A Architecture

Your brother has won an award at the recent Breakthroughs in Architectural Problems Conference and has been given the once in a lifetime opportunity of redesigning the city center of his favorite city Nijmegen. Since the most striking parts of a city's layout are the skylines, your brother has started by drawing ideas for how he wants the northern and eastern skylines of Nijmegen to look. However, some of his proposals look rather outlandish, and you are starting to wonder whether his designs are possible.

For his design, your brother has put an $R \times C$ grid on the city. Each cell of the city will contain a building of a certain height. The eastern skyline is given by the tallest building in each of the R rows, and the northern skyline is given by the tallest building in each of the C columns.

A pair of your brother's drawings of skylines is possible if and only if there exists some way of assigning building heights to the grid cells such that the resulting skylines match these drawings.

Figure A.1 shows a possible city with the northern and eastern skylines exactly as given in the input of the first sample.

1	2	3	4
0	1	2	3
1	2	1	1
1	0	1	1

Figure A.1: Example city showing sample 1 has a valid solution.

Input

- The first line consists of two integers $1 \leq R, C \leq 100$, the number of rows and columns in the grid.
- The second line consists of R integers $x_1, \ldots, x_R$ describing the eastern skyline ($0 \leq x_i \leq 1000$ for all i).
- The third line consists of C integers $y_1, \ldots, y_C$ describing the northern skyline ($0 \leq y_j \leq 1000$ for all j).

Output

Output one line containing the string `possible` if there exists a city design that produces the specified skyline, and `impossible` otherwise.

Sample Input 1	**Sample Output 1**

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

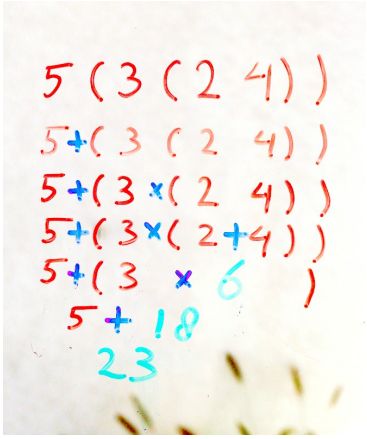
Sample Input 2

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

Sample Output 2

```
impossible
```

B Bracket Sequence



Two great friends, Eddie John and Kris Cross, are attending the Brackets Are Perfection Conference. They wholeheartedly agree with the main message of the conference and they are delighted with all the new things they learn about brackets.

One of these things is a *bracket sequence*. If you want to do a computation with + and ×, you usually write it like so:

$$(2 \times (2 + 1 + 0 + 1) \times 1) + 3 + 2.$$

The brackets are only used to group multiplications and additions together. This means that you can remove all the operators, as long as you remember that addition is used for numbers outside any parentheses! A *bracket sequence* can then be shortened to

$$(\ 2\ (\ 2\ 1\ 0\ 1\ )\ 1\ )\ 3\ 2.$$

That is much better, because it saves on writing all those operators. Reading bracket sequences is easy, too. Suppose you have the following bracket sequence

$$5\ 2\ (\ 3\ 1\ (\ 2\ 2\ )\ (\ 3\ 3\ )\ 1\ ).$$

You start with addition, so this is the same as the following:

$$5 + 2 + (\ 3\ 1\ (\ 2\ 2\ )\ (\ 3\ 3\ )\ 1\ ).$$

You know the parentheses group a multiplication, so this is equal to

$$5 + 2 + (3 \times 1 \times (\ 2\ 2\ ) \times (\ 3\ 3\ ) \times 1).$$

Then there is another level of parentheses: that groups an operation within a multiplication, so the operation must be addition.

$$5 + 2 + (3 \times 1 \times (2 + 2) \times (3 + 3) \times 1) = 5 + 2 + (3 \times 1 \times 4 \times 6 \times 1) = 5 + 2 + 72 = 79.$$

Since bracket sequences are so much easier than normal expressions with operators, it should be easy to evaluate some big ones. We will even allow you to write a program to do it for you.

Note that ( ) is not a valid bracket sequence, nor a subsequence of any valid bracket sequence.

Input

- One line containing a single integer $1 \leq n \leq 3 \cdot 10^5$.
- One line consisting of n tokens, each being either `(`, `)`, or an integer $0 \leq x < 10^9 + 7$. It is guaranteed that the tokens form a bracket sequence.

