Problem A

Number Assignment

Given N integers, you have to assign these integers into M groups such that each group contains at least one element and each integer belongs to exactly one group (let's call this process as number assignment). The cost of a group is defined as the difference between the largest and the smallest element in the group. If the group has only one element, then the cost for that group will be zero. The cost of an assignment is defined as the sum of all groups' cost.

For example, let there be 8 integers: 5, 2, 3, 10, 7, 2, 6 and 8. Let's say that you have to assign these integers into 3 groups. There are plenty of ways to do that and some of those are shown in these three figures below.

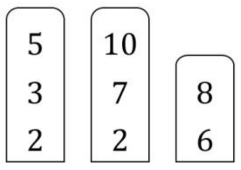


Figure 1

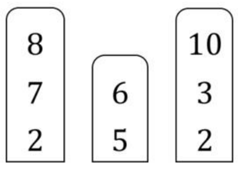


Figure 2

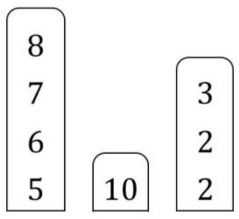


Figure 3

The cost of assignment in Figure 1 is: (5 – 2) + (10 – 2) + (8 – 6) = 3 + 8 + 2 = 13.
The cost of assignment in Figure 2 is: (8 – 2) + (6 – 5) + (10 – 2) = 6 + 1 + 8 = 15.
The cost of assignment in Figure 3 is: (8 – 5) + 0 + (3 – 2) = 3 + 0 + 1 = 4.

Apparently the assignment in Figure 3 has the lowest cost among all possible assignments.

Your task is to find the minimum cost of such number assignment.

Input

The first line of input contains an integer T ($T \leq 100$) denoting the number of cases. Each case begins with two integers N and M ($1 \leq M \leq N \leq 100$) denoting the number of integers and the number of groups respectively. The following line contains N integers A_i ($0 \leq A_i \leq 10^9$) each separated by a single space representing the given integers from the problem statement.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the minimum cost of the number assignment.

Sample Input	Output for Sample Input

`4`
`8 3`
`5 2 3 10 7 2 6 8`
`5 5`
`10 1 29 15 6`
`4 1`
`105 3 27 86`
`10 2`
`37 290 15 60 4 39 2 8 275 301`
	`Case #1: 4`
	`Case #2: 0`
	`Case #3: 102`
	`Case #4: 84`

This page is intentionally left blank

Problem B
Network Packet Ordering

Some of us sometimes overestimated the speed of light. In fact, light at vacuum travels at "only" 299,492,758 m/s, which is actually pretty slow for transffering network packets through fiber optic cables.

Imagine transferring network packets from New York to Jakarta through 20,000 km-long fiber optic cables. The photons in the fiber optics will need 66.7 msec at a minimum to travel such distance, and this assumes the cable is very optimally arranged to minimize its length, light travels with minimum reflections inside the cable, and there is no other delays caused by network routing, switching, relaying, etc.

For some large internet companies, like Gogololo, Inc. which deals with data centers across the globe, the delays are actually quite annoying. Suppose Gogololo's server in New York sends packets of data labelled $a_1, a_2, \ldots, a_N$ to a database server in Dublin, and its Jakarta's server also sends packets of data labelled $b_1, b_2, \ldots, b_M$ to the same Dublin database. The data $a_1, a_2, \ldots, a_N$ is expected to arrive respectively at time $t_{a_1}, t_{a_2}, \ldots, t_{a_N}$ and the data $b_1, b_2, \ldots, b_M$ is expected to arrive respectively at time $t_{b_1}, t_{b_2}, \ldots, t_{b_M}$ (all timestamps are in milliseconds)

However, due to the "slow" speed of light and uncertainty in the network delays, each packet from the two sources may experiences delay up to D milliseconds, so that a packet that is expected to arrive at time t can arrive at a time t' where:

$$t \le t' < t + D$$

(note: t' can be fractional)

However, ordering protocol on the packets guarantee that all the packets from New York will arrive in the order which is relative to other packets from New York, and all the packets from Jakarta will arrive in the order which is relative to other packets from Jakarta. But it does not guarantee any fixed ordering between the packets from New York and the packets from Jakarta. Assume that any two packets' arrival times must differ by at least some infinitesimal fraction of millisecond, so there is no confusion in their ordering.

