ACM-ICPC Indonesia National Contest 2015

Problem A

Remedial Test

Time Limit: 2 seconds

Ms. Lina is a high school teacher, specialized in mathematics. She loves to teach and awe her students with the wonder of mathematics (that's how she described her job). One thing she hates of being a math teacher is preparing the test problems.

There are N students in all Ms. Lina's classes; those students have taken the math exams, and Ms. Lina has graded each of them into score of 0 to 100. By school regulation, student whose score is strictly lower than 60, is obliged to take a remedial test (a.k.a. repeating test for failed students). For example, let there be 10 students whose math score are: {100, 89, 45, 78, 66, 50, 30, 90, 95, 60}. Among these 10 students, three students have score less than 60, i.e. 45, 50 and 30. The last student has score of exactly 60, s/he is safe. Those three students have to take the remedial test and that is Ms. Lina's job to prepare them.

Given the scores of N students, find out how many students are there who has to take the remedial test.

Input

The first line of input contains an integer T ($T \leq 100$) denoting the number of cases. Each case begins with an integer N ($1 \leq N \leq 100$) denoting the number of students in Ms. Lina's classes. The following line contains N integers S_i ($0 \leq S_i \leq 100$) representing the math score of the i-th student.

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1, and `Y` is the number of student who has to take the remedial test.

Sample Input

```
100 89 45 78 66 50 30 90 95 60

45 20 31 59 0

80 49 75 90 88 95 62
```

Output for Sample Input

```
Case #1: 3
Case #2: 0
Case #3: 5
Case #4: 1
```

Explanation for 2nd sample case

Ms. Lina only has 1 (smart) student with a perfect score of 100; there's no need to do remedial test for this student.

Explanation for 3rd sample case

Ms. Lina has 5 students, and unfortunately, all of her students have bad score (lower than 60). All of them have to take the remedial test.

Explanation for 4th sample case

Only 1 student has to take the remedial test, i.e. the one with math score of 49.

ACM-ICPC Indonesia National Contest 2015

Problem B

Ticket Itinerary

Time Limit: 2 seconds

When travelling overseas, one thing we should pay attention to is the time zone difference. For example, consider the following ticket itinerary.

Date	Flight Number	Departing	Arriving
20 Aug 15	BN001	Jakarta (CGK) 18:15	Tokyo (NRT) 04:20 +1

Flight BN001 departs from Jakarta - Soekarno-Hatta International Airport at Cengkareng - at 18:15 on 20 Aug 2015, and arrives in Tokyo - Narita International Airport - at 04:20 the following day, 21 Aug 2015. From the itinerary, it appears the flight take 10 hours and 5 minutes, from 18:15 to 04:20. However, Tokyo has a different time zone than Jakarta. Jakarta is GMT+7 while Tokyo is GMT+9. In other word, 04:20 in Tokyo time is equal to 02:20 in Jakarta time, thus, the actual flight only takes 8 hours and 5 minutes.

Note that +1 in 04:20 means it's 04:20 in the following day. Similarly, -1 means the day before.

Given the time zone of each city and several flight itineraries, determine the flight duration for each itinerary. You may assume that there are no flights which take more than 20 hours.

* GMT+K is K hours offset added to time at the Royal Observatory in Greenwich, London, which has GMT+0. Similarly, GMT-K is K hours offset subtracted to time at GMT+0. For example, 08:00 GMT+0 is equal to 06:00 GMT-2 or 11:00 GMT+3.

Input

The first line of input contains an integer N (2 ≤ N ≤ 100) denoting the number of cities. The next N lines, each contains a string S followed by "GMT+K" or "GMT-K" denoting the time zone on city S. S consists of uppercase alphabets with length between 1 and 15, inclusive; K is between (-)14 and (+)12, inclusive. The next line contains an integer T (1 ≤ T ≤ 100) denoting the number of cases. Each case consists of string in the following format: A HH_A:MM_A B HH_B:MM_B D. A is the origin city, and the flight departs at HH_A:MM_A local time (A's time zone). B is the destination city and the flight arrives at HH_B:MM_B local time (B's time zone). HH:MM constitutes the time (hour and minute, respectively), it ranged from 00:00 to 23:59. D is optional and denotes the difference between departure and arrival date; D is either -1, +1, or none. See sample input for clarity.

Output

For each case, output in a line "`Case #X: H hours and M minutes`" where `X` is the case number, starts from 1. H and M institute the flight duration in hour and minute (0 ≤ H ≤ 20; 0 ≤ M ≤ 59), respectively, from the origin to the destination city. You may safely assume there are no flights with more than 20 hours, and H and M are strictly non-negative.

Sample Input

```
6
JAKARTA GMT+7
TOKYO GMT+9
SINGAPORE GMT+8
GREENWICH GMT+0
BANGKOK GMT+7
CHICAGO GMT-6
4
JAKARTA 18:15 TOKYO 04:20 +1
JAKARTA 04:00 TOKYO 14:30
BANGKOK 19:25 JAKARTA 21:50
SINGAPORE 00:00 CHICAGO 23:30 -1
```

Output for Sample Input

```
Case #1: 8 hours and 5 minutes
Case #2: 8 hours and 30 minutes
Case #3: 2 hours and 25 minutes
Case #4: 13 hours and 30 minutes
```

Explanation for 2nd sample case

14:30 in TOKYO time (GMT+9) is 12:30 in JAKARTA time (GMT+7); therefore, the flight takes 8 hours and 30 minutes.

