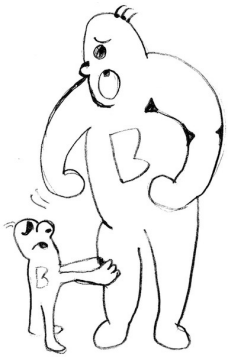
Demarcation



For a long time the island of Bytopia was ruled by the fair king Byteasar. But after the sudden death of the king, his two sons—twins Biteon and Byteon—could not come to an agreement which one of them should ascend the throne. Therefore they decided to divide the island into two provinces to rule them independently.

On a rectangular map Byteotia is shaped as a polygon of N vertices. Every side of the polygon is parallel to a side of the map, and every two consecutive sides are perpendicular to each other. No side of the polygon crosses or touches any other side, except for the common endpoints of consecutive sides.

Biteon and Byteon want to divide the polygon into two congruent polygons, using one line segment contained in the polygon and parallel to a side of the map. (Two polygons are congruent if one can be transformed into the other using a combination of reflections, rotations and translations.) Coordinates of the polygon vertices and the endpoints of the dividing segment are integers.

The king's sons asked you to verify whether such a division is possible.

Task

Given the shape of the island, determine if it can be partitioned by a horizontal or vertical segment into two congruent pieces. If it can, find one such segment.

Input

The first line of the input contains a single integer N, the number of vertices. The ith of the next N lines contains a pair of integers X_i and Y_i ($0 \leq X_i, Y_i \leq 10^9$) which are the coordinates of the ith vertex.

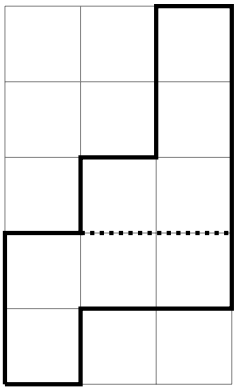
The vertices are given in order, i.e. the line segments $(X_1, Y_1) - (X_2, Y_2)$, $(X_2, Y_2) - (X_3, Y_3)$, $\ldots$, $(X_{N-1}, Y_{N-1}) - (X_N, Y_N)$ and $(X_N, Y_N) - (X_1, Y_1)$ are all sides of the polygon. Furthermore, any two consecutive line segments are perpendicular to each other.

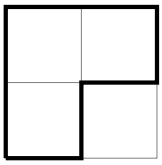
Output

Your program should output a single line. If it is possible to divide the island into congruent parts with a horizontal or vertical segment with endpoints (x_1, y_1) and (x_2, y_2), print 4 integers x_1, y_1, x_2 and y_2, separated by spaces. Either $x_1 = x_2$ or $y_1 = y_2$ must hold. The segment should be contained within the polygon and only its endpoints should touch the boundary.

If a suitable division cannot be found, output a single word `NO`.

Examples

Input	Output	Comments
10 0 0 1 0 1 1 3 1 3 5 2 5 2 3 1 3 1 2 0 2	1 2 3 2	Note that 3 2 1 2 is also a valid solution. 

Input	Output	Comments
6 0 0 1 0 1 1 2 1 2 2 0 2	NO	In this case there is no way to divide the island into two congruent parts. 

Scoring

Subtask 1 (12 points). $4 \leq N \leq 100\,000$. Any horizontal or vertical line that divides the polygon divides it into exactly two parts.

Subtask 2 (15 points). $4 \leq N \leq 200$.

Subtask 3 (23 points). $4 \leq N \leq 2\,000$.

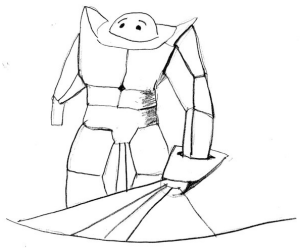
Subtask 4 (50 points). $4 \leq N \leq 100\,000$.

Constraints

Time limit: 0.5 s.

Memory limit: 256 MB.

Portals



There is a cake placed in a labyrinth and you desperately want to eat it. You have a map of the labyrinth, which is a grid of R rows and C columns. Each grid cell contains one of the following characters:

- `#` (number sign) which denotes a wall block,
- `.` (dot) which denotes an open square,
- `S` (uppercase letter s) which denotes an open square of your current location,
- `C` (uppercase letter c) which denotes an open square with the cake.

You may only walk on the open squares and move from one open square to another if they share a side. Additionally, the rectangular area depicted on the map is completely surrounded by wall blocks.

In order to reach the cake faster you have acquired a portal gun from Aperture Science™, which operates as follows. At any time it can fire a portal in one of the four directions *up*, *left*, *down* and *right*. When a portal is fired in some direction, it will fly in that direction until it reaches the first wall. When this happens, a portal will be spawned on the wall block, on the side that faces you.

At most two portals can exist at any given time. If two portals are already placed in the labyrinth, then one of them (selected by you) will be removed immediately upon using the portal gun again. Firing a portal at an existing portal will replace it (there may be at most one portal per side of wall block). Note that there may be two portals placed on different sides of the same wall block.

Once two portals are placed in the labyrinth you can use them to teleport yourself. When standing next to one of the portals, you can walk into it and end up at the open square next to the other portal. Doing this takes as much time as moving between two adjacent squares.

You may assume that firing portals does not take time and moving between two adjacent squares or teleporting through portals takes one unit of time.