As an engineer of Gogololo, Inc. you are tasked with calculating the number of possible final orderings of the packets arriving at the Dublin database.

Input

The first line of input contains an integer T (T ≤ 15) denoting the number of cases. Each case begins with three integers N, M and D (1 ≤ N, M ≤ 50,000; 1 ≤ D ≤ 100) denoting the number of packet from New York, the number of packet from Jakarta, and the uncertainty delay respectively. The next line contains N integers t_{a_i} (1 ≤ t_{a_i} ≤ 1,000,000; $t_{a_i} < t_{a_j}$ for all $i < j$) denoting the expected arrival time of all packets from New York. The following line contains M integers t_{b_i} (1 ≤ t_{b_i} ≤ 1,000,000; $t_{b_i} < t_{b_j}$ for all $i < j$) denoting the expected arrival time of all packets from Jakarta.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the number of possible ordering modulo by 1,000,000,009.

Sample Input	Output for Sample Input

2
3 3 1
5 6 7
3 4 5
3 2 4
1 2 3
1 2
	Case #1: 2
	Case #2: 10

Explanation for 1st sample case
There are only two variations of the final ordering: b_3 before a_1 or b_3 after a_1.

Explanation for 2nd sample case
All possible interleavings between the New York packets and the Jakarta packets are possible, forming 10 possible final orderings.

Problem C
The Busiest City

Tree Land Kingdom is a prosperous and lively kingdom. It has N cities which are connected to each other by roads such that there is exactly one path to go from one city to any other city. Each road in the kingdom connects exactly two different cities.

Every day a lot of merchants travel from one city to other cities in the kingdom making this kingdom famous for its commerce. The king of this kingdom wonders, which city is the busiest one in his entire kingdom. The busyness of a city is defined as the number of merchants who visits this city on each day. A merchant is considered as visiting city c if and only if city c lies on the path when the merchant travels from city a to city b.

Unfortunately, we need a lot of resources and time to answer the king's question. Therefore, the ministers come up with an idea to approximate the answer so they can provide the king with an "early" answer while they are working on the actual answer. To approximate the answer, the ministers modify the definition of a city's busyness a bit. The busyness of a city a is now defined as the number of different pair of cities a-b such that c lies in a simple path from a to b (note that c is neither a nor c). A path is considered simple if and only if it does not visit any city more than once.

Consider the example as shown in Figure 1 below.

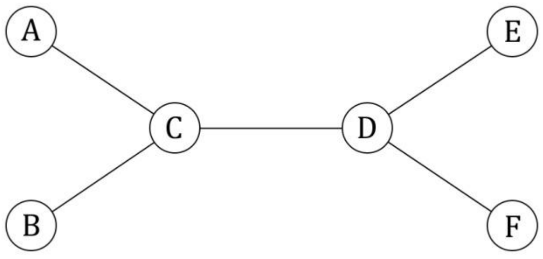


Figure 1

In this example, the busyness of city A, B, E and F are 0 because there is no pair of cities which path visits those nodes. The busyness of city C is 7 (the pairs are: A-B, A-D, A-E, A-F, B-D, B-E, B-F) and the busyness of city D is also 7 (the pairs are: A-E, A-F, B-E, B-F, C-E, C-F, E-F). Therefore, the highest busyness in this example is 7, which occurs in city C and D.

Given the kingdom's road structure, your task is to determine the highest busyness among all cities in the kingdom.

Input

The first line of input contains an integer T ($T \leq 50$) denoting the number of cases. Each case begins with an integer N ($3 \leq N \leq 20{,}000$) denoting the number of cities in the kingdom. The cities are numbered from 1 to N. The following N-1 lines each contains two integers a and b ($1 \leq a, b \leq N$) denoting a road which connects city a and city b.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the highest busyness among all cities in the kingdom for that case.