Explanation for 3rd sample case

BANGKOK and JAKARTA has the same time zone (GMT+7).

Explanation for 4th sample case

The flight from SINGAPORE arrives at the previous day in CHICAGO. 23:30 in CHICAGO (GMT-6) is 13:30 in SINGAPORE (GMT+8); thus the flight takes 13 hours and 30 minutes.

ACM-ICPC Indonesia National Contest 2015

Problem C

Simple Arithmetic Progression

Time Limit: 2 seconds

Arithmetic progression is a sequence of numbers such that the difference between any two consecutive terms is constant. The first term in arithmetic progressing is a_1 and the difference with consecutive terms is d. For example,

- 2, 5, 8, 11, 14, 17, ...
  is an arithmetic progression with $a_1 = 2$ and $d = 3$.

- 5½, 12½, 19½, 26½, 33½, 40½, ...
  is an arithmetic progression with $a_1 = 5½$ and $d = 7$.

- 27 ,46, 65, 84, 103, 122, ...
  is an arithmetic progression with $a_1 = 27$ and $d = 19$.

- 10, 11, 12, 13, 14, 15, ...
  is an arithmetic progression with $a_1 = 10$ and $d = 1$.

On the other hand, arithmetic series is the sum of a finite arithmetic progression. The sum of n-th first term of arithmetic progression is denoted as S_n. Consider the following arithmetic progression: 2, 5, 8, 11, 14, 17, ...; in this sequence, S_1 = 2, S_2 = 2 + 5 = 7, S_3 = 2 + 5 + 8 = 15, and so on.

In this problem, we're interested only on one particular arithmetic progression, i.e. those which start from positive integer and have a difference d of exactly 1. How many integer N are there in [A, B] (between A and B, inclusive), such that there are exactly K finite arithmetic progression of $d = 1$ which start from positive integer and sum up to N, and K is [R, S] (between R and S, inclusive)?

For example, let ⟨A, B, R, S⟩ be ⟨1, 15, 2, 3⟩.

With N = 1, there is only one possible sequence: (1), thus there is 1 finite arithmetic progression of $d = 1$ which start from positive integer number and sum up to 1, or for simplicity, we say f(1) = 1. With N = 5, there are two possible sequences: (5), and (2, 3); thus f(5) = 2. With N = 15, there are four possible sequences: (15), (7, 8), (4, 5, 6), and (1, 2, 3, 4, 5); thus f(15) = 4. If we summarize for all N = 1..15, then we'll get:

f(1) = 1 : (1)	f(6) = 2 : (6 | 1, 2, 3)	f(11) = 2 : (11 | 5, 6)
f(2) = 1 : (2)	f(7) = 2 : (7 | 3, 4)	f(12) = 2 : (12 | 3, 4, 5)
f(3) = 2 : (3 | 1, 2)	f(8) = 1 : (8)	f(13) = 2 : (13 | 6, 7)
f(4) = 1 : (4)	f(9) = 3 : (9 | 4, 5 | 2, 3, 4)	f(14) = 2 : (14 | 2, 3, 4, 5)
f(5) = 2 : (5 | 2, 3)	f(10) = 2 : (10 | 1, 2, 3, 4)	f(15) = 4 : (15 | 7, 8 | 4, 5, 6 | 1, 2, 3, 4, 5)

Set X = {3, 5, 6, 7, 9, 10, 11, 12, 13, 14} satisfies $2 \le f(x) \le 3$ and $x \in X$, where $X \subset \{1..15\}$ and X is maximal. Therefore, there are 10 values for N where $\langle 1, 15, 2, 3 \rangle$ can be satisfied.

Input

The first line of input contains an integer T ($1 \le T \le 100$) denoting the number of cases. Each case contains four integers A, B, R, and S ($1 \le A \le B \le 5{,}000{,}000$; $1 \le R \le S \le 5{,}000{,}000$).

Output

For each case, output in a line the "`Case #X: Y`" where `X` is the case number starts from 1, and `Y` is the number of integer N between A and B, inclusive, such that there are exactly K (between R and S, inclusive) finite arithmetic progressions with $d = 1$ and sum up to N.

Sample Input

```
4
1 15 2 3
1 15 3 4
11 30 3 5
1 100 1 100
```

Output for Sample Input

```
Case #1: 10
Case #2: 2
Case #3: 6
Case #4: 100
```

Explanation for 2nd sample case

Only 9 and 15 satisfy $\langle 1, 15, 3, 4 \rangle$; consult previous example given in the problem statement.

Explanation for 3rd sample case

Only 15, 18, 21, 25, 27, and 30 satisfy $\langle 11, 30, 3, 5 \rangle$.

