Problem K
Krazy Taxi
Time limit: 1 second

There have been several complaints about a taxi that has been driving the wrong direction down one-way streets. Jennifer, a police officer, is responding to the complaints and has pulls the suspected taxi over.

After questioning the driver, Jennifer has determined that the driver started his night at his house and has travelled to the current location, but he claims that he has not done anything illegal. Jennifer goes back to her car and looks at the map:

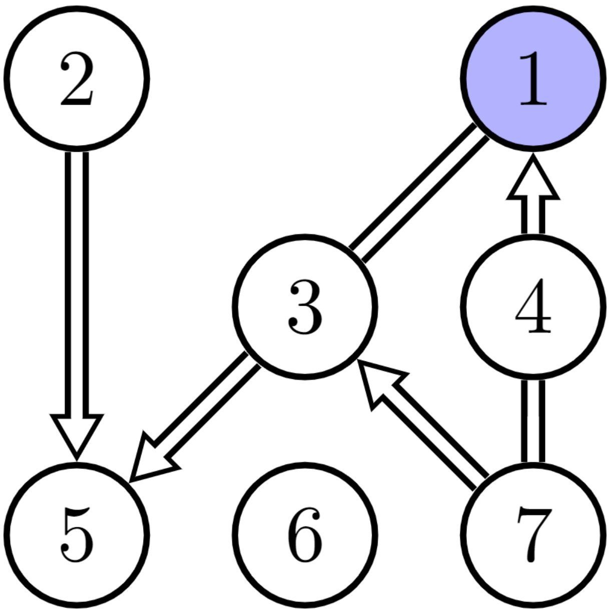

Figure K.1: Edges with arrows depict one-way streets. Edges without arrows depict two-way streets.

The driver's house is labeled as 1 on the map. Since they are currently on location 4, Jennifer can be sure that the driver went the wrong way down a one-way street (either 1-to-4 or 3-to-7). So Jennifer gives the driver a ticket. The next night, Jennifer pulls over the same driver at location 5. Since it is possible to get to location 5 via two-way streets and one-way streets in the correct direction, she cannot give him a ticket.

Since this is going to happen again and again, help Jennifer by writing a program that shows if she should give a ticket or if it is impossible to get to a location even by using one-way roads in the wrong direction from the driver's house.

Input

The first line contains three integers n ($1 \leq n \leq 1\,000$), which is the number of locations, x ($0 \leq x \leq 1\,000$), which is the number of two-way streets, and y ($0 \leq y \leq 1\,000$), which is the number of one-way streets.

The next x lines describe the two-way streets. Each of these lines contains two integers u ($1 \leq u \leq n$) and v ($1 \leq v \leq n$), which are the two end-points of the street.

The next y lines describe the one-way streets. Each of these lines contains two integers u ($1 \leq u \leq n$) and v ($1 \leq v \leq n$), which are the two end-points of the streets. The street goes from u to v.

It is possible that there are multiple streets between a pair of locations, but all streets connect two distinct locations. The locations are numbered 1 to n and the driver's house is location 1.

Output

For each of the locations in ascending order in order display whether or not Jennifer should give a ticket or if it is impossible to get there.

Sample Input 1	Sample Output 1

5 0 2
3 2
4 1
	No Ticket
	Impossible
	Impossible
	Ticket
	Impossible

Sample Input 2	Sample Output 2

7 2 4
1 3
4 7
2 5
3 5
4 1
7 3
	No Ticket
	Ticket
	No Ticket
	Ticket
	No Ticket
	Impossible
	Ticket

Sample Input 3	Sample Output 3
2 0 0	No Ticket Impossible

acm icpc International Collegiate Programming Contest

icpc.foundation

icpc global programming tools sponsor

Problem K
Keeping Cool
Time limit: 2 seconds

Source: Pexels

Kevin has just gotten back to his car after a morning at the beach and is about to drive away when he realises that he has left his ball somewhere. Thankfully, he remembers exactly where it is! Unfortunately for Kevin, it is extremely hot outside and any sand that is exposed to direct sunlight is very hot. Kevin's pain tolerance allows him to only run for at most k seconds in the hot sand at one time. Kevin runs at exactly 1 metre per second on hot sand.

Scattered around the beach are umbrellas. Each umbrella is a perfect circle and keeps the sand underneath it cool. Each time Kevin reaches an umbrella, he will wait there until his feet cool down enough to run for another k seconds on the hot sand. Note that Kevin will not run more than k seconds in the hot sand at one time, so if two umbrellas are more than k metres apart, Kevin will not run between them.