Sample Input	Output for Sample Input

4
6
1 3
2 3
3 4
4 5
4 6
3
1 2
2 3
4
1 2
2 3
2 4
7
2 5
1 2
7 4
3 7
2 3
7 6
	Case #1: 7
	Case #2: 1
	Case #3: 3
	Case #4: 9

Explanation for 1st sample case
This sample case corresponds to Figure 1 in the problem statement.

Explanation for 2nd sample case
The busiest city is city 2 with busyness of 1 (the pair is: 1-3).

Explanation for 3rd sample case
The busiest city is city 2 with busyness of 3 (the pairs are: 1-3, 1-4, 3-4).

Problem D

Power Plant

Indonesia is the world's largest archipelago with approximately 17,000 islands scattered for more than 5,000 km from Sabang (west most) in Sumatra island to Merauke (east most) in Papua island. Thus, providing electricity in all cities and towns across all islands is a challenging problem for the government.

Power plants, cities, towns and all other important sites can be represented as a graph where each node represents a site and each edge which connects two different sites represents a cable transferring electricity between the two sites in both directions. You may assume that all sites are connected through some cables and for each pair of sites there is at most one cable connecting them. There is of course a cost to maintain each cable and some of them probably have a very high cost to maintain.

The government has a plan to calculate the minimum total cost to maintain only necessary cables such that all sites are connected to at least one power plant, except for the power plants (which are already "connected" to themselves). A site is considered connected to a power plant if and only if there is a path which consists of only maintained cables from the site to a power plant.

Consider the following example. There are 9 sites and 3 of them are power plants (A, H and I). The connectivity and the cost of each cable are shown in Figure 1.

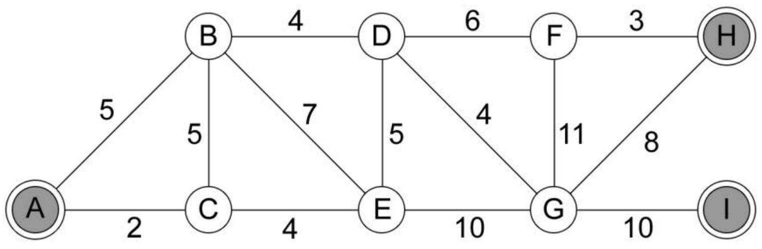


Figure 1

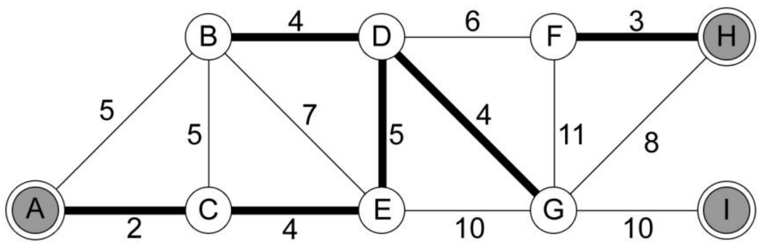


Figure 2

The minimum total cost to maintain the cables in this example is 22 as shown in Figure 2.

Given an undirected graph which represents the connectivity between all sites, determine the minimum total cost needed to maintain cables such that all sites except power plants are connected to at least one power plant.

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 K ($1 \le K \le N \le 200$) denoting the number of sites, the number of cables and the number of power plants respectively. All sites are numbered from 1 to N. The following line contains K integers P_i ($1 \le P_i \le N$; $P_i \ne P_j$ for all $i \ne j$) denoting the sites which are power plants. The next M lines each contains three integers a_i, b_i, c_i ($1 \le a_i$, $b_i \le N$; $a_i \ne b_i$; $1 \le c_i \le 10^6$) denoting that there is a cable connecting site a_i and b_i with cost of c_i. Assume that there is at most one cable connecting site a_i and b_i and all sites are connected to each other.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the minimum total cost needed to maintain the cables such that all sites except power plants are connected to at least one power plant.

Sample Input	Output for Sample Input

3
9 14 3
1 8 9
1 2 5
1 3 2
2 3 5
2 4 4
2 5 7
3 5 4
4 5 5
4 6 6
4 7 4
5 7 10
6 7 11
6 8 3
7 8 8
7 9 10
4 5 1
1
1 2 5
1 3 5
1 4 5
2 3 10
3 4 10
10 9 5
1 4 6 9 10
1 2 3
2 3 8
3 4 5
4 5 1
5 6 2
6 7 6
7 8 3
8 9 4
9 10 1
	Case #1: 22
	Case #2: 15
	Case #3: 16

Explanation for 1st sample case

This sample corresponds to example in the problem statement.