Problem D

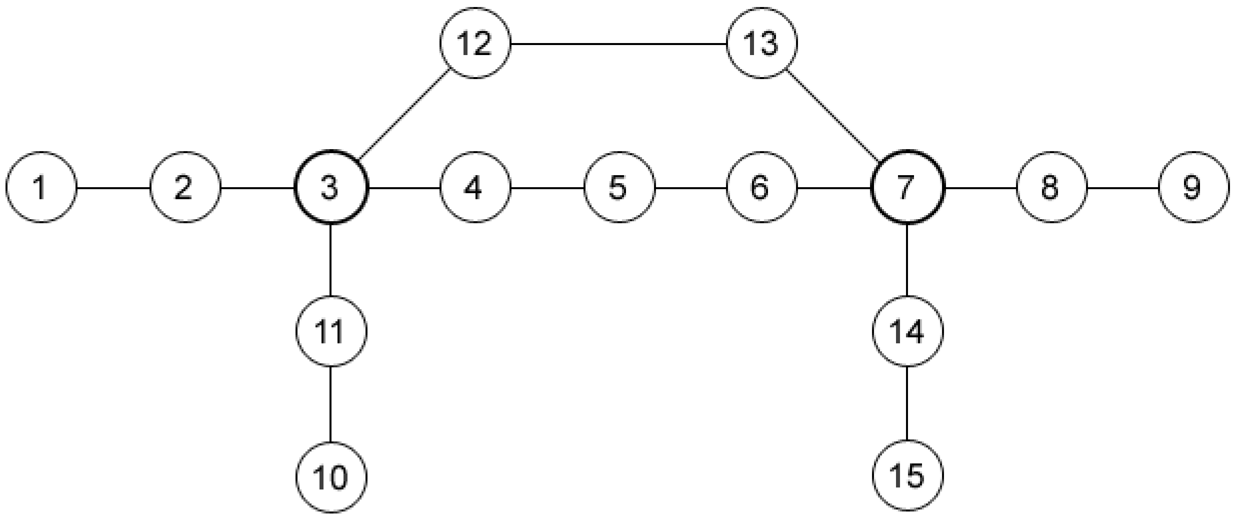
Rapid Transit

Time Limit: 6 seconds

Rapid transit is a type of intra-city public transport (similar to conventional train, in shape) generally found in urban areas. Unlike conventional train, rapid transit operates on exclusive electric railways and cannot be accessed or crossed by any pedestrians or other vehicles. Thus, usually rapid transit runs in tunnels or elevated railways. Because of this exclusivity, rapid transit is able to operate at high speed and short interval (typically every 3-5 minutes). Rapid transit comes with various names, e.g., in Singapore, it's known as MRT (Mass Rapid Transit); in Hong Kong, it's MTR (Mass Transit Railway); in Japan, Metro Subway.

Jakarta – currently, the densest city in Indonesia – has decided to build a rapid transit system, which will be named "Jakarta MRT". Of course, building a rapid transit system is a long term project and cannot be accomplish in one term of office, it can take years to complete (~5 years for stage 1). To illustrate, our neighboring country, Singapore, foresaw the problem in 1967, the (physical) project started in 1983, and by 1987 they have their first MRT line, which consists of only 5 stations; the development still continues up to today and now they have 5 lines and 109 stations in operation.

Several problems any operator needs to be ready when running an MRT are plans for emergency or unexpected breakdown, overcoming rush/peak hours, and reducing overstayed passenger. One possible solution for the last one is by applying an extra charge for passengers who overstay in MRT (station or train). To determine whether a passenger overstay, first we should calculate the number of stations travelled by the passenger. Of course, it is an easy task if there is only one line. However, in reality, there might be multiple lines with overlapping stations. Consider the following example (for simplicity, each station is represented by a number).



In this example, there are 2 lines: A (1, 2, 3, 4, 5, 6, 7, 8, 9) and B (10, 11, 3, 12, 13, 7, 14, 15), with two interchanges: station 3 and 7. Passenger can freely switch line in these interchanges without any charge. Train travels in both direction of each line; for example, from 1 → 2 → 3 → 4 → ... → 9, and in its reversed direction, 9 → 8 → 7 → 6 → ... → 1. To travel from station 2 to station 9, there are 2 possible ways:

- 2 line A | → 3 → 4 → 5 → 6 → 7 → 8 → 9; which consists of 7 stations.
- 2 line A | → 3 (switch to B) → 12 → 13 → 7 (switch to A) → 8 → 9; which consists of 6 stations.

Therefore, station 9 can be reached from station 2 by traveling at least 6 stations (excluding the origin station, station 2).

In this problem, you have to calculate the minimum number of stations one needed to travel from the origin to the destination's station. Assume there are at most 10 interchanges in each line, and each station appears at most once in each line (otherwise, the lines will be very messy, isn't it?).

Input

The first line of input contains an integer T ($1 \le T \le 100$) denoting the number of cases. Each case begins with an integer N ($1 \le N \le 100$) denoting the number of MRT lines. The next N lines, each begins with an integer M_i ($2 \le M_i \le 100$) denoting the number of stations in i-th line ($i = 1..N$, respectively), follows by M_i integer $S_{i,j}$ ($1 \le S_{i,j} \le 10{,}000$; $S_{i,a} \ne S_{i,b}\ \forall\ a \ne b$) denoting the stations' number in order (any two adjacent stations are connected directly by a rail).

Station $S_{i,j}$ which appear in more than one line are interchanges. You may safely assume that there are at most 10 interchanges in one line. The next line contains an integer Q (1 ≤ Q ≤ 100) denoting the number of queries. The next Q lines, each contains two integers A and B (1 ≤ A, B ≤ 10,000; A ≠ B) denoting the starting and destination stations, respectively for each query.

Output

For each case, output in a line "`Case #X:`" where `X` is the case number, starts from 1. The next Q lines each contain an integer denoting the answer to the queries in the respective case, in input order. If there is no way to travel from the given origin to destination stations, output -1 for that query.

Sample Input

```
2
2
9 1 2 3 4 5 6 7 8 9
8 10 11 3 12 13 7 14 15
3
2 9
4 20
10 7
4
4 1 2 3 4
3 5 3 6
4 8 11 15 20
3 7 11 20
4
1 6
8 20
7 3
1000 2000
```

Output for Sample Input

```
Case #1:
6
-1
5
Case #2:
3
2
-1
-1
```

Explanation for 1st sample case

The lines are as pictured in the problem statement.

