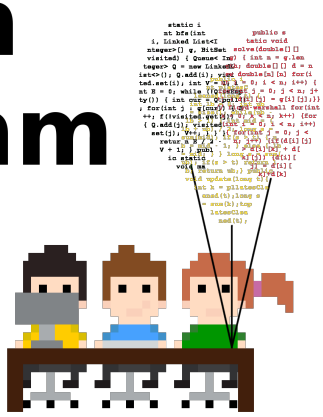
Belgium Algorithm Contest



Belgium Algorithm Contest
Round 2 - 2018

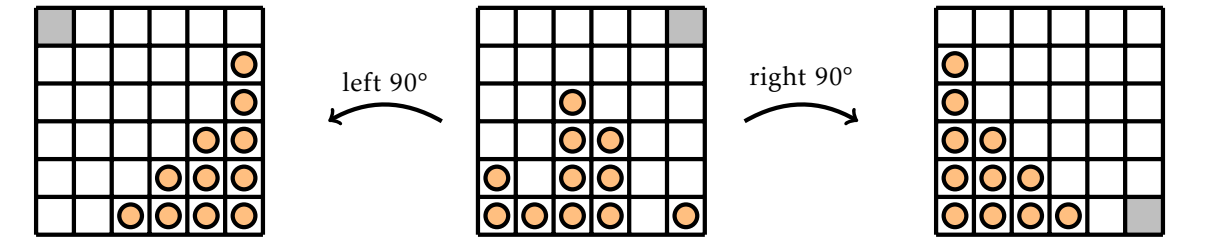
Do not open before the start of the contest.

This page is not really blank.

Belgium Algorithm Contest

Problem A
Gravity
Time limit: 1s

You have a $n \times n$ grid with some stones inside. When you rotate it 90° left or right, the stones will fall down pilling on top of each other. The following figure illustrates the two possible rotations. The gray square is just used for reference, it does not mean anything.



result of the left rotation | original grid | result of the right rotation

You will be given a grid configuration and you have to output the resulting grid after rotating it twice to the right and then twice to the left.

Input

The first line contains a single integer n giving the size of the grid.

Then follow n lines each with a string of length n over the alphabet $\{.,o\}$ giving the grid initial configuration. The .'s represent the empty spaces and the o's represent the stones.

There are no floating stones in the initial configuration, meaning that if there is a stone at some position then, the position bellow it will also contain a stone unless there is no row below it.

Constraints

1. $2 \leq n \leq 30$

Output

Output n lines each with a string of length n over the alphabet $\{.,o\}$ giving the final configuration of the grid after making two rotations to the right followed by two rotations to the

left.

Sample test cases

Input 1

```
2
o .
oo
```

Output 1

```
. o
oo
```

Input 2

```
3
. . o
. oo
ooo
```

Output 2

```
. . o
. oo
ooo
```

Belgium Algorithm Contest

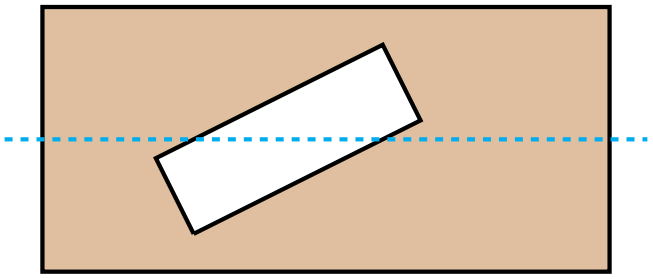
Problem B
Evil cake thief
Time limit: 1s

You just baked the most amazing rectangular cake ever and you planned to eat with Alice and Bob. Unfortunately, before you knew it, Bob sneakily ate a piece of the cake. But that is not the worst of it, he very awkwardly cut his piece shapped as a parallelogram from the middle of the cake! What sane person does such an atrocity...

Now you and Alice are very mad at him and decide to split the remainder of the cake into two pieces of equal area to share between the two of you. To split it, the you will do single cut along a whole straight line in any direction.

Example:

The piece taken by Bob is shown in white and the remaining of the cake in brown. The blue line shows a possible way to split the remainder of the cake in half.



Input

The first line of the input contains two integers w and h giving the width and the height of the cake. Treat the cake as the rectangle with corners $(0,0),(w,0),(w,h)$ and $(0,h)$.

The next line contains eight integers separated by single spaces giving the coordinates of the 4 corners (x_1,y_1), (x_2,y_2), (x_3,y_3) and (x_4,y_4) of the parallelogram shapped piece of cake that Bob took. The coordinates are given in counter-clockwise order. Note that this parallelogram needs not to be aligned with the axis as show in the above example.

You may assume that Bob did not eat the whole cake meaning that the remaining area is non-zero.

Constraints

1. $1 \leq w, h \leq 100$
2. $2 \leq w \cdot h$
3. $0 \leq x_i \leq w$
4. $0 \leq y_i \leq h$

Output

A single line with four numbers separated by single spaces giving two **distinct** points (x_1, y_1) and (x_2, y_2) that belong to a line that separates the remaining cake into two pieces of equal area.

