 3
1 2
```
output
```
-1
```

Note

Picture to the first sample test:



In the second sample, any line containing the origin is good.