- Query 1 – ⟨2, 9⟩: as explained in problem statement.
- Query 2 – ⟨4, 20⟩: there is no station 20.
- Query 3 – ⟨10, 7⟩: 10 | → 11 → 3 → 12 → 13 → 7.

Explanation for 2nd sample case

- Query 1 – ⟨1, 4⟩: 1 | → 2 → 3→ 4.
- Query 2 – ⟨8, 20⟩: 8 | → 11 → 20.
- Query 3 – ⟨7, 3⟩: one cannot travel from station 7 to station 3.
- Query 4 – ⟨1000, 2000⟩: there is neither station 1000 nor station 2000.

ACM-ICPC Indonesia National Contest 2015

Problem E

Matryoshka Dolls

Time Limit: 3 seconds

Matryoshka doll, or also known as Russian nesting doll, is a set of wooden dolls of different size placed one inside another, i.e. a doll of size A can be put inside another doll of size B if and only if A < B. For example, if we have 3 Matryoshka dolls of size A, B, and C, with A < B < C, then we can put A inside B, and B (with A inside) inside C; thus, from outside, it looks like there is only one doll: C.

There are N Matryoshka dolls of different size lined up in a single row and labeled from 1 to N, from leftmost to rightmost, and you want to select as many dolls as possible with two constraints:

1. Supposed there are K dolls selected, then $(L(\pi_i) < L(\pi_{i+1})) \cap (\pi_i < \pi_{i+1}) \; \forall \; 0 \le i < K$ must be satisfied, where π_i is the size of i-th selected doll, $L(\pi_i)$ is the label of the doll with size π_i(recall that all dolls have different size).

2. There are M dolls which must be selected: $\rho_1, \rho_2, ..., \rho_M$. You are guaranteed that these M dolls satisfy the previous first constraint.

Unfortunately, we don't know the absolute size of each doll, we only know the relative size (compared to another doll), but rest assure, the information we provide is enough to determine the total ordering of all dolls.

Input

The first line of input contains an integer T ($T \le 100$) denoting the number of cases. Each case begins with three integers N, M, and R ($1 \le M \le N \le 10{,}000$; $R \le 20{,}000$) denoting the number of dolls available, the number of dolls that must be selected, and the number of size relation between two dolls. The next line contains M integers ρ_i ($1 \le \rho_i \le N$) denoting the label of dolls which must be selected. The next R lines each contains two integer α and β ($1 \le \alpha, \beta \le N$) denoting that the size of doll with label α is smaller than doll with label β. Assume the relation information is enough to determine the total ordering of all dolls.

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1, and Y is the maximum number of dolls which can be selected in that particular case, while satisfying the constraints.

Sample Input

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

Output for Sample Input

```
Case #1: 4
Case #2: 1
Case #3: 2
Case #4: 4
```

```
3 2
4 2 6
1 3
1 2
1 3
1 4
2 3
2 4
3 4
```

Explanation for 1st sample case

Supposed the dolls' size are A, B, C, D, E, F, where A < B < C < D < E < F. The only possible arrangement which complies with the given relations is: B C A E D F.

Label	1	2	3	4	5	6
Size	B	C	A	E	D	F

For example,

- "1 2" means B (label 1) < C (label 2),
- "3 1" means A (label 3) < B (label 1), etc.

Please note that the input is given such that there will be exactly one possible arrangement (total ordering). Among these 6 dolls, the 2nd and 4th dolls should be selected, which means, dolls with size C and E. The maximum numbers of dolls which can be selected while satisfying the constraints are 4, i.e. B, C, E, F.

→ The remaining explanations use the same notation as in explanation above.

Explanation for 2nd sample case

The arrangement: B A, and {B} should be selected. Only B can be selected in this case.

Explanation for 3rd sample case

The arrangement: A C B, and {C} should be selected. We can select {A, C} in this case.

Explanation for 4th sample case

The arrangement: A B C D, and {A, C} should be selected. We can select all dolls {A, B, C, D} in this case.

ACM-ICPC Indonesia National Contest 2015

Problem F

Pointers

Time Limit: 30 seconds

You are given an R rows x C columns grid map where each cell contains an arrow (one of the following: '^', '>', '<', or 'v'). If one stands on a cell, then he/she has to go to the neighbouring cell pointed by the arrow in the cell where he/she stands. Supposed you are at cell (r, c) – row r and column c, then:

- '^' means you should go to cell (r-1, c).
- '>' means you should go to cell (r, c+1).
- '<' means you should go to cell (r, c-1).
- 'v' means you should go to cell (r+1, c).

If the new cell is out of the grid map, then you fall out of the map.

Your task is to modify the arrows, such that if you start from any cell in the map and follow the arrows, then you will get back to the starting point. Determine the minimum number of arrows that you will have to change.

For example, consider the following map of 4 x 2.

```
>v
v<
>v
><
```

There are several ways to accomplish our goal; here are 3 solution examples:

```
><    v<    >v
><    v^    ^<
><    v^    ><
><    >^    ><
```

In the first solution example, we have to change 3 arrows: {(1, 2), (2, 1), (3, 2)}. In the second one, we have to change 6 arrows: {(1, 1), (1, 2), (2, 2), (3, 1), (3, 2), (4, 2)}. In the last one, we have to change 2 arrows: {(2, 1), (3, 2)}. There are many other solution examples, but among those, the minimum number of arrows you need to change is 2 (as in solution example 3).

