Codeforces Round #340 (Div. 2)

A. Elephant

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

An elephant decided to visit his friend. It turned out that the elephant's house is located at point 0 and his friend's house is located at point $x (x > 0)$ of the coordinate line. In one step the elephant can move 1, 2, 3, 4 or 5 positions forward. Determine, what is the minimum number of steps he need to make in order to get to his friend's house.

Input

The first line of the input contains an integer x ($1 \le x \le 1\,000\,000$) — The coordinate of the friend's house.

Output

Print the minimum number of steps that elephant needs to make to get from point 0 to point x.

Examples

input
5
output
1

input
12
output
3

Note

In the first sample the elephant needs to make one step of length 5 to reach the point x.

In the second sample the elephant can get to point x if he moves by 3, 5 and 4. There are other ways to get the optimal answer but the elephant cannot reach x in less than three moves.

B. Chocolate

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

Bob loves everything sweet. His favorite chocolate bar consists of pieces, each piece may contain a nut. Bob wants to break the bar of chocolate into multiple pieces so that each part would contain **exactly** one nut and any break line goes between two adjacent pieces.

You are asked to calculate the number of ways he can do it. Two ways to break chocolate are considered distinct if one of them contains a break between some two adjacent pieces and the other one doesn't.

Please note, that if Bob doesn't make any breaks, all the bar will form one piece and it still has to have exactly one nut.

Input

The first line of the input contains integer n ($1 \le n \le 100$) — the number of pieces in the chocolate bar.

The second line contains n integers a_i ($0 \le a_i \le 1$), where 0 represents a piece without the nut and 1 stands for a piece with the nut.

Output

Print the number of ways to break the chocolate into multiple parts so that each part would contain exactly one nut.

Examples

input
3 0 1 0
output
1

input
5 1 0 1 0 1
output
4

Note

In the first sample there is exactly one nut, so the number of ways equals 1 — Bob shouldn't make any breaks.

In the second sample you can break the bar in four ways:

`10|10|1`

`1|010|1`

`10|1|01`

`1|01|01`

C. Watering Flowers

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

A flowerbed has many flowers and two fountains.

You can adjust the water pressure and set any values $r_1(r_1 \geq 0)$ and $r_2(r_2 \geq 0)$, giving the distances at which the water is spread from the first and second fountain respectively. You have to set such r_1 and r_2 that all the flowers are watered, that is, for each flower, the distance between the flower and the first fountain doesn't exceed r_1, or the distance to the second fountain doesn't exceed r_2. It's OK if some flowers are watered by both fountains.

You need to decrease the amount of water you need, that is set such r_1 and r_2 that all the flowers are watered and the ${r_1}^2 + {r_2}^2$ is minimum possible. Find this minimum value.

Input

The first line of the input contains integers n, x_1, y_1, x_2, y_2 ($1 \leq n \leq 2000$, $-10^7 \leq x_1, y_1, x_2, y_2 \leq 10^7$) — the number of flowers, the coordinates of the first and the second fountain.

Next follow n lines. The i-th of these lines contains integers x_i and y_i ($-10^7 \leq x_i, y_i \leq 10^7$) — the coordinates of the i-th flower.

It is guaranteed that all $n + 2$ points in the input are distinct.

Output

Print the minimum possible value ${r_1}^2 + {r_2}^2$. Note, that in this problem optimal answer is always integer.

Examples

input
2 -1 0 5 3 0 2 5 2
output
6

input
4 0 0 5 0 9 4 8 3 -1 0 1 4
output
33

Note

The first sample is (${r_1}^2 = 5$, ${r_2}^2 = 1$):

The second sample is (${r_1}^2 = 1$, ${r_2}^2 = 32$):

D. Polyline

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

There are three points marked on the coordinate plane. The goal is to make a simple polyline, without self-intersections and self-touches, such that it passes through all these points. Also, the polyline must consist of only segments parallel to the coordinate axes. You are to find the minimum number of segments this polyline may consist of.

Input

Each of the three lines of the input contains two integers. The i-th line contains integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) — the coordinates of the i-th point. It is guaranteed that all points are distinct.

Output

Print a single number — the minimum possible number of segments of the polyline.

Examples

input
1 -1 1 1 1 2
output
1

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

input
1 1 2 3 3 2
output
3

Note

The variant of the polyline in the first sample:

The variant of the polyline in the second sample:

The variant of the polyline in the third sample:

E. XOR and Favorite Number

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bob has a favorite number k and a_i of length n. Now he asks you to answer m queries. Each query is given by a pair l_i and r_i and asks you to count the number of pairs of integers i and j, such that $l \le i \le j \le r$ and the xor of the numbers $a_i, a_{i+1}, \ldots, a_j$ is equal to k.

Input

The first line of the input contains integers n, m and k ($1 \le n, m \le 100\,000$, $0 \le k \le 1\,000\,000$) — the length of the array, the number of queries and Bob's favorite number respectively.

The second line contains n integers a_i ($0 \le a_i \le 1\,000\,000$) — Bob's array.

Then m lines follow. The i-th line contains integers l_i and r_i ($1 \le l_i \le r_i \le n$) — the parameters of the i-th query.

Output

Print m lines, answer the queries in the order they appear in the input.

Examples

input
6 2 3 1 2 1 1 0 3 1 6 3 5

output
7 0

input
5 3 1 1 1 1 1 1 1 5 2 4 1 3

output
9 4 4

Note

In the first sample the suitable pairs of i and j for the first query are: $(1, 2)$, $(1, 4)$, $(1, 5)$, $(2, 3)$, $(3, 6)$, $(5, 6)$, $(6, 6)$. Not a single of these pairs is suitable for the second query.

In the second sample xor equals 1 for all subarrays of an odd length.