Task

Given the map of the labyrinth together with your starting location and the location of the cake, calculate the minimum possible time needed for you to reach the cake.

Input

The first line of the input contains two integers: the number of rows in the map R, and the number of columns C. The next R lines describe the map. Each of these lines contains C characters: `#`, `.`, `S` or `C` (whose meaning is described above).

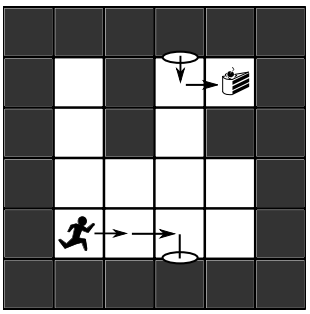
It is guaranteed that characters `S` and `C` each appear exactly once in the map.

Output

The output should contain a single integer — the minimum time that is needed to reach the cake from the starting location.

You may assume that it is possible to reach the cake from your starting location.

Example

Input	Output	Comments
4 4 .#.C .#.# .... S...	4	One quickest sequence of moves is as follows: 1) move right, 2) move right, shoot one portal up, and one portal down, 3) move through the bottom portal, 4) move one square right and reach the cake. 

Scoring

Subtask 1 (11 points): $1 \leq R \leq 10, 1 \leq C \leq 10$.

Subtask 2 (20 points): $1 \leq R \leq 50, 1 \leq C \leq 50$.

Subtask 3 (20 points): $1 \leq R \leq 200, 1 \leq C \leq 200$. Every open square has at least one wall block adjacent to it.

Subtask 4 (19 points): $1 \leq R \leq 200, 1 \leq C \leq 200$.

Subtask 5 (30 points): $1 \leq R \leq 1000, 1 \leq C \leq 1000$.

Constraints

Time limit: 1 s.

Memory limit: 256 MB.

Senior Postmen

It is the year 2036 and Europe is crowded by senior citizens. In order to keep them healthy, the European ministry for majority groups (seniors *are* a majority!) suggests to have them deliver the small amount of paper mail that is still being sent — typically to seniors. This suggestion is going to be implemented all over Europe.

The ministry has devised a "senior postmen system" in the following way: Europe has been divided into mail districts. A mail district has a street network of streets and junctions. Every street in the network can be walked in both directions. In each district, arbitrarily many senior citizens are available to be hired as mailman. Every morning, each mailman receives a bag with mail to be delivered on a tour that covers a part of the street network. Every tour must be senior–compatible, i.e. it must satisfy the following conditions:

- It starts and ends at the same junction.
- It never passes a junction more than once. (The seniors shall not be confused.)
- It must not have a street in common with any other tour; hence, any street in the district is to be served by exactly one mailman. (The seniors shall not fight with each other.)

Together, the tours must cover the given network: each street in the network must be part of exactly one tour.

Task

The ministry now needs a software that, for a given mail district's street network, will compute a set of senior–compatible tours that covers the network.

Input

The input describes the street network.

The first input line contains two integers N and M. N is the number of junctions, and M is the number of streets. Junctions are numbered from 1 to N.

Each of the following M lines contains two integers u and v ($1 \leq u, v \leq N, u \neq v$), meaning that there is a street connecting junctions u and v.

For any input holds:

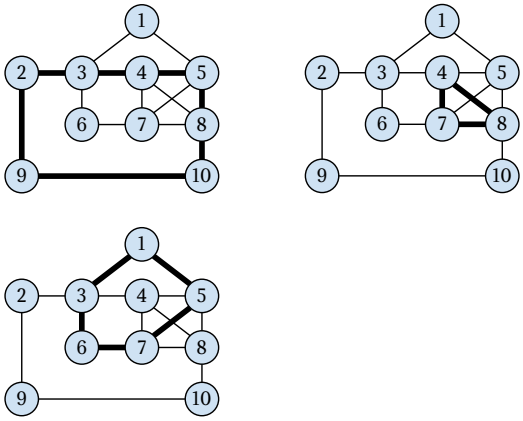
1. Any two junctions can be connected by no more than one street.
2. You can reach any junction from any other by traveling along one or more streets.
3. There is a solution, i.e. a set of senior–compatible tours can be computed that cover the network.

Output

Each line of the output should correspond to one senior–compatible tour, and should list the numbers of the junctions in that tour. Junction numbers must be output in the order the junctions are passed by the mailman, with the starting (and ending) junction being output first (and only once).

If more than one solution exists, your program may output any one of them.

Example

Input	Output	Comments
10 15 1 3 5 1 2 3 9 2 3 4 6 3 4 5 7 4 4 8 5 7 8 5 6 7 7 8 8 10 10 9	2 3 4 5 8 10 9 7 8 4 1 5 7 6 3	The following picture illustrates the street network and the three senior–compatible tours that may be used to cover it.  Note that there are several solutions to this example, among them some with only two tours.

Scoring

Subtask 1 (38 points): $3 \leq N \leq 2\,000$, $3 \leq M \leq 100\,000$.

Subtask 2 (17 points): $3 \leq N \leq 100\,000$, $3 \leq M \leq 100\,000$.

Subtask 3 (45 points): $3 \leq N \leq 500\,000$, $3 \leq M \leq 500\,000$.

Constraints

Time limit: 0.5 s.

Memory limit: 256 MB.