Input

The first line of input contains an integer T (T ≤ 100) denoting the number of cases. Each case begins with two integers R and C (1 ≤ R, C ≤ 14) denoting the number of rows and columns of the grid map. The following R lines each contains C characters (one of the following: '^', '>', '<', 'v') representing the arrow in each cell.

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1. and Y is the minimum number of arrows you need to change to accomplish the given goal. Output -1 for Y, if it's not possible to do so.

Sample Input	**Output for Sample Input**
`4`	`Case #1: 2`

```
4 2
>v
v<
>v
><
3 4
v<>v
v^^v
>^^<
3 4
v<<<
v^v<
>>>^
1 2
>>
```

```
Case #2: 0
Case #3: 2
Case #4: 1
```

Explanation for 2nd sample case

The map already satisfies the goal; there's no need to change any arrow.

Explanation for 3rd sample case

We need to change at least 2 arrows:

```
v<><
v^v<
>^>^
```

Explanation for 4th sample case

One arrow has to be changed:

```
><
```

Problem G

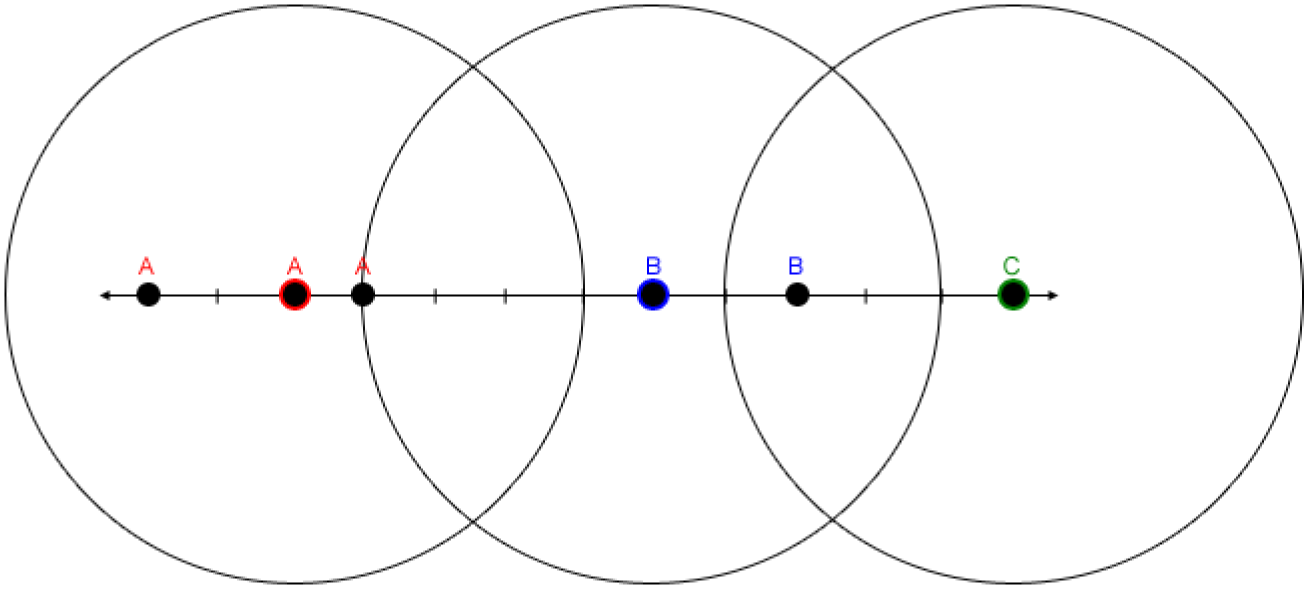
Communication Problem

Time Limit: 2 seconds

In Flatland, there are N army camps located across points in one straight line; they formed a defense line against any invasion threat from its neighboring (hostile) country. These armies have one crucial problem, communication. At their current state, there's no way for each camp to communicate with other camps, despite of the distance. As you might already know, information is one of the most important things in war. In order to solve this problem, the army general is going to build communication towers, so each camp can communicate with any other camp.

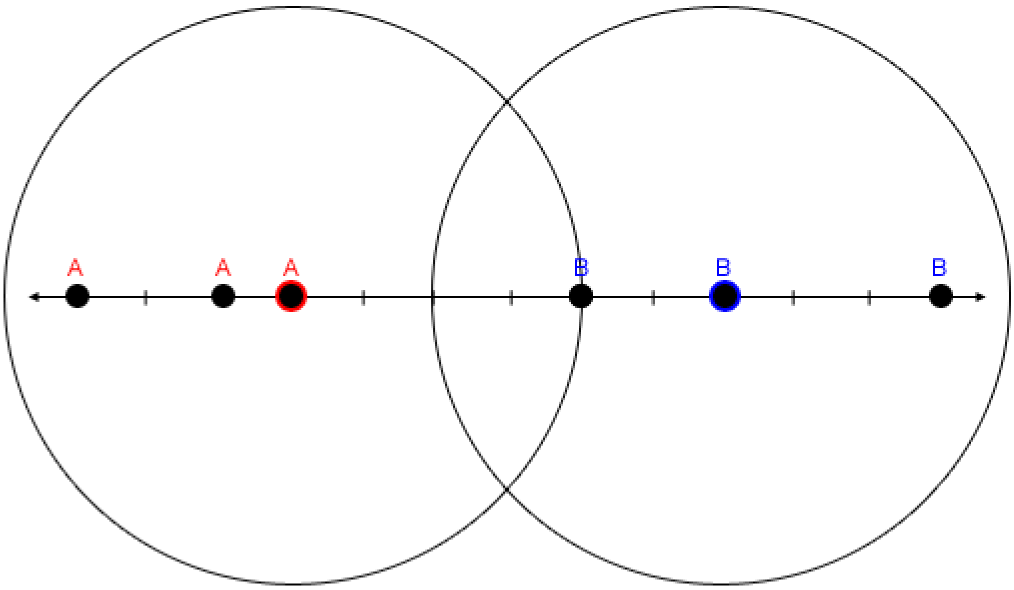
Technology is not this country's strength. Their communication tower can only cover at most K camps within a certain radius R, i.e. if a communication tower located at (10, 0) and the strength of the tower is K = 3 and R = 4, then at most 3 army camps within radius of 4 from (10, 0) can be covered by this tower. Army camps which are covered by a communication tower can communicate with those which are also covered by a tower (not necessarily the same tower).