Problem E

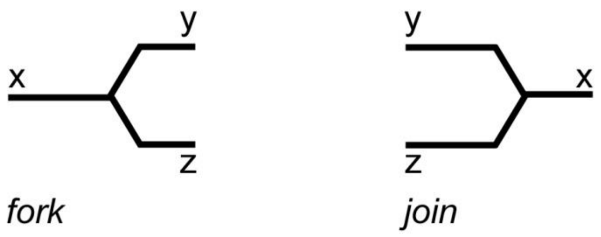
Railroad

For an archipelago like Indonesia, rail transport is not a terribly effective means of transportation, but given the majority of Indonesian population lives in the small Java island (where Jakarta is also in), rail transport is still an attractive solution to connect parts of the island.

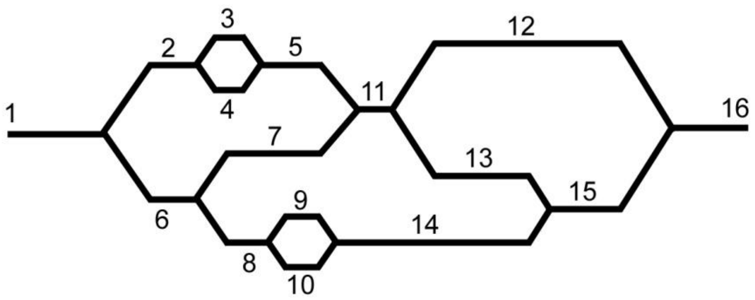
A hypothetical consortium of Java island development proposes a 2030 trans-Java railroad project. It will create a railroad system from the West to the East of Java (for some reasons they are not thinking about the East-to-West rails yet) that starts in Merak (westernmost city of Java) and ends in Banyuwangi (easternmost city of Java).

The rails follow some well-defined patterns. The westernmost segment is numbered 1 and the easternmost segment is numbered N, while all the other segments will be numbered $2 \dots N-1$. All segments are one-directional - it can only be traveled from the west to the east.

The basic building blocks of the railroad system are forks and joins. A fork (x, y, z) forks the segment x to two segments, y on the north, and z on the south (always in that order). A join (y, z, x) joins the segment y from the north and z from the south to segment x (always in that order).



Thus we can imagine creating certain complex rail system using only these forks and joins, such as in the figure below:



There are no crossings and the Java island for this purpose is guaranteed to be planar.

After the railroad system is built, it now comes the time for quality control. The consortium asks you to create a plan to test the rail segments by running trains through the segments. A train can only move from the west to the east, so multiple train runs may be needed to be able to go through each segment at least once. You're asked to optimize the number of train runs to minimize cost of testing.

Input

The first line contains the number of test cases, T ($1 \le T \le 10$). Each case begins with two integers N and K ($4 \le N \le 100{,}000$; $1 \le K \le N$) in a line, where N is the number of segments and K is the number of joins and forks that follow for this case respectively. For the next K lines, each line contains a character ('`F`' or '`J`' indicating fork or join), followed by three segment numbers that define the fork or the join (x y z for forks and y z x for joins). You may assume the given railroad system is valid, that is:

- The rail starts from segment 1 and ends at segment N.
- All segments from 1 to N exist in the railroad system.
- Each segment is a part of at most one fork and one join.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the number of minimum train runs in order to go through all the segments at least once.

