A. Mahmoud and a Triangle

2 seconds, 256 megabytes

Mahmoud has n line segments, the i-th of them has length a_i. Ehab challenged him to use **exactly** 3 line segments to form a non-degenerate triangle. Mahmoud doesn't accept challenges unless he is sure he can win, so he asked you to tell him if he should accept the challenge. Given the lengths of the line segments, check if he can choose exactly 3 of them to form a non-degenerate triangle.

Mahmoud should use exactly 3 line segments, he can't concatenate two line segments or change any length. A non-degenerate triangle is a triangle with positive area.

Input

The first line contains single integer n $(3 \le n \le 10^5)$ — the number of line segments Mahmoud has.

The second line contains n integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^9)$ — the lengths of line segments Mahmoud has.

Output

In the only line print "YES" if he can choose exactly three line segments and form a non-degenerate triangle with them, and "NO" otherwise.

input

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

output

```
YES
```

input

```
3
4 1 2
```

output

```
NO
```

For the first example, he can use line segments with lengths 2, 4 and 5 to form a non-degenerate triangle.

B. Towers

2 seconds, 256 megabytes

Little Vasya has received a young builder’s kit. The kit consists of several wooden bars, the lengths of all of them are known. The bars can be put one on the top of the other if their lengths are the same.

Vasya wants to construct the minimal number of towers from the bars. Help Vasya to use the bars in the best way possible.

Input

The first line contains an integer N $(1 \le N \le 1000)$ — the number of bars at Vasya’s disposal. The second line contains N space-separated integers l_i — the lengths of the bars. All the lengths are natural numbers not exceeding 1000.

Output

In one line output two numbers — the height of the largest tower and their total number. Remember that Vasya should use all the bars.

input

```
3
1 2 3
```

output

```
1 3
```

input

```
4
6 5 6 7
```

output

```
2 3
```

C. Queue

1 second, 256 megabytes

Little girl Susie went shopping with her mom and she wondered how to improve service quality.

There are n people in the queue. For each person we know time t_i needed to serve him. A person will be disappointed if the time he waits is more than the time needed to serve him. The time a person waits is the total time when all the people who stand in the queue in front of him are served. Susie thought that if we swap some people in the queue, then we can decrease the number of people who are disappointed.

Help Susie find out what is the maximum number of not disappointed people can be achieved by swapping people in the queue.

Input

The first line contains integer n $(1 \le n \le 10^5)$.

The next line contains n integers t_i $(1 \le t_i \le 10^9)$, separated by spaces.

Output

Print a single number — the maximum number of not disappointed people in the queue.

input

```
5
15 2 1 5 3
```

output

```
4
```

Value 4 is achieved at such an arrangement, for example: $1, 2, 3, 5, 15$. Thus, you can make everything feel not disappointed except for the person with time 5.

D. Soldier and Badges

3 seconds, 256 megabytes

Colonel has n badges. He wants to give one badge to every of his n soldiers. Each badge has a *coolness factor*, which shows how much it's owner reached. Coolness factor can be increased by one for the cost of one coin.

For every pair of soldiers one of them should get a badge with strictly higher factor than the second one. Exact values of their factors aren't important, they just need to have distinct factors.

Colonel knows, which soldier is supposed to get which badge initially, but there is a problem. Some of badges may have the same factor of coolness. Help him and calculate how much money has to be paid for making all badges have different factors of coolness.

Input

First line of input consists of one integer n $(1 \le n \le 3000)$.

Next line consists of n integers a_i $(1 \le a_i \le n)$, which stand for coolness factor of each badge.

Output

Output single integer — minimum amount of coins the colonel has to pay.

input

```
4
1 3 1 4
```

output

```
1
```

input

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

output

```
2
```

In first sample test we can increase factor of first badge by 1.

In second sample test we can increase factors of the second and the third badge by 1.

E. Next Test

2 seconds, 256 megabytes

«Polygon» is a system which allows to create programming tasks in a simple and professional way. When you add a test to the problem, the corresponding form asks you for the test index. As in most cases it is clear which index the next test will have, the system suggests the default value of the index. It is calculated as the smallest positive integer which is not used as an index for some previously added test.

You are to implement this feature. Create a program which determines the default index of the next test, given the indexes of the previously added tests.

Input

The first line contains one integer n $(1 \le n \le 3000)$ — the amount of previously added tests. The second line contains n distinct integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 3000)$ — indexes of these tests.

Output

Output the required default value for the next test index.

input

```
3
1 7 2
```

output

```
3
```

F. Vanya and Exams

1 second, 256 megabytes

Vanya wants to pass n exams and get the academic scholarship. He will get the scholarship if the average grade mark for all the exams is at least avg. The exam grade cannot exceed r. Vanya has passed the exams and got grade a_i for the i-th exam. To increase the grade for the i-th exam by 1 point, Vanya must write b_i essays. He can raise the exam grade multiple times.

What is the minimum number of essays that Vanya needs to write to get scholarship?

Input

The first line contains three integers n, r, avg $(1 \le n \le 10^5,\ 1 \le r \le 10^9,\ 1 \le avg \le min(r, 10^6))$ — the number of exams, the maximum grade and the required grade point average, respectively.

Each of the following n lines contains space-separated integers a_i and b_i $(1 \le a_i \le r,\ 1 \le b_i \le 10^6)$.

Output

In the first line print the minimum number of essays.

input

```
5 5 4
5 2
4 7
3 1
3 2
2 5
```

output

```
4
```

input

```
2 5 4
5 2
5 2
```

output

```
0
```

In the first sample Vanya can write 2 essays for the 3rd exam to raise his grade by 2 points and 2 essays for the 4th exam to raise his grade by 1 point.

In the second sample, Vanya doesn't need to write any essays as his general point average already is above average.