Building a communication tower cannot be done instantly, thus it needs protection from an army camp. For this reason, communication towers can only be built and deployed at army camp. Consider the following example.



There are 6 camps. Assuming the left most camp is at coordinate x = 1, then the position of those 6 camps are: $X_{1..6}$ = {1, 3, 4, 8, 10, 13}. In the figure above, there are 3 towers of K = 3 and R = 4, each located at 2^{nd}, 4^{th}, and 6^{th} camp, at {3, 8, 13}. 1^{st}, 2^{nd}, and 3^{rd} camps are covered by 1^{st} tower (labeled as 'A'); 4^{th} and 5^{th} camps are covered by 2^{nd} tower (labeled as 'B'); 6^{th} camp is covered by 3^{rd} tower (labeled as 'C'). Each tower covers at most K = 3 camps within R = 4 radius. In this example, 3^{rd} camp actually can also be covered by 2^{nd} tower.

Following is another tower deployment example with the same army camps.



In this example, there are only 2 communication towers which located at 3rd and 5th camp, i.e. {4, 10}. The 4th camp can only be covered by 2nd tower even though it lies within 1st range, as each tower can only cover at most K = 3 camps for this example.

Given the position of N camps and the strength of a communication tower (K and R), determine the minimum number of towers which must be built to cover all the given camps.

Input

The first line of input contains an integer T (T ≤ 100) denoting the number of cases. Each case begins with three integers: N, K, R ($1 \le N \le 1{,}000$; $1 \le K \le 1{,}000$; $1 \le R \le 10^9$) denoting the number of camps and the strength of a communication tower as described in the problem statement, respectively. The next line contains N integers: X_i ($1 \le X_i \le 10^9$) representing the position of i^{th} camp for i = 1..N.

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1, and `Y` is the minimum number of communication towers which must be built to cover all camps.

Sample Input

```
4
6 3 4
1 3 4 8 10 13
6 3 4
4 8 13 10 1 3
5 2 10
100 100 100 100 100
7 2 3
15 21 150 151 207 208 209
```

Output for Sample Input

```
Case #1: 2
Case #2: 2
Case #3: 3
Case #4: 5
```

Explanation for 2nd sample case

i^{th} camp is not necessarily to the left of $(i+1)^{th}$ camp. Note that the positions of those 6 camps are the same with example in the problem statement.

Explanation for 3rd sample case

There are 5 camps at the same position (recall from the problem statement that, without communication towers, all camps cannot communicate to each other despite of the distance, even on zero distance) and a communication tower can only cover 2 camps at once. Thus at least 3 towers should be built.

Explanation for 4th sample case

One possible towers deployment:

- 1st tower at 1st camp (15) and covers only 1st camp.
- 2nd tower at 2nd camp (21) and covers only 2nd camp.
- 3rd tower at 3rd camp (150) and covers 3rd and 4th camp.
- 4th tower at 5th camp (207) and covers 5th and 6th camp.
- 5th tower at 7th camp (209) and covers only 7th camp. Note that 4th tower can only cover K = 2 camps.

ACM-ICPC Indonesia National Contest 2015

Problem H

Rank Query

Time Limit: 8 seconds

You are given an array of N integers – A[1..N], and Q queries of ⟨L, R, K⟩. For each query, output the K^{th} smallest element between the index range L..R in array A.

For example, let A[1..10] = {5, 7, 10, 2, 1, 1, 4, 6, 13, 3}, and Q = 3.

- Query ⟨1, 10, 3⟩. $A_{1..10}$ is {5, 7, 10, 2, 1, 1, 4, 6, 13, 3} (the whole array), and the 3^{rd} smallest element is A_4 = 2 (while the 1^{st} and 2^{nd} smallest are both 1).
- Query ⟨2, 6, 4⟩. $A_{2..6}$ is {7, 10, 2, 1, 1}, and the 4^{th} smallest element is A_2 = 7.
- Query ⟨7, 10, 1⟩. $A_{7..10}$ is {4, 6, 13, 3}, and the 1^{st} smallest element is A_{10} = 3.

Input

The first line of input contains an integer T (T ≤ 100) denoting the number of cases. Each case begins with two integers N and Q (1 ≤ N, Q ≤ 10,000) denoting the number integers in the array and the number of query respectively. The next line contains N integers A_i ($1 \le A_i \le$ 1,000,000,000) denoting the element in the array for i = 1..N respectively. The following Q lines each contains three integers L, R, and K (1 ≤ L ≤ R ≤ N; 1 ≤ K ≤ R - L + 1) which represent a query as described in the problem statement.

