Japanese Olympiad in Informatics Open Contest

(JOI Ninja Contest)

Monday, September 17th, 2012

Hosted by

The Japanese Committee for International Olympiad in Informatics (JCIOI)

Task name	**Code**	**Jumps**	**Mansion**
Time Limit	5.0 sec	2.0 sec	2.0 sec
Memory Limit	256 MB	256 MB	256 MB
Points	100	100	100
Input	stdin (keyboard)		
Output	stdout (screen)		

Language	**Compiler version**	**Compiler options**
C	gcc version 4.6.3	-m64 -O2 -lm
C++	g++ version 4.6.3	-m64 -O2 -lm
Pascal	fpc version 2.4.4	-O2 -Sd -Sh

Code

It is said that, when a ninja sent a message to his/her colleague, he/she used a secret code so that the message could not be read by the enemy's ninjas. Ninjas were using various types of secret codes. There is a secret code which uses a special coding sheet.

A grid of squares is written on the coding sheet. It consists of M rows and N columns. In each of MN squares, a character is written. The square in the a-th row from above and b-th column from left is denoted by (a, b).

The message is encoded as a shortest path from the upper left square $(1, 1)$ to the lower right square (M, N). A shortest path is a sequence of squares from the starting point to the destination obtained by moving only to the right or down. The message encoded as a shorest path is the string of $M + N - 1$ characters written in the squares on the path in order.

Among their colleagues, ninjas choose a shortest path beforehand. They can send messages to their collesgues using various coding sheets. Since the enemy's ninjas do not know the chosen shortest path, the message cannot be read by them even if the coding sheet was stolen.

However, there is a problem about the coding sheets; the total number of shortest paths on the grid is so big that the chosen shortest path and another shortest path may represent the same message. In some arrangement of characters on the grid, the number of shorest paths representing the same message is large, and the message is more likely to be read by the enemy's ninjas.

Write a program that, given the arrangement of characters on the grid, measures the strength of the coding sheet. Here, the strength of the coding sheet is measured as follows.

- One starts from the upper left square $(1, 1)$. One chooses to move to the right or down with equal probability. If one is in the bottom row of the grid, one moves to the right with probability 1. If one is in the rightmost column of the grid, one moves down with probability 1. One repeats this until it gets to the lower right square (M, N), and obtains a shortest path (a sequence of squares) s.
- Let $f(s)$ be the number of possible shortest paths on the grid with the same message as the path s. Note that $f(s)$ is at least 1 because the path s itself is considered.
- Let E be the expected value of $f(s)$ for any s. In other words, E is the sum of the product of $f(t)$ and the probability that one takes the path t for each possible shortest path t. We consider the value of E as the strength of the coding sheet.

Since E might not be an integer, we want to know $E \times 2^{M+N}$ instead of E. However, since this value might be too big, we need only to know the value of the remainder when $E \times 2^{M+N}$ is divided by 1 000 000 007.

Task

Write a program that, given the size of the grid on the coding sheet and the arrangement of characters on it, calculates the remainder when $E \times 2^{M+N}$ is divided by 1 000 000 007.

Constraints

$1 \leqq M \leqq 300$	The number of rows in the grid
$1 \leqq N \leqq 300$	The number of columns in the grid

Input

Read the following data from the standard input.

- The first line of input contains two space separated integers M, N, where M is the number of rows in the grid and N is the number of columns in the grid.
- The following M lines contain the arrangement of characters on the grid. The $(1+i)$-th line $(1 \leqq i \leqq M)$ contains a string of N characters in capital letters. The j-th character $(1 \leqq j \leqq N)$ of this string is the character written in the square (i, j) in the grid.

Output

Write the remainder when $E \times 2^{M+N}$ is divided by 1 000 000 007 to the standard output.

Grading

In test cases worth 20% of the full score, $M \leqq 10$ and $N \leqq 10$.

Sample Input and Output

Sample Input 1	Sample Output 1

`2 3`
`JOI`
`III`
	`48`

In this example, all possible shortest paths, the messages represented by them, and the probability one takes each of them are as follows.

- If one moves to right, right, down in this order, one gets the message `JOII`. The probability that one takes this path is $\frac{1}{4}$.
- If one moves to right, down, right in this order, one gets the message `JOII`. The probability that one takes this path is $\frac{1}{4}$.
- If one moves to down, right, right in this order, one gets the message `JIII`. The probability that one takes this path is $\frac{1}{2}$.

Since $E = 1.5$, one should output $E \times 2^{M+N} = 48$.

Sample Input 2	Sample Output 2
4 4 AAAA AAAA AAAA AAAA	5120

In this example, all shortest paths represent the same message `AAAAAAA`. Therefore, for any shortest path t, $f(t) = 20$. Hence $E = 20$, and one should output $E \times 2^{M+N} = 5120$.

Jumps

Jumping techniques are essential to ninjas. A group of ninjas will have a training of jumps on a large pond.

There are N rocks in the pond, numbered from 1 to N. The position of a rock can be considered as a point in the 2-dimensional cartesian plane. The coordinate of the rock i ($1 \leqq i \leqq N$) is (X_i, Y_i). They want to find a route to visit all of the N rocks by jumping from one rock to another and to return to the starting point after visiting all of the N rocks. They want to visit each of the N rocks only once. A jump from one rock to another is the straight line segment connecting them. Moreover, for safety reasons, their route should not have self-intersection. In other words, if we look down the pond from the sky, the route should not pass the same place more than once.