Sample Input	Output for Sample Input

`2`
`16 10`
`F 1 2 6`
`F 2 3 4`
`J 3 4 5`
`J 5 7 11`
`F 6 7 8`
`F 8 9 10`
`J 9 10 14`
`F 11 12 13`
`J 13 14 15`
`J 12 15 16`
`7 4`
`F 1 2 3`
`J 2 3 4`
`F 4 5 6`
`J 5 6 7`
	`Case #1: 5`
	`Case #2: 2`

Explanation for 1st sample case

This sample corresponds to example in the problem statement. At least five trains are needed to test the entire rail segments. Here is an example of the trains' tracks:

- Train #1: 1, 2, 3, 5, 11, 12 and 16 (starts from segment 1 and then subsequently moves to segment 2, 3, 5, 11, 12 and ends at segment 16).
- Train #2: 1, 2, 4, 5, 11, 12 and 16.
- Train #3: 1, 6, 7, 11, 13, 15 and 16.
- Train #4: 1, 6, 8, 9, 14, 15 and 16.
- Train #5: 1, 6, 8, 10, 14, 15 and 16.

Explanation for 2nd sample case

At least two trains are needed to test the entire rail segments. Here is an example of the trains' tracks:

- Train #1: 1, 2, 4, 5 and 7.
- Train #2: 1, 3, 4, 6 and 7.

Problem F
Pasti Pas!

Pertamina is an Indonesian government-owned oil and gas corporation based on Jakarta. It has by far the largest distribution network of petroleum products compared to other oil companies in Indonesia. When several foreign oil companies like Shell (Dutch) and Petronas (Malaysia) expanded their business to Indonesia, Pertamina was faced with a great challenge to improve their service quality and strengthen its consumer's trust, especially on their gas stations. At that time, Pertamina started a campaign called "Pasti Pas!" to inform consumers that Pertamina will satisfy them by providing an accurate measurement when they buy oil/gas at their gas stations. "Pasti Pas!" itself means "absolutely accurate". Despite of what happened out there, we will focus our attention to the phrase "Pasti Pas!" in this problem.

"Pasti Pas!" is an interesting phrase. Most of you must already know what a palindrome is, i.e. a word or phrase which can be read the same way in either forward or backward direction, e.g., MADAM, AMOREROMA, etc. Of course "Pasti Pas!" is not a palindrome by this definition; however, if we replace one or more substring into another symbol, we will get a palindrome! For convenience, let's remove all non alphabetical characters from the phrase. Let α = "PAS" and β = "TI", then the phrase "PASTIPAS" will become "α β α" which is a palindrome.

Now we are interested in what the palindrome value of a string S is. Palindrome value of a string S is defined as the length of the longest palindrome string S' where S' is derived from S by replacing one or more substring by some symbols. As for the previous example, PASTIPAS has a palindrome value of 3. Note that when deriving string S, each unique substring can only be mapped into a unique symbol.

Here is another example. Let S = "ABCADDABCA". There are several derivations of S, e.g.:

- Let α="ABCA", β="DD", then S'=" α β α" which has a length of 3.
- Let α="ABCA", β="D", then S'=" α β β α" which has a length of 4.
- Let α="A", β="BC", γ="D", then S'=" α β α γ γ α β α" which has a length of 8.

Among all possible derivations of S, the longest palindrome can be formed has the length of 8.

Input

The first line of input contains an integer T ($T \leq 100$) denoting the number of cases. Each case contains a string S ($1 \leq |S| \leq 50{,}000$; $S \in$ {'A'…'Z'}) in a line.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the palindrome value of S for each case.

Sample Input	Output for Sample Input
`4`	`Case #1: 3`
`PASTIPAS`	`Case #2: 8`
`ABCADDABCA`	`Case #3: 11`
`MADAMIAMADAM`	`Case #4: 1`
`ACMICPCJAKARTASITE`	

This page is intentionally left blank

Problem G

Emergency Handling

Indonesia, as well as some neighboring Southeast Asian countries and some other East Asian countries, is located in the Pacific Ring of Fire which rather frequently causes volcanic eruptions, large earthquakes, and tsunamis. Handling hospital emergency situations effectively is critical for such countries laden with natural disasters.

A leading local hospital recently has asked some computer scientists to help them optimize their emergency handling especially under large volume of cases.