Output

For each case, output in a line "`Case #X:`" where `X` is the case number, starts from 1. The next Q lines each contains an integer representing the K^{th} smallest element in the segment of given array for that particular query, as in input order.

Sample Input

```
4
10 3
5 7 10 2 1 1 4 6 13 3
1 10 3
2 6 4
7 10 1
5 2
1 2 3 4 5
3 5 2
1 4 3
5 1
1002 49301 2339 1004 17
```

Output for Sample Input

```
Case #1:
2
7
3
Case #2:
4
3
Case #3:
1004
Case #4:
11
18
```

```
2 4 1
6 2
18 3 74 11 15 11
2 6 3
1 4 3
```

Explanation for 2nd sample case

- Query ⟨3, 5, 2⟩. $A_{3..5}$ = {3, 4, 5}, the 2nd smallest is A_4 = 4.
- Query ⟨1, 4, 3⟩. $A_{1..4}$ = {1, 2, 3, 4}, the 3rd smallest is A_3 = 3.

Explanation for 3rd sample case

- Query ⟨2, 4, 1⟩. $A_{2..4}$ = {49301, 2339, 1004}, the 1st smallest is A_4 = 1004.

Explanation for 4th sample case

- Query ⟨2, 6, 3⟩. $A_{2..6}$ = {3, 74, 11, 15, 11}, the 3rd smallest is A_4 = A_6 = 11 (both are the 2nd and the 3rd smallest).
- Query ⟨1, 4, 3⟩. $A_{1..4}$ = {18, 3, 74, 11}, the 3rd smallest is A_1 = 18.

ACM-ICPC Indonesia National Contest 2015

Problem I

Hackbook Password

Time Limit: 2 seconds

Budi wants to create a new account on Hackbook, a growingly popular website where programmers post interesting statuses about their daily lives.

When he opens the registration form, there are strange rules on a valid password that must be followed by any users:

- A password should consist of lower case alphabet.
- A password should be similar to an Indonesian word (so that only few users will forget their passwords).
- A password is said to be similar to an Indonesian word if and only if it consists of one or more syllables joined together.
- A syllable must be in the form of CV or CVC, where C is a consonant and V is a vowel.
- Indonesian vowels are 'a', 'e', 'i', 'o', and 'u'. The rest of the alphabets are consonants.

Budi really wants to use a string S for his password on Hackbook. If S is not a valid Hackbook password, Budi wants to rearrange the letters in S so that the final string is a valid Hackbook password.

Help Budi to rearrange the letters!

Input

The first line of input contains an integer T (T ≤ 20) denoting the number of cases. Each case consists of a string S (between 1 and 100 letters) in a line. Each letter in S will be a lower case alphabet ('a' - 'z').

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1, and `Y` is an arrangement of letters in S which comply with a valid Hackbook password. If there is more than one solution, output any of them. If no valid arrangements exist, Y should be the string "IMPOSSIBLE" without quotes.

Sample Input	Output for Sample Input
<code>5 slolo ek jakarta hulkbana icpc</code>	<code>Case #1: lolos Case #2: ke Case #3: jakarta Case #4: bukanlah Case #5: IMPOSSIBLE</code>

ACM-ICPC Indonesia National Contest 2015

Problem J

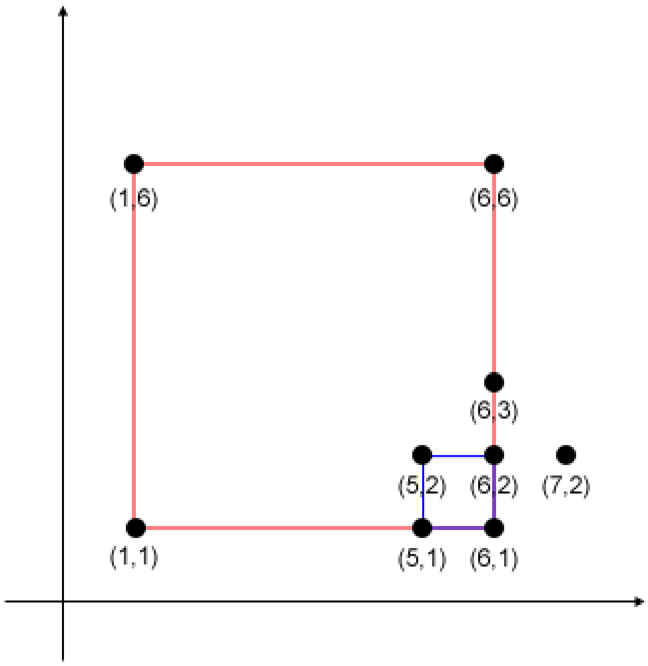
Counting Square Stars

Time Limit: 2 seconds

Lately, Windarik has a new hobby, star-gazing. In order to do this, he has to go to a secluded area (preferably mountain) with a minimum light pollution at night so that he can observe those beautiful stars. It's hard to observe stars in a big city, as usually the light pollution is strong enough to interfere. Windarik still has a lot to learn about stars, e.g., constellations. Yeah, like those Aries, Taurus, Sagittarius, etc. No, I'm not talking about Zodiacs, those are astrology; I'm talking about astronomy. For example, Aries constellation contains several stars with known planet; its three brightest stars are named Alpha Arietis (orange giant), Beta Arietis (blue-white), and Gamma Arietis (binary white star).