Any correct answer will be accepted. The line defined by the two output points must split the cake into two parts whose areas differ, in absolute value, by at most 10^{-5}.

Sample test cases

Input 1

```
15 7
4 1 10 4 9 6 3 3
```

Output 1

```
0 3.5 15 3.5
```

Input 2

```
4 4
2 0 4 2 2 4 0 2
```

Output 2

```
0 4 4 0
```

Belgium Algorithm Contest

● Problem C
First day on the job
Time limit: 2s

You have just been hired by a taxi company and assigned your first task. There are n customers requesting rides. Each ride consists of the customer origin (x_1, y_1), the customer destination (x_2, y_2) and a time interval $[s, e]$ meaning that the customer wants a taxi ride from (x_1, y_1) to (x_2, y_2), the ride must start any time between s and e and end at most at time e. The **taxi starts at the origin** $(0, 0)$ **at time** 0.

It takes $|x_1 - x_2| + |y_1 - y_2|$ time units for the taxi to travel from position (x_1, y_1) to (x_2, y_2). Because of start times, sometimes the taxi will need to wait at the start position of a ride before starting it.

You already wrote a complicated algorithm to find out in which order you should visit the customers so that you respect all time constraints. After a few hours of waiting, the algorithm finally came through outputting a solution. Yay!

The solution computed by your algorithm consists of an ordering of the rides so that if you go from one ride to the next in that order, waiting at the start of the rides if you arrive too early, then all time constraints will be satisfied. In other words, it is a feasible schedule.

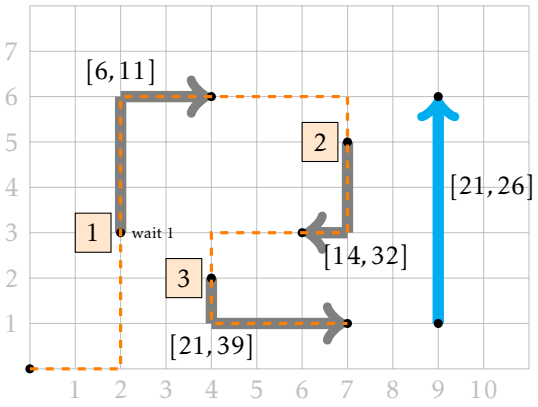
Unfortunately, you did not see that you actually had $n + 1$ customers in total and your solution is incomplete. To avoid getting fired, you are going to try to insert the missing ride on the solution that you already have **without changing the order in which the n other customers are visited**. This means that if ride i occurred before ride j in the original schedule, then it will also occur before j in the new schedule. The starting time of the rides may change in the new solution. But they must all respect the time constraints.

Note that the missing customer can be placed before the first ride, between any two rides or after the last ride.

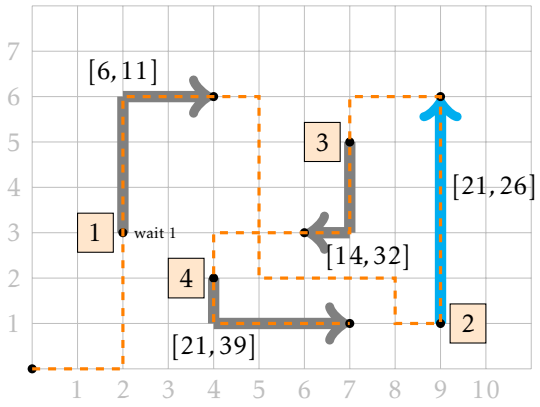
Example:

Consider the following example. The rides given to the algorithm are shown in gray and numbered according to the order in which they should be performed. The new ride to consider is in blue.

On the right we show the only possible way to insert the new ride. Note that the order of the rides in the original solution is preserved.



original solution



new solution, new ride inserted after the first ride

Input

The first line of the input contains a single integer n giving the number of customers given as input to the algorithm.

Then follow n lines each with six integers x_1, y_1, x_2, y_2, s and e as described above. These n lines are provided in the order in which the algorithms tells you to visit them and the order is feasible.

The last line contains the ride that you forgot to consider also described by six integers in the same format as the other rides.

Constraints

1. $1 \leq n \leq 10^5$
2. $1 \leq x_1, y_1, x_2, y_2 \leq 10000$
3. $0 \leq s < e \leq 10^6$

Output

A single line with `possible` if it is possible to insert the missing ride amongst the other ones **without changing the order in which they are given in the input** and satisfying the time constraints. If this is not possible, print `impossible`.

Sample test cases

Input 1

```
3
2 3 4 6 6 11
7 5 6 3 14 32
4 2 7 1 21 39
9 1 9 6 21 26
```

Output 1

```
possible
```

Input 2

```
3
2 3 4 6 6 11
7 5 6 3 14 32
4 2 7 1 21 39
9 1 9 7 21 26
```

Output 2

```
impossible
```

This page is not really blank.