During a day, patients may come to the emergency department with different severity (e.g. suffering from broken bone is more severe than suffering from flu) and with different rate of increase of severity (e.g., burn injury if not treated immediately can be very dangerous, compared to broken bone)

For a simplified model, suppose a patient arrives at time t_0 with initial severity $s(t_0)$ and rate of increase of severity per unit of time r. Then the severity at time t becomes

$$s(t) = s(t_0) + r \cdot (t - t_0)$$

The hospital has only limited facility and workforce to handle the patients so they want to prioritize the handling the best they can. More precisely, at any time t, if the hospital can admit one of the patients to treat, it must choose the patient with the highest severity $s(t)$ at that time t. If there are ties, it must choose the patient with the highest rate of increase of severity r. If there are still ties, any one of the tied patients can be chosen. We assume that once a patient is chosen and given care, the patient will be safe and is no longer in the list of waiting patients.

Given the sequence of incoming patients and admission of patients, help the hospital to determine the best patient to admit at each admission event.

Input

The first line of input contains an integer T ($T \le 5$) denoting the number of cases. Each case begins with an integer N ($1 \le N \le 100{,}000$) denoting the number of events. For the next N lines, each line describes either an incoming patient or an admission event.

- For an incoming patient, the line contains a character ‘`P`’ and three integers t_0, $s(t_0)$ and r that describe the patient ($0 \le t_0 \le 10^6$; $0 \le s(t_0) \le 10^8$; $0 \le r \le 100$).
- For an admission, the line contains a character ‘`A`’ followed by an integer t, which is the time of the event. You may assume that there is at least one patient waiting in the emergency department when this admission event occurs.

The events are specified in strictly increasing order of time. The number of ‘`P`‘ and ‘`A`’ events will be roughly balanced.

Output

For each case, output “`Case #X:`” in a line, where `X` is the case number starts from 1. For each of the admission event, output two integers separated by a single space: the current severity of the chosen patient at the admission time and that patient's rate of increase of severity.

Sample Input	Output for Sample Input
2 9 P 10 10 1 P 30 20 1 A 35 P 40 20 2 P 60 50 3 A 75 P 80 80 3 A 100 A 110 6 P 1 10 2 A 5 P 10 10 1 P 11 1 10 A 15 A 20	Case #1: 35 1 95 3 140 3 160 2 Case #2: 18 2 41 10 20 1

Problem H
Horrible Quiz

Gori, a programmer, participated in the Horrible Quiz and passed the elimination stage. Therefore, he was invited to the final stage: the Horrible Quiz Onsite! Gori will start with a balance of $15,000. He will be given N problems. For each problem, if he solves it, his balance remains unchanged otherwise his balance is multiplied by -1.

He asked you (whom he sees as a friend) to help him. You decided to give him a slip of paper containing the solution to each problem. You claimed that you are one of the staff but you will not guarantee that those solutions are correct. Specifically, you may give wrong solutions to at most M of the problems. You know the correct solution to each of the problems before the quiz starts and it is entirely up to you whether you want to give the wrong or correct solution to each problem.

When the ith problem is given, there is a C_i% chance that Gori will come up and submit a correct solution by himself, a W_i% chance that he will come up and submit a wrong solution, and a (100 – W_i – C_i)% chance that he can't come up with any solution and is forced to submit the solution in the paper you gave him.

Gori has a strange sense of sportsmanship. Although he is cheating by asking for your help, he will not deliberately submit a wrong solution, that is, if he submits a wrong solution it must be either from the solution you gave him or the solution he came up with the W_i% probability.

You are not a very nice person and you have decided to give him the worst possible set of solutions, that is, the one that minimized Gori's expected balance after the game. Output this expected balance, up to three digits after decimal points.

You should assume that Gori's performance of each problem is independent to his performance on all other problems previously given to him. You must provide all solutions to Gori before the start of the quiz. It's not possible to change them during the quiz.

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 M (1 ≤ N ≤ 1,250; 0 ≤ M ≤ N) denoting the number of problems and the maximum number of wrong solution you might give Gori respectively. The next line contains N integers C_i (0 ≤ C_i ≤ 100) representing the chance that Gori will come up and submit a correct solution to the ith problem. The last line of each case contains N integers W_i (0 ≤ W_i ≤ 100; C_i + W_i ≤ 100) representing the chance that Gori will come up and submit a wrong solution to the ith problem.

Output
For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the minimum expected balance with three digits decimal points. Output which differs by at most 10^{-3} to the actual output will be considered correct.