G. Vasya and Basketball

2 seconds, 256 megabytes

Vasya follows a basketball game and marks the distances from which each team makes a throw. He knows that each successful throw has value of either 2 or 3 points. A throw is worth 2 points if the distance it was made from doesn't exceed some value of d meters, and a throw is worth 3 points if the distance is larger than d meters, where d is some **non-negative** integer.

Vasya would like the advantage of the points scored by the first team (the points of the first team minus the points of the second team) to be maximum. For that he can mentally choose the value of d. Help him to do that.

Input

The first line contains integer n $(1 \le n \le 2 \cdot 10^5)$ — the number of throws of the first team. Then follow n integer numbers — the distances of throws a_i $(1 \le a_i \le 2 \cdot 10^9)$.

Then follows number m $(1 \le m \le 2 \cdot 10^5)$ — the number of the throws of the second team. Then follow m integer numbers — the distances of throws of b_i $(1 \le b_i \le 2 \cdot 10^9)$.

Output

Print two numbers in the format `a:b` — the score that is possible considering the problem conditions where the result of subtraction a - b is maximum. If there are several such scores, find the one in which number a is maximum.

input

```
3
1 2 3
2
5 6
```

output

```
9:6
```

input

```
5
6 7 8 9 10
5
1 2 3 4 5
```

output

```
15:10
```

H. Points on Line

2 seconds, 256 megabytes

Little Petya likes points a lot. Recently his mom has presented him n points lying on the line OX. Now Petya is wondering in how many ways he can choose three distinct points so that the distance between the two farthest of them doesn't exceed d.

Note that the order of the points inside the group of three chosen points doesn't matter.

Input

The first line contains two integers: n and d ($1 \leq n \leq 10^5$; $1 \leq d \leq 10^9$). The next line contains n integers $x_1, x_2, ..., x_n$, their absolute value doesn't exceed 10^9 — the x-coordinates of the points that Petya has got.

It is guaranteed that the coordinates of the points in the input **strictly increase**.

Output

Print a single integer — the number of groups of three points, where the distance between two farthest points doesn't exceed d.

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

input

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

output

```
4
```

input

```
4 2
-3 -2 -1 0
```

output

```
2
```

input

```
5 19
1 10 20 30 50
```

output

```
1
```

In the first sample any group of three points meets our conditions.

In the seconds sample only 2 groups of three points meet our conditions: `{-3, -2, -1}` and `{-2, -1, 0}`.

In the third sample only one group does: `{1, 10, 20}`.

I. PolandBall and Game

1 second, 256 megabytes

PolandBall is playing a game with EnemyBall. The rules are simple. Players have to say words in turns. You cannot say a word which was already said. PolandBall starts. The Ball which can't say a new word loses.

You're given two lists of words familiar to PolandBall and EnemyBall. Can you determine who wins the game, if both play optimally?

Input

The first input line contains two integers n and m ($1 \leq n, m \leq 10^3$) — number of words PolandBall and EnemyBall know, respectively.

Then n strings follow, one per line — words familiar to PolandBall.

Then m strings follow, one per line — words familiar to EnemyBall.

Note that one Ball **cannot** know a word more than once (strings are unique), but some words **can** be known by both players.

Each word is non-empty and consists of no more than 500 lowercase English alphabet letters.

Output

In a single line of print the answer — "`YES`" if PolandBall wins and "`NO`" otherwise. Both Balls play optimally.

input

```
5 1
polandball
is
a
cool
character
nope
```

output

```
YES
```

input

```
2 2
kremowka
wadowicka
kremowka
wiedenska
```

output

```
YES
```

input

```
1 2
a
a
b
```

output

```
NO
```

In the first example PolandBall knows much more words and wins effortlessly.

In the second example if PolandBall says `kremowka` first, then EnemyBall cannot use that word anymore. EnemyBall can only say `wiedenska`. PolandBall says `wadowicka` and wins.

J. To Add or Not to Add

2 seconds, 256 megabytes

A piece of paper contains an array of n integers $a_1, a_2, ..., a_n$. Your task is to find a number that occurs the maximum number of times in this array.

However, before looking for such number, you are allowed to perform not more than k following operations — choose an arbitrary element from the array and add 1 to it. In other words, you are allowed to increase some array element by 1 no more than k times (you are allowed to increase the same element of the array multiple times).

Your task is to find the maximum number of occurrences of some number in the array after performing no more than k allowed operations. If there are several such numbers, your task is to find the minimum one.

Input

The first line contains two integers n and k ($1 \leq n \leq 10^5$; $0 \leq k \leq 10^9$) — the number of elements in the array and the number of operations you are allowed to perform, correspondingly.

The third line contains a sequence of n integers $a_1, a_2, ..., a_n$ ($|a_i| \leq 10^9$) — the initial array. The numbers in the lines are separated by single spaces.

Output

In a single line print two numbers — the maximum number of occurrences of some number in the array after at most k allowed operations are performed, and the minimum number that reaches the given maximum. Separate the printed numbers by whitespaces.

input

```
5 3
6 3 4 0 2
```

output

```
3 4
```

input

```
3 4
5 5 5
```

output

```
3 5
```

input

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

output

```
4 2
```

In the first sample your task is to increase the second element of the array once and increase the fifth element of the array twice. Thus, we get sequence $6, 4, 4, 0, 4$, where number 4 occurs 3 times.

In the second sample you don't need to perform a single operation or increase each element by one. If we do nothing, we get array $5, 5, 5$, if we increase each by one, we get $6, 6, 6$. In both cases the maximum number of occurrences equals 3. So we should do nothing, as number 5 is less than number 6.

In the third sample we should increase the second array element once and the fifth element once. Thus, we get sequence $3, 2, 2, 2, 2$, where number 2 occurs 4 times.