Windarik defined his own constellation system, which he called Square Stars (SS). In astronomy, a constellation is a specific area (region) of a celestial sphere which may contain many celestial objects such as stars, planets, and other objects. However, since Windarik is an amateur star-gazer (not to mention, he's quite naive), he thought a constellation is an arrangement of a certain number of stars; thus, a star can belong to more than one constellation as long as the shape or arrangement is satisfied.

In this problem, simple 2D Cartesian coordinate system will be used as opposed to celestial coordinate system which is the standard coordinate system for celestial objects. Consider the following 9 stars' coordinate.



Windarik defined a Square Stars as 4 stars which form a perfect square with its sides parallel to x-axis or y-axis. In the above examples, we can observe 2 SS: {(1, 1), (1, 6), (6, 6), (6, 1)}, and {(5, 1), (5, 2), (6, 2), (6, 1)}, where the first SS has a side of length 5, and the later one has a side of length 1. On the other hand, {(5, 2), (6, 3), (7, 2), (6, 1)} are not SS even though they form a square as their side are not parallel to x-axis or y-axis. Among those 9 stars, 7 stars belong to at least one SS: {(1, 1), (1, 6), (6, 6), (6, 1), (5, 1), (5, 2), (6, 2)}, especially the star at (6, 1) belong to two SS. Stars at (6, 3) and (7, 2) are not belong to any SS.

Given the position of N stars in Cartesian coordinate system, determine how many stars belong to at least one Square Stars system.

Input

The first line of input contains an integer T (T ≤ 50) denoting the number of cases. Each case begins with an integer N (1 ≤ N ≤ 30) denoting the number of stars. The next N lines, each contains two integers X and Y (0 ≤ X, Y ≤ 1,000) denoting the coordinate (x, y) of a star. You may assume there are no two stars in the same position.

Output

For each case, output in a line "`Case #X: Y`" where `X` is the case number, starts from 1, and `Y` is the number stars which belong to at least one Square Stars constellation.

Sample Input

```
4
9
1 1
1 6
6 6
6 1
5 1
5 2
6 2
6 3
7 2
3
10 10
20 20
10 20
4
10 10
20 20
10 20
20 10
10
1 1
1 2
1 3
2 1
2 2
2 3
3 1
3 2
3 3
5 5
```

Output for Sample Input

```
Case #1: 7
Case #2: 0
Case #3: 4
Case #4: 9
```

Explanation for 2nd sample case

An SS should contain 4 stars, thus in this case, there are no stars belong to an SS.

Explanation for 3rd sample case

Those 4 stars belong to an SS.

Explanation for 3rd sample case

9 of those stars, except star at (5, 5), belong to at least one SS. BTW, there are 5 SS in this case, can you find them all?

ACM-ICPC Indonesia National Contest 2015

Problem K

Going Home

Time Limit: 3 seconds

Suppavin lives in our neighboring country (though, might not in the same dimension), Indinesia. Unlike our beloved nation, Indinesia is a continental nation, which means it's mainly spanned in a continent, as opposed to archipelago in Indonesia. There are N cities, numbered from 0 to N-1, which are connected by M unidirectional roads in Indinesia. Suppavin's home is in city N-1, and every working day, he has to commute from his office in city 0, the capital. Suppavin doesn't have to worry about his journey from his home to his office; his home has a one-way teleportation device which connected to his office (sponsored by the government, of course). The problem is the going home journey from his office to his home. Even though he has some kind of superpower (he moves extremely fast, just like Flash), given the distance, commuting from his office to his home surely is tiring.

Indinisia government put roads development to support their citizens' movement into plan. The government proposed Q (one-way) roads to be built, but given the budget constraint, only one which will be executed; the selection will be decided by voting from the citizens. Suppavin got excited from this news. He's going to evaluate each of the proposed roads, and determine whether the new road will shorten the commuting distance from his office to his home.

Brawl and bravery is Suppavin's merit, but not brain. He needs your help to evaluate all the proposed roads.

Input

The first line of input contains an integer T (T ≤ 100) denoting the number of cases. Each case begins with three integers N, M, and Q (1 ≤ N ≤ 20,000; 1 ≤ M ≤ 50,000; 1 ≤ Q ≤ 50,000) denoting the number of cities, the number of unidirectional roads, and the number of proposed new roads. The following M lines each contains three integers A_i, B_i, and C_i ($0 \le A_i$,

B_i < N; 0 ≤ C ≤ 1,000,000,000) denoting a one-way road from city A to city B with length of C. Note that there may be more than one road connecting from one city to another city. The next Q lines each contains three integers X, Y, and Z (0 ≤ X, Y < N; 0 ≤ Z ≤ 1,000,000,000) denoting a proposal of new one-way road connecting from city X to city Y with distance of Z.

Output

For each case, output in a line "`Case #X:`" where `X` is the case number, starts from 1. The next Q lines, each contains either "YES" or "NO" (without quotes) whether the respective proposed road will shorten the distance from Suppavin's office to his home (in input order).

Sample Input

```
2
4 3 2
0 1 10
1 2 20
2 3 30
0 2 10
0 2 40
5 6 3
0 1 1000
1 2 1000
2 3 1000
3 4 1000
0 2 10
2 1 10
1 3 20
3 0 10
0 3 10
```

Output for Sample Input

```
Case #1:
YES
NO
Case #2:
YES
NO
YES
```