Task

Write a program that, given the position of N rocks, finds a route satisfying the conditions.

Constraints

$3 \leqq N \leqq 100\,000$	The number of rocks
$0 \leqq X_i \leqq 1\,000\,000\,000$	The X-coordinate of the rock i
$0 \leqq Y_i \leqq 1\,000\,000\,000$	The Y-coordinate of the rock i

Input

Read the following data from the standard input.

- The first line of input contains an integer N.
- The $(1 + i)$-th line ($1 \leqq i \leqq N$) contains two space separated integers X_i, Y_i, indicating that the coordinates of rock i are (X_i, Y_i). There is no pair of rocks that share the same place.

Output

If there exists a route satisfying the conditions, write it to the standard output. The output should consist of N lines, and the j-th line ($1 \leqq j \leqq N$) should contain the number of the j-th rock in the route. If several routes satisfy the conditions, the program may output any one of them.
If there is no route satisfying the conditions, the output should consist of one line containing '`0`'.

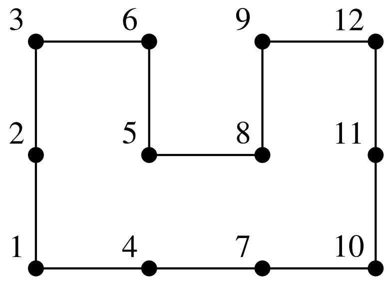
Grading

In test cases worth 10% of the full score, $N \leqq 8$.
In test cases worth 20% of the full score, $N \leqq 16$.
In test cases worth 50% of the full score, $N \leqq 1\,000$.

Sample Input and Output

Sample Input 1	Sample Output 1
12 0 0 0 10 0 20 10 0 10 10 10 20 20 0 20 10 20 20 30 0 30 10 30 20	9 12 11 10 7 4 1 2 3 6 5 8

The figure corresponding to this example is as follows. Note that there is more than one route satisfying the conditions.



Sample Input 2	Sample Output 2
3 23 7 91 27 40 12	0

In this example, there is no route satisfying the conditions. Therefore, the program should output '0'.

Mansion

You are a ninja who will sneak into a huge mansion. The mansion consists of square halls. All halls are placed regularly to the north, south, east, and west. Namely, halls are arranged in a square grid. All halls have exactly the same shape.

Although you are a well-trained ninja, you can only take limited routes to move because the mansion is heavily secured. In each hall, there is a door in the center of each of four walls which connects two adjacent halls. There are N safe places in each hall, including the positions of four doors, where you can hide safely. In each hall, there are M safe routes to move between safe places in a hall. When you move in a hall, you can only take safe routes. We know how long it takes to take each of M routes.

Now, you are in the safe place V in a hall, which is not a door. Your destination is the safe place W, which is not a door, in a hall located X halls to the east and Y halls to the north from your current hall. You want to reach your destination in the shortest possible time.

You should not count the amount of time needed to pass through a door to move to an adjacent hall. Also, you should not count the amount of time to hide in a safe place. The mansion is sufficiently large, and you will not get out of it.

Task

Write a program, given the information of the mansion, which determines whether it is possible to reach your destination, and, if it is possible, outputs the shortest possible time to the destination.

Constraints

$5 \leqq N \leqq 100\,000$	The number of safe places in a hall
$1 \leqq M \leqq 200\,000$	The number of safe routes between safe places in a hall
$1 \leqq T_i \leqq 1\,000$	The amount of time to take the i-th route
$5 \leqq V \leqq N$	Your current position
$\lvert X \rvert \leqq 1\,000\,000\,000$	The position of the hall of your destination
$\lvert Y \rvert \leqq 1\,000\,000\,000$	The position of the hall of your destination
$5 \leqq W \leqq N$	Your destination

Input

Read the following data from the standard input.

- The first line of input contains six space separated integers N, M, V, X, Y, W, where N is the number of safe places in each hall, M is the number of safe routes between safe places, V is your current position, and X, Y, W describe your destination. Safe places in a hall are numbered from 1 to N. Doors of the east, north, west, south walls are numbered $1, 2, 3, 4$.
- The following M lines contain the information of safe routes. The $(1 + i)$-th line $(1 \leqq i \leqq M)$ contains three space separated integers A_i, B_i, T_i. This means, in each hall, you can move between the safe place A_i and the

safe place B_i in both directions by taking a safe route, and it takes T_i seconds to move. For any i, $A_i \neq B_i$. If $i \neq j$, $(A_i, B_i) \neq (A_j, B_j)$ and $(A_i, B_i) \neq (B_j, A_j)$.

Output

Write the shortest possible time to reach your destination to the standard output. If it is impossible to reach your destination, write '-1'.

Grading

In test cases worth 30% of the full score, $|X| \leqq 3$ and $|Y| \leqq 3$.

Sample Input and Output

Sample Input 1	Sample Output 1
7 9 5 1 0 6 1 2 1 1 5 4 2 3 2 3 6 5 4 5 2 4 6 3 4 7 1 5 6 2 6 7 1	7

Sample Input 2	Sample Output 2
5 3 5 3 3 5 1 5 1 2 4 2 3 5 1	-1