Belgium Algorithm Contest

Problem D
All the way up
Time limit: 2s

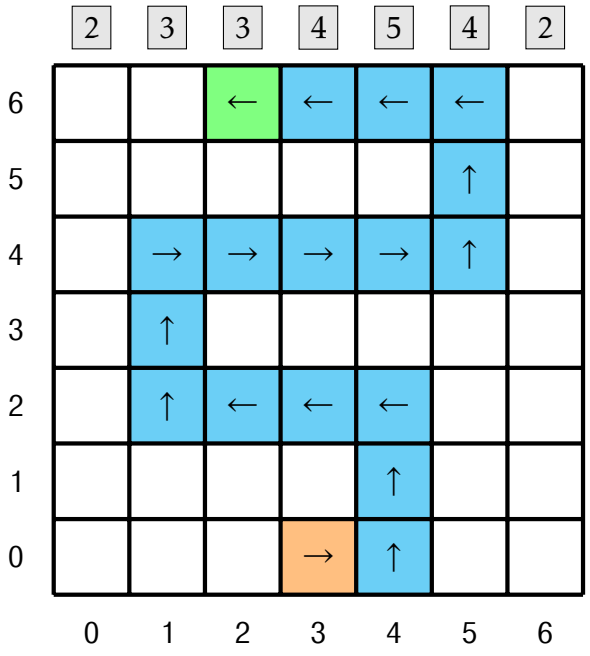
Suppose that you have grid of width w and height h. You are at the bottom row on column s and you want to reach the top row at column e. However, each column has a maximum number of steps that your path can make on it. The number of steps that a path take in a column is defined as the number of cells of that column that the path contains (passing more than once in the same cell counts as several steps on that column).

Example:

Consider the grid a 7 by 7 grid as show in the figure. The numbers of the top show the maximum number of steps that the path can take in each column. The number of the left and bottom are simply the row and column indexes.

The start is show in orange ($s = 3$) and the destination in green ($e = 2$). A possible path is show in color.

This example corresponds to the first sample input.



Input

The first line of the input contains two integers w and h giving the width and the height of the grid, respectively.

Then follows a line with w integers $c_1, \ldots, c_w$ giving the maximum number of steps that the path can take in each column.

The last line contains two integers s and e giving the start and end columns of the path. The path must always go from the bottom row to the top row.

Rows and columns are numbered starting at 0 from left to right and bottom to top.

Constraints

1. $2 \leq h \leq 50000$
2. $1 \leq w \leq 50000$
3. $0 \leq c_i \leq h$
4. $0 \leq s, e < w - 1$

Output

If a path exists, output the coordinates of the path in order. There should be a line for each cell that the path visits and it should contains two integers r and c (0-indexed) giving the row and the column of each position. If there is no solution print `impossible`.

If several solutions exists, any of them will be accepted.

Sample test cases

Input 1

```
16 19
2 2 2 2 2 2 3 2 2 3 1 5 3 2 3 4
12 4
```

Output 1

```
impossible
```

Input 2

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

Output 2

```
0 3
0 4
1 4
2 4
2 3
2 2
2 1
3 1
4 1
4 2
4 3
4 4
4 5
5 5
6 5
6 4
6 3
6 2
```

This page is not really blank.

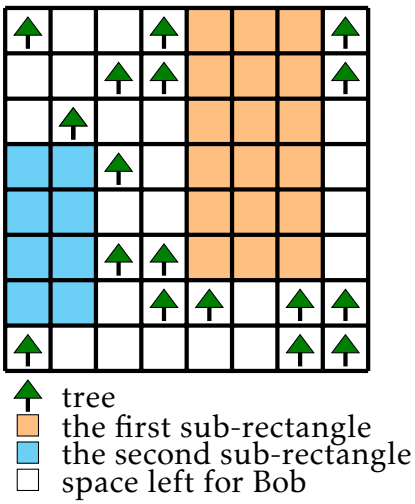
Belgium Algorithm Contest

Problem E
Gimmy land
Time limit: 3s

You and Alice and Bob just inherited a big piece of land. The land is represented by an $n \times m$ grid. Each cell in the land either contains a tree or an empty space. The will specifies that Alice will receive all the cells that contain a tree, as she is a lumberjack. From the empty spaces, you are allowed to select two **non-intersecting** sub-rectangles and Bob will keep the rest. Of course, your sub-rectangles cannot contain trees as those belong to Alice. Since you are an expert in algorithms, it should be an easy task for you to decide what is the maximum area you can get out of the two rectangles that you select.

Example:

The following figure shows a possible configuration of the land together a possible solution that maximizes the total area that you get.



Input

The first line of the input contains two integers n and m giving the dimensions of the land.

Then follow n lines each with a string of length m over the alphabet $\{.,\mathtt{T}\}$. The dots represent empty positions while the others represent the trees.

There are at least two empty positions on the land so that it is always possible to select two pieces of land.

Constraints

1. $2 \leq n, m \leq 500$

Output

A single line with the maximum area that you can get by selecting two disjoint sub-rectangles that do not contains any trees.

Note that the two sub-rectangles that you select cannot intersect but their border can touch.

Sample test cases

Input 1

```
8 8
T..T...T
..TT...T
.T......
..T.....
........
..TT....
...TT.TT
T.....TT
```

Output 1

```
26
```

Input 2

```
4 4
....
....
....
....
```

Output 2

```
16
```

This page is not really blank.