Knights

(Running time - 1s.)

We are given a chess-board of size $n*n$, from which some fields have been removed. The task is to determine the maximum number of knights that can be placed on the remaining fields of the board in such a way that none of them check each other.

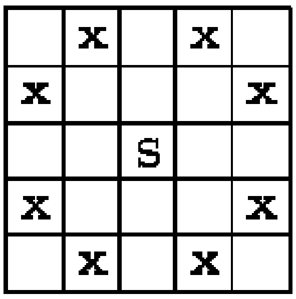


Fig.1: A knight placed on the field S checks fields marked with x.

Task

Write a program, that:

- reads the description of a chess-board with some fields removed, from the input file `kni.in`,
- determines the maximum number of knights that can be placed on the chess-board in such a way that none of them check each other,
- writes the result to the output file `kni.out`.

Input

The first line of the input file `kni.in` contains two integers n and m, separated by a single space, $1<=n<=200, 0<=m<n^2$; n is the chess-board size and m is the number of removed fields. Each of the following m lines contains two integers: x and y, separated by a single space, $1<=x,y<=n$ -- these are the coordinates of the removed fields. The coordinates of the upper left corner of the board are (1,1), and of the bottom right are (n,n). The removed fields are not repeated in the file.

Output

The output file `kni.out` should contain one integer (in the first and only line of the file). It should be the maximum number of knights that can be placed on the given chess-board without checking each other.

Example

Input file `kni.in`:

```
3 2

1 1

3 3
```

correct output file `kni.out`

```
5
```

Mars Maps

(Running time - 1s.)

In the year 2051, several Mars expeditions have explored different areas of the red planet and produced maps of these areas. Now, the BaSA (Baltic Space Agency) has an ambitious plan: they would like to produce a map of the whole planet. In order to calculate the necessary effort, they need to know the total size of the area for which maps already exist. It is your task to write a program that calculates this area.

Task

Write a program that:

- reads the description of map shapes from the input file `mar.in`,
- computes the total area covered by the maps,
- writes the result to the output file `mar.out`.

Input

The input file `mar.in` starts with a line containing a single integer N ($1<=N<=10\ 000$), the number of available maps. Each of the following N lines describes a map. Each of these lines contains four integers x_1, y_1, x_2 and y_2 ($0<=x_1<x_2<=30\ 000$, $0<=y_1<y_2<=30\ 000$). The values (x_1,y_1) and (x_2,y_2) are the coordinates of, respectively, the bottom-left and the top-right corner of the mapped area. Each map has rectangular shape, and its sides are parallel to the x- and y-axis of the coordinate system.

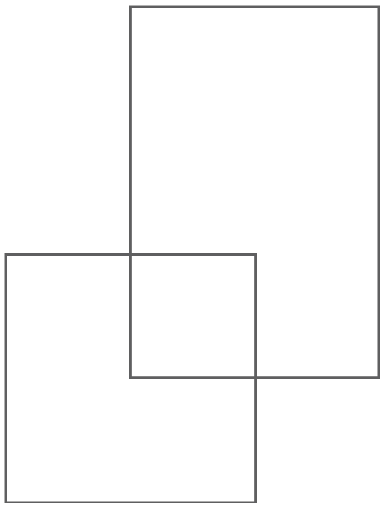
Output

The output file `mar.out` should contain one integer A, the total explored area (i.e. the area of the union of all rectangles).

Example

Input file `mar.in`:

```
2

10 10 20 20

15 15 25 30
```



Correct output file `mar.out`

```
225
```

Teleports

(Running time - 1s.)

Great sorcerer Byter created two islands on the Baltic See: Bornholm and Gotland. On each island he has installed some magical teleports. Each teleport can work in one of two modes:

- receiving--one can be teleported to it,
- sending--anyone who enters the teleport is transfered to the specific destination teleport on the other island, provided that the other teleport is in the receiving mode.

Once, Byter gave his apprentices the following task: they must set the teleports' modes in such a way, that no teleport is useless, i.e. for each teleport set in the receiving mode there must be at least one teleport sending to it, set in the sending mode; and vice versa, for each teleport set in the sending mode, the destination teleport must be set in the receiving mode.

Task

Write a program that:

- reads the description of the teleports on both islands, form the input file `tel.in`,
- determines the appropriate modes for teleports,
- writes the result to the output file `tel.out`.

If there are several possible solutions, your program should output just one of them.

Input

In the first line of the text file `tel.in`, there are two integers *m* and *n*, *1<=m,n<=50 000*, separated by a single space; *m* is the number of teleports on Bornholm, and *n* is the number of teleports on Gotland. Teleports on both islands are numbered from 1 to *m* and *n* respectively. The second line of the file contains *m* positive integers, not greater than *n*, separated by single spaces--*k*-th of these integers is the number of the teleport on Gotland that is the destination of the teleport *k* on Bornholm. The third line contains analogous data for teleports on Gotland, i.e. *n* positive integers, not greater than *m*,separated by single spaces--*k*-th of these integers is the number of the teleport on Bornholm that is the destination of the teleport *k* on Gotland.

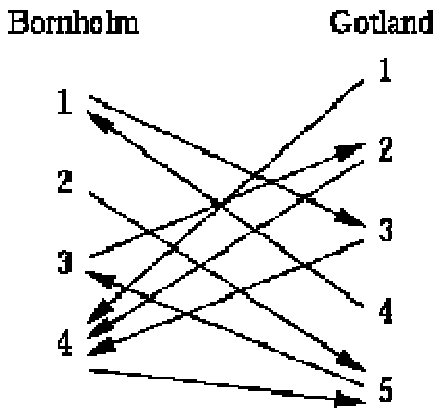
Output

Your program should write two lines describing the modes of the teleports, respectively, on Bornholm and Gotland, to the output file `tel.out`. Both lines should contain a string of, respectively, *m* or *n* ones and/or zeros. If the *k*-th character in the string is `1`, then the *k*-th teleport is in the sending mode, and if it is `0`, than the corresponding teleport is in the receiving mode.

Example

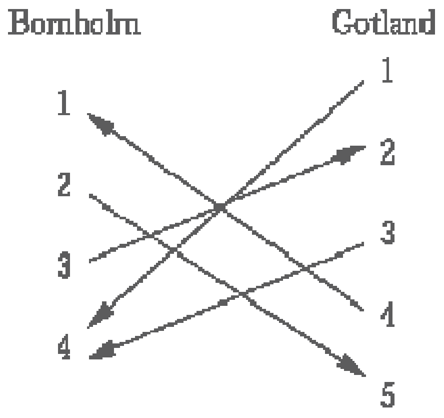
Input file `tel.in`:

```
4 5

3 5 2 5

4 4 4 1 3
```



Correct output file `tel.out`

```
0110

10110
```


Bornholm
Gotland
1
2
3
4
1
2
3
4
5