# Problem 1. How Many Soldiers

(Time Limit: 2 seconds)

## Problem Description

There are n cities and m roads. One road connected two cities, and the road can be controlled by deploying a soldier at one of two cities. Now the question is to find the minimum number of soldiers such that all the roads can be controlled.

## Input Format

The first line contains two integers *m* and *n*, in which *m* is the number of roads and *n* is the number of cities. The cities are labeled from 0 to *n*-1, and each road is given by its two endpoints line by line. For example, the sample input contains 6 roads with 5 cites. The six roads are (0,1), (1,2), (1,3), (1,4), (2,3), and (3,4). You can assumed that $1 < n < 51$, $m<3n/2$, and each city is connected by at least one road.

## Output Format

Output the minimum number of soldiers in one line. For the example, we only need two soldiers putting at cities 1 and 3.

## Example

| **Sample Input:** | **Sample Output:** |
|---|---|
| 6 5<br>0 1<br>1 2<br>1 3<br>1 4<br>2 3<br>3 4 | 2 |

## Problem 2. Fifth Avenue

(Time Limit: 3 seconds)

### Problem Description

Nowadays, telecommunication companies set up base stations everywhere, so that people can use high-speed networks by connecting their devices to the base stations.

Fifth Avenue is a long, straight street, along which there are $n$ base stations $B_1$, $B_2$, …, $B_n$. Fifth Avenue is a good place for window shopping. The city government sets up $m$ free devices $D_1$, $D_2$, ..., $D_m$ along the street, so that tourists can access to networks by using the devices. Each base station $B_i$ has a *radius* $r_i$. That is, a device $D_j$ can connect to a base station $B_i$ if and only if their distance is at most $r_i$. A base station can serve any number of devices. In order to get the best quality, a device $D_j$ is always connected to the closest station which can serve $D_j$. Consider the example in Figure 1. There are $n = 4$ base stations $B_1$, $B_2$, $B_3$, $B_4$ located, respectively, at 1, 3, 4, 11; and there are two devices $D_1$ and $D_2$ located, respectively, at 7 and 9. Let $r_1 = 9$, $r_2 = 10$, $r_3 = 1$, and $r_4 = 3$. The device $D_1$ can be served by $B_1$ and $B_2$. Since $B_2$ is nearer, $D_1$ is connected to $B_2$. The device $D_2$ is connected to $B_4$. The total distance from the devices $D_1$ and $D_2$ to the base stations serving them is 4 + 2 = 6.

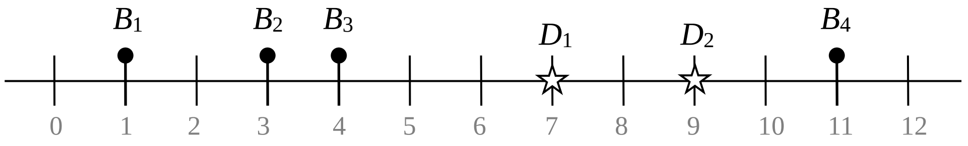

**Figure 1.** Four base stations at 1, 3, 4, 11 and two devices at 7 and 9.

Given locations of the base stations $B_1$, $B_2$, ..., $B_n$ and the devices $D_1$, $D_2$, …, $D_m$, please compute the total distance from the devices $D_1$, $D_2$, ..., $D_m$ to the base stations serving them.

## Technical Specification

- The number of base stations $n \leq 10^5$.
- The number of devices $m \leq 10^5$.
- The location of each base station or device is a non-negative integer $\leq 10^7$.
- The radius of each base station is a positive integer $\leq 10^7$.

## Input Format

The first line contains an integer $T \leq 10$ indicating the number of test cases. The first line of each test case contains two integers $n$ and $m$, $1 \leq n, m \leq 10^5$, indicating the numbers of base stations and devices, respectively. There are $n$ pairs of integers in the following $n$ lines, where the pair of integers $b_i$ and $r_i$ in $i$th line represents the location and radius of the $i$th base station. The given $b_i$ and $r_i$ satisfy the following: $0 \leq b_1 \leq b_2 \leq \ldots \leq b_n \leq 10^7$ and $1 \leq r_i \leq 10^7$. Finally, there are $m$ integers $d_1, d_2, \ldots, d_m$ in a line, where $d_j$ is the location of the $j$th device and $0 \leq d_1 \leq d_2 \leq \ldots \leq d_m \leq 10^7$. It's guaranteed that every device can be served by at least one base station.

## Output Format

For each test case, output in a line the total distance from the devices $D_1, D_2, ..., D_m$ to the base stations serving them.

## Example

| **Sample Input:** | **Sample Output:** |
|---|---|
| 2<br>4 2<br>1 9<br>3 10<br>4 1<br>11 3<br>7 9<br>3 2<br>10 40<br>20 1<br>30 7<br>20 22 | 6<br>12 |

# Problem 3. Lucky Palindrome

(Time Limit: 1 seconds)

## Problem Description

We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits 4 and 7. For example, numbers 4, 47, 744 are lucky and 5, 14, 470 are not.

We all also know that a palindrome is a string that reads the same forward and backward. For example, “noon”, “tmt” and “a” are all palindromes, while “test” and “tmt514” are not.

Thus we can define lucky palindrome numbers! We say a number is a lucky palindrome number if it is not only a lucky number, but also a palindrome in its decimal representation. For example, 4, 44, 747, 777 are all lucky palindrome numbers, while 47, 514, 50216 are not.

Karen is a big fan of lucky palindrome numbers, now he want to know how many lucky numbers within range $[L, R]$, can you help him?

