Building a Fence

Leopold is indeed a lucky fellow. He just won a huge estate in the lottery. The estate contains several grand buildings in addition to the main mansion, in which he intends to live from now on. However, the estate lacks a fence protecting the premises from trespassers, which concerns Leopold to a great extent. He wants to build a fence and, in order to save money, he decides it is sufficient to have a fence that encloses the main mansion, except for one important restriction: the fence must not lie too close to any of the buildings. To be precise, seen from above, each building is enclosed in a surrounding forbidden rectangle within which no part of the fence may lie. The rectangles' sides are parallel to the x- and y-axis. Each part of the fence must also be parallel either to the x-axis or the y-axis.

Help Leopold to compute the minimum length of any allowed fence enclosing the main mansion.

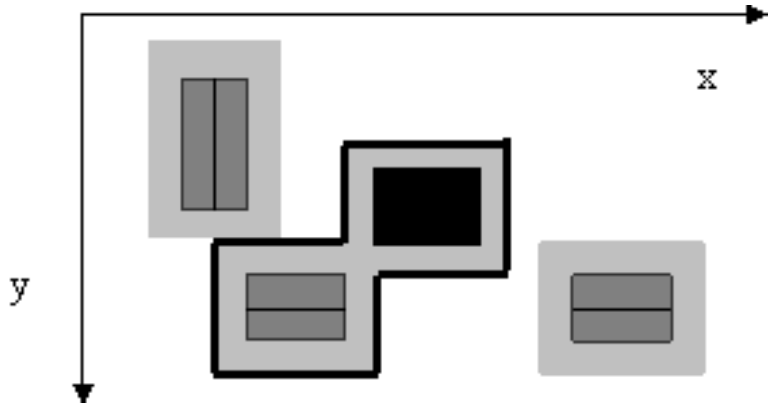

Figure 1: The main mansion (black) and three other buildings with surrounding forbidden rectangles. The thick black line shows a shortest allowed fence enclosing the main mansion.

Input

The input is read from a text file named `fence.in`. The first line of the input file contains a positive integer m ($1 \leq m \leq 100$), the number of buildings of the estate. Then follow m lines each describing a forbidden rectangle enclosing a building. Each row contains four space-separated integers tx, ty, bx, and by, where (tx, ty) are the coordinates of the upper left corner and (bx, by) the coordinates of the bottom right corner of the rectangle. All coordinates obey $0 \leq tx < bx \leq 10{,}000$ and $0 \leq ty < by \leq 10{,}000$. The first rectangle is the forbidden rectangle enclosing the main mansion.

Output

The output is written into a text file named `fence.out`. It contains one line with a single positive integer equal to the minimum length of any allowed fence enclosing the main mansion.

Example

fence.in	fence.out
4 8 4 13 8 2 1 6 7 4 7 9 11 14 7 19 11	32

Grading

In 30% of the testcases $m \leq 10$ holds.

Connected Points

Consider a regular grid of $3 \times N$ points. Every point in the grid has up to eight neighboring points (see Fig. 1).

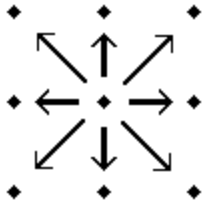

Figure 1: Neighboring points (marked by arrows).

We are interested in counting the number of different ways to connect the points of the grid to form a polygon that fulfills the following conditions:

1. The set of vertices of the polygon consists of all $3 \times N$ points.
2. Adjacent vertices of the polygon are neighboring points in the grid.
3. Each polygon is simple, i.e. there must not be any self-intersections.

Two possible polygons for $N = 6$ are given in the Fig. 2.

Figure 2: Two possible connections of points for $N = 6$.

Write a program that calculates for a given N the number of possible ways to connect the points as described modulo 1,000,000,000.

Input

The input is read from a text file named `points.in`. The first and only line contains one positive integer N ($N \leq 1{,}000{,}000{,}000$).

Output

The output is written into a text file named `points.out`. The only line to be written contains the remainder of the number of ways to connect the points modulo 1,000,000,000.

Examples

points.in	points.out
3	8

points.in	points.out
4	40

Grading

- 30% of the test cases have values of $N \leq 200$.
- 70% of the test cases have values of $N \leq 100{,}000$.

Sequence

We are given a sequence $a_1, ..., a_n$. We can manipulate this sequence using the operation $reduce(i)$, which replaces elements a_i and a_{i+1} with a single element $max(a_i, a_{i+1})$, resulting in a new shorter sequence. The cost of this operation is $max(a_i, a_{i+1})$. After $n - 1$ operations $reduce$, we obtain a sequence of length 1. Our task is to compute the cost of the optimal reducing scheme, i.e. the sequence of $reduce$ operations with minimal cost leading to a sequence of length 1.

Input

The input is read from a text file named `sequence.in`. The first line contains n ($1 \leq n \leq 1{,}000{,}000$), the length of the sequence. The following n lines contain one integer a_i, the elements of the sequence ($0 \leq a_i \leq 1{,}000{,}000{,}000$).

Output

The output is written into a text file named `sequence.out`. In the first and only line of the output print the minimal cost of reducing the sequence to a single element.

Example

sequence.in	sequence.out
3 1 2 3	5

Grading

In 30% of the test cases $n \leq 500$ holds.
In 50% of the test cases $n \leq 20{,}000$ holds.