Determine the minimum amount of time that Kevin must be in the sun in order to retrieve his ball and return back to the car.

Input

The first line of input contains four integers n $(0 \leq n \leq 100)$, which is the number of umbrellas, k $(1 \leq k \leq 100)$, which is the number of metres that Kevin can run on the hot sand, x $(-100 \leq x \leq 100)$ and y $(-100 \leq y \leq 100)$, which are the coordinates of the beach ball. Kevin starts at his car at $(0, 0)$. You may treat Kevin and the ball as single points.

The next n lines describe the umbrellas. Each of these lines contains three integers x $(-100 \leq x \leq 100)$, y $(-100 \leq y \leq 100)$ and r $(1 \leq r \leq 100)$. The umbrella is a circle centred at (x, y) with radius r.

There may be multiple items (ball, umbrella(s) or Kevin) at a single location. All measurements are in metres.

Output

Display the minimum amount of time (in seconds) that Kevin must be in the sun. If it is impossible for Kevin to get to the ball and return back to the car, display `-1` instead. Your answer should have an absolute or relative error of less than 10^{-6}.

Sample Input 1	Sample Output 1
`0 1 0 0`	`0.0000000000`

Sample Input 2	Sample Output 2
`0 20 1 2`	`4.4721359550`

Sample Input 3	Sample Output 3
`0 10 20 20`	`-1`

Sample Input 4

```
2 2 7 4
6 2 2
2 2 1
```

Sample Output 4

```
6.1289902045
```

Sample Input 5

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

Sample Output 5

```
4.0000000000
```

Problem E
Election Frenzy
Time limit: 10 seconds

Source: Reboot Wiki

Checks and balances are some of the most important parts of any democratic government system. After a long campaign, the election between Phong, from the Sprites Party, and Megabyte, from the Viruses Party, has just finished. The only thing left to do is count the votes and declare a winner.

The votes are counted in a room with t tables. At each table, there is a person counting votes. These people are called *counters*. Members from the two political parties are allowed to be in the room to ensure that the counting is done fairly. These people are called *scrutineers*. In a perfect system, one scrutineer from each political party would be present at each table. Unfortunately, this is not possible since there is only enough room for one scrutineer per table.

We need your help in deciding how to assign the scrutineers to the tables. Each table must have exactly one scrutineer: from either the Sprites or the Viruses. Some of the tables are close to one another. This means that a scrutineer will be able to monitor the counting at their table and all surrounding tables. An assignment of scrutineers is considered fair if every table is monitored by at least one Sprite and at least one Virus.

The counter at each table was asked to submit a list of all tables that can be seen from their table. For some reason, some of the counters submitted a list of the tables that *can* be seen from their table and some of the counters submitted a list of tables that *cannot* be seen from their table.

Given this information, find a fair assignments of scrutineers.

Input

The first line of input contains a single integer t ($1 \leq t \leq 200\,000$), which is the number of tables.

The next t lines describe the lists that the counters submitted for each table. Each of these lines starts with a letter p (either `C` or `N`), which is the type of the list, and an integer k ($0 \leq k \leq t-1$), which is the length of the list. This is followed by k distinct integers $a_1, \ldots, a_k$ ($1 \leq a_i \leq n$), which are the items on the list. If $p =$ `C`, then each table a_i can be monitored by the scrutineer at this table and all other tables cannot be monitored. If $p =$ `N`, then each table a_i cannot be monitored by the scrutineer at this table and all other tables can be monitored. The list does not contain its own table number since the scrutineer can always monitor the table they are sitting at.

The first table in the input is table 1, the second table is table 2, and so on. It is guaranteed that if table x can monitor table y, then table y can monitor table x. The total length of all lists is at most $500\,000$.

Output

Display any fair assignment of scrutineers. The assignment should be described as a string of length t containing only the letters `S`, for a Sprite scrutineer, and `V`, for a Virus scrutineer. The first letter of the string is the scrutineer for table 1, the second letter is the scrutineer for table 2, and so on. If there are multiple solutions, display any of them. If there is no such assignment, display `Impossible` instead.

Sample Input 1

```
5
C 3 3 4 5
C 3 3 4 5
C 2 1 2
C 2 1 2
C 2 1 2
```

Sample Output 1

```
SSVVV
```

Sample Input 2

```
4
C 2 2 4
N 1 4
N 2 1 4
C 1 1
```

Sample Output 2

```
SSVV
```

Sample Input 3

```
1
N 0
```

Sample Output 3

```
Impossible
```