## Technical Specification

- $1 \leq L \leq R \leq 10^{18}$
- The number of test case $T \leq 10000$

## Input Format

The first line contains an integer $T$ indicating the number of the test cases. For each test case, there are two integers $L, R$ in one line.

## Output Format

For each test case, output the number of lucky palindrome numbers within range $[L, R]$.

## Example

| Sample Input: | Sample Output: |
|---|---|
| 4<br>4 7<br>5 14<br>50 216<br>1 1000 | 2<br>1<br>1<br>8 |

# Problem 4. Detect the First Rectangle

(Time Limit: 1 seconds)

## Problem Description

We have a sequence of $N$ axis-parallel line segments (i.e. horizontal or vertical), and they are added to an infinite board one by one. We want to detect if there are two horizontal lines $H_1$ and $H_2$ and two vertical lines $V_1$ and $V_2$ such that each horizontal line intersects with each vertical line, so that their sub-segments will form a rectangle. If it is possible, we want to output the minimum integer $k$, such that the first $k$ line segments contain such four line segments $H_1$, $H_2$, $V_1$, $V_2$. Otherwise, we output the string "none" instead.

Note that, in this problem, we do not consider a rectangle formed by more than four input lines. That is, you should not consider the case that two horizontal/vertical lines are joined into one line.

## Technical Specification

- $0 < x_1, y_1, x_2, y_2 < 1000000$.
- $0 \leqq N \leqq 200$.
- Each segment has length at least one.

## Input Format

The first line is an integer $T$ ($0 < T < 15$) which indicates the number of test cases.

The first line of each test contains an integer $N$. In the following N lines, each line contains four integers $x_1$, $y_1$, $x_2$, $y_2$, separated by space, which denote the two ending points of line segment. Precisely, the four integers mean that we have a line segment from point ($x_1$, $y_1$) to point ($x_2$, $y_2$).

## Output Format

For each test case, outputs in one line the minimum number $k$ such that a rectangle is formed by four of the first $k$ segments. If no rectangle is formed by any four segments of all $N$ segments, output "none" instead.

## Example

| **Sample Input:** | **Sample Output:** |
|---|---|
| 2 | 10 |
| 10 | none |
| 2 13 2 4 | |
| 5 2 5 12 | |
| 1 5 10 5 | |
| 4 7 13 7 | |
| 12 12 12 3 | |
| 10 11 3 11 | |
| 7 9 11 9 | |
| 8 12 8 8 | |
| 9 10 9 6 | |
| 6 6 6 12 | |
| 7 | |
| 1 13 13 13 | |
| 2 1 2 9 | |
| 3 1 3 9 | |
| 4 9 4 1 | |
| 5 9 5 1 | |
| 6 1 6 9 | |
| 7 3 1 3 | |

# Problem 5. Traveling in Home Town

(Time Limit: 3 seconds)

## Problem Description

This weekend, John's cousins will come to his home town. Since there're multiple beautiful places in this town, he wants to invite them to visit ALL these wonderful scenery by driving his car.

John finds out a map of this town, and he realizes that he can drive in four directions: east, west, north and south. Driving one unit distance costs one unit of gas. Initially, the tank of his car is FULL of gas. Whenever his car is out of gas, he cannot move anymore.

Luckily, there are many gas stations in the town, so John can fill up his tank to continue his tour. Note that John is stingy, thus he will only fill up his tank ONCE for any gas station. Of course he can choose to bypass any gas station without fill up the tank when passing through it. There are also some deep valleys in this town where he cannot drive through them.

Unfortunately, his car broke down yesterday, and he needs to rent a car instead. Because it is more expensive to rent a car with larger fuel tank than the smaller one, John wants to know the minimum size of tank needed in order to complete this tour, starting from his home.

For example, the minimum size of tank is 3 in the following map, shown as Figure 1, where 'H' denotes his home, 'V' for valleys, 'G' for gas stations, 'S' for the scenery to visit, and '.' for the plane. John can drive to any of these places except for the valleys. One of tour for Figure 1 with tank size 3 is shown in Figure 2. Note that John does not need to return to his home.

| | | | |
|---|---|---|---|
| G | . | . | . |
| S | . | V | S |
| . | H | V | S |
| G | . | G | . |

Figure 1

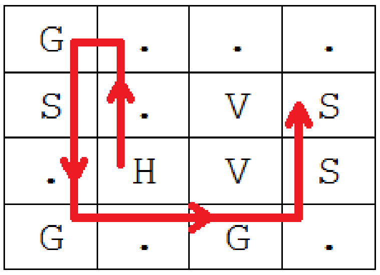

Figure 2

## Technical Specification

- Number of test cases $T$, $1 \le T \le 10$
- Row $R$ and Column $C$ of the map, $1 \le R, C \le 15$
- Total number of gas station $N_G$ and scenery $N_S$ is less than 15, i.e., $0 \le N_G+N_S < 15$
- It is guaranteed that there's exactly one home.

## Input Format

The first line of input file contains an integer $T$, denoting the number of test cases. Each of the following test cases starts with two numbers $R$ and $C$, denoting the rows and columns of the map of this town. Each of the next $R$ line contains $C$ characters, denoting the map of this town. As described before, 'H' denotes his home, 'V' for valleys, 'G' for gas stations, 'S' for the scenery to visit, and '.' for the plane.

## Output Format

For each test case, output a single integer $M$ in a line, denoting the minimum size of the tank John needs in order to visit all beautiful scenery, starting from his home. Output -1 if he cannot complete the tour no matter how large the tank is.

## Example

| **Sample Input:** | **Sample Output:** |
|---|---|
| 2<br>4 4<br>G...<br>S.VS<br>.HVS<br>G.G.<br>1 4<br>H.VS | 3<br>-1 |

*The first sample input is the same as Figure 1 and 2.