Output

Output the value of the given bracket sequence. Since this may be very large, you should print it modulo $10^9 + 7$.

Sample Input 1

```
2
2 3
```

Sample Output 1

```
5
```

Sample Input 2

```
8
( 2 ( 2 1 ) ) 3
```

Sample Output 2

```
9
```

Sample Input 3

```
4
( 12 3 )
```

Sample Output 3

```
36
```

Sample Input 4

```
6
( 2 ) ( 3 )
```

Sample Output 4

```
5
```

Sample Input 5

```
6
( ( 2 3 ) )
```

Sample Output 5

```
5
```

Sample Input 6

```
11
1 ( 0 ( 583920 ( 2839 82 ) ) )
```

Sample Output 6

```
1
```

C Canyon Crossing

The Bridge And Passageway Creators are responsible for making new paths through the local mountains. They have approved your plan to build a new route through your favorite canyon. You feverishly start working on this beautiful new path, when you realize you failed to take into account the flow of a nearby river: the canyon is flooded! Apparently this happens once every blue moon, making some parts of the path inaccessible. Because of this, you want to build a path such that the lowest point on the path is as high as possible. You quickly return to the village and use all of your money to buy rope bridges. You plan to use these to circumvent the lowest parts of the canyon.

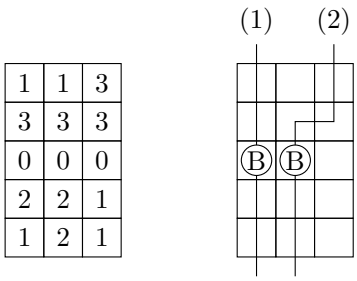


Figure C.1: Canyon and two possible paths with minimal height 1 and 2 for sample input 1. The `B` indicate bridges.

Your map of the canyon consists of a rectangular grid of cells, each containing a number giving the height of the terrain at that cell. The path will go from the south side of the canyon (bottom on your map) to the north side (top of your map), moving through a connected sequence of cells. Two cells are considered connected if and only if they share an edge. In particular, two diagonally touching cells are not considered to be connected. This means that for any cell not on the edge of the map, there are 4 other cells connected to it. The left of figure C.1 contains the map for the first sample input.

The path through the canyon can start on any of the bottom cells of the grid, and end on any of the cells in the top tow, like the two paths on the right in C.1. The lowest height is given by the lowest height of any of the cells the paths goes through. Each bridge can be used to cross exactly one cell. This cell is then not taken into account when calculating the minimal height of the path. Note that is allowed to chain multiple bridges to use them to cross multiple cells,

Given the map of the canyon and the number of bridges available, find the lowest height of an optimal path.

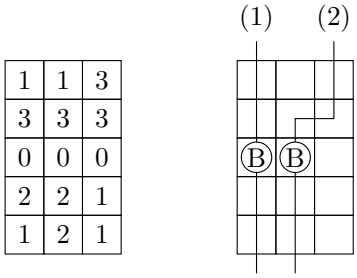


Figure C.2: Canyon and an optimal path for sample input 2.

Input

- A single line containing three integers: $1 \leq R \leq 1000$ and $1 \leq C \leq 1000$, the size of the map, and $0 \leq K \leq R - 1$, the number of bridges you can build.
- This is followed by R lines each containing C integers. The j-th integer on the i-th line corresponds to the height $0 \leq H_{i,j} \leq 10^9$ of the canyon at point (i, j). The first line corresponds to the northern edge of the canyon, the last line to the southern edge.

Output

Output a single integer, the lowest height of the optimal path.

Sample Input 1

```
5 3 1
1 1 3
3 3 3
0 0 0
2 2 1
1 2 1
```

Sample Output 1

```
2
```

Sample Input 2

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

Sample Output 2

```
2
```

Sample Input 3

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

Sample Output 3

```
4
```

H Hexagonal Rooks

It is game night and Alice and Bob are playing chess. After beating Bob at chess several times, Alice suggests they should play a chess variant instead called *hexagonal chess*. Although the game is very rarely played nowadays, Alice knows the rules very well and has obtained a hexagonal chessboard from her subscription to the magazine of Bizarre Artifacts for Playing Chess.

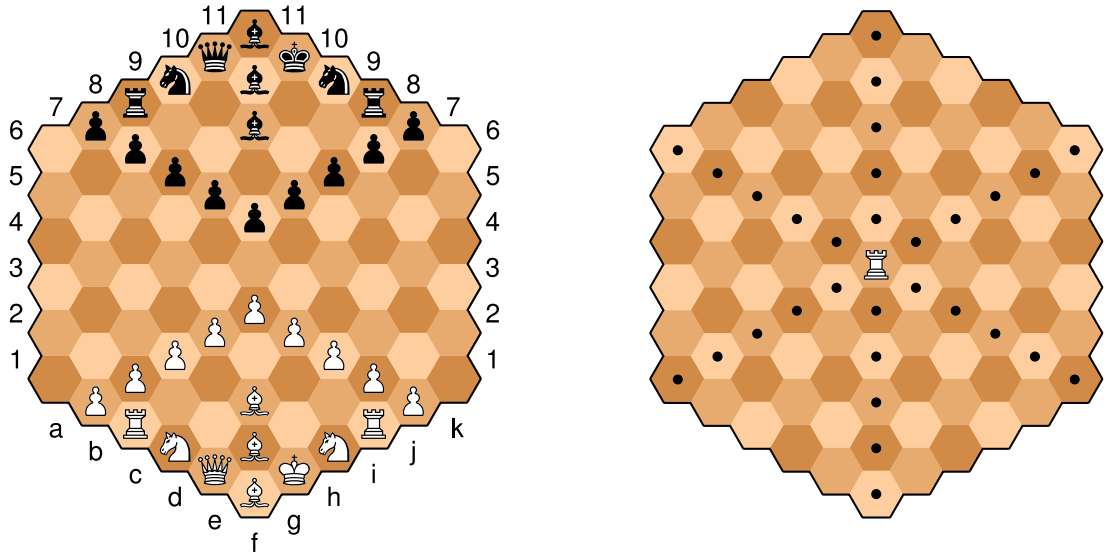


Figure H.1: The field naming of the hexagonal chess board and the directions in which a rook can move.

The hexagonal chess board, shown above, consists of 91 hexagonal cells arranged in the shape of a hexagon with side length 6 as depicted in the above diagrams. The board is divided into 11 columns, each called a file, and the files are labeled `a` to `k` from left to right. It is also divided into 11 v-shaped rows, each called a rank, which are labeled `1` to `11` from bottom to top. The unique cell in file x and rank y is then denoted by the coordinate xy. For example, rank `11` contains only a single cell `f11` and rank `7` is occupied entirely by the black player's pawns.

Alice begins by explaining how all the pieces move. The simplest piece is the rook, which can move an arbitrary positive number of steps in a straight line in the direction of any of its 6 adjacent cells, as depicted in the figure on the right. Bob immediately realises that the hexagonal rook already is more difficult to work with than its regular chess counterpart.

In order to attack one of the opponents pieces, it is useful to know which cells his rook can move to such that it attacks the opposing piece. The more of these cells there are, the more valuable the current position of his rook is. However, calculating this number is too much for Bob. After losing so many games of regular chess, Alice allows Bob to use a program to assist in his rook placement. While Alice explains the rest of the game you get busy coding.

As a small simplification, Bob will compute the number of ways his rook can move to the destination cell assuming there are no other pieces on the board, not even the piece he wants to attack.

Input

- The input consists of one line, containing two different coordinates on the hexagonal chess board, the current positions of your rook and the piece you want to attack.

Output

Output a single integer, the number of ways the rook can move from its current position to the position of the piece it wants to attack in exactly two moves, assuming there are no other pieces on the board.

Sample Input 1	**Sample Output 1**
`c4 h4`	`6`

Sample Input 2	**Sample Output 2**
`a1 a2`	`5`

I Inquiry I

The Bureau for Artificial Problems in Competitions wants you to solve the following problem: Given n positive integers $a_1, \ldots, a_n$, what is the maximal value of

$$\left(a_1^2 + \cdots + a_k^2\right) \cdot (a_{k+1} + \cdots + a_n)?$$

Input

- A single line containing an integer $2 \leq n \leq 10^6$.
- Then follow n lines, the ith of which contains the integer $1 \leq a_i \leq 100$.

Output

Output the maximal value of the given expression.

Sample Input 1	**Sample Output 1**
5 2 1 4 3 5	168

Sample Input 2	**Sample Output 2**
2 1 1	1

Sample Input 3	**Sample Output 3**
10 8 5 10 9 1 4 12 6 3 13	10530