Sample Input

```
5
2 0
0 0
0 0
2 2
0 0
0 0
4 1
100 100 100 100
0 0 0 0
1 0
50
10
3 1
25 25 25
25 25 25
```

Output for Sample Input

```
Case #1: 15000.000
Case #2: -15000.000
Case #3: 15000.000
Case #4: 12000.000
Case #5: -1875.000
```

Explanation for 1st sample case
Gori never comes up with his own solution, so he submits all your (correct) solutions.

Explanation for 2nd sample case
Give (only) one wrong solution and Gori will certainly end up with $ -15,000 balance.

Explanation for 3rd sample case
Gori is able to answer all questions correctly without your help.

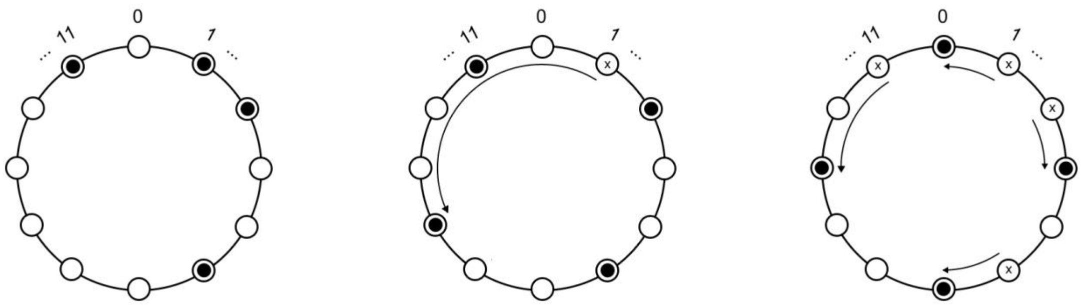
Explanation for 4th sample case
Gori has 90% chances of getting the correct answer.

Explanation for 5th sample case
Give a wrong solution to the first problem and correct solutions to the rest.

Problem I
Coins on a Ring

In this problem, you're going to work with a ring which has N equidistant slots numbered from 0 to $N - 1$ (refer to Figure 1.a). There are M coins which are located at some of those N slots on the ring (one slot may contain more than one coin). Your task is to move some of the coins if needed such that the distances between all adjacent coins are equal and the cost to achieve such configuration is as low as possible. The cost to achieve a configuration is equal to the maximum distance of any coin moved from its original position. Note that there must be no two or more coins which are located at a same slot in the final configuration.

For example, let there be a ring with 12 slots (0…11) and 4 coins at slot position 11, 1, 2 and 5 as shown in Figure 1(a).



(a) (b) (c)

Figure 1

One way to achieve a configuration where all adjacent coins are equidistant is shown in Figure 1(b):

- Move coin from slot 1 to slot 8 counter clockwise, cost = 5.

The cost to achieve this configuration is 5.

Another configuration is shown in Figure 1(c):

- Move coin from slot 1 to slot 0 counter clockwise, cost = 1.
- Move coin from slot 2 to slot 3 clockwise, cost = 1.
- Move coin from slot 5 to slot 6 clockwise, cost = 1.
- Move coin from slot 11 to slot 9 counter clockwise, cost = 2.

The cost to achieve this configuration is 2, which is better than the previous one. Actually, the minimum cost needed for this case is 2 and Figure1(c) shows one such example configuration.

Given a ring with N slots and M coins as described above, determine the minimum cost to move the coins (if needed) such that all adjacent coins are equidistant. You may assume M always divides N.

Input

The first line of input contains an integer T ($T \leq 100$) denoting the number of cases. Each case begins with two integers N ($2 \leq N \leq 1{,}000{,}000$) and M ($2 \leq M \leq 20{,}000$; M divides N) in a line which represent the number of slots and the number of coins on the ring respectively. The next line contains M integers ($0 \leq C_i < N$) which represent the position of each coin on the ring.

Output

For each case, output "`Case #X: Y`", where `X` is the case number starts from 1 and `Y` is the minimum cost needed to move all coins such that all adjacent coins are equidistant.

Sample Input	Output for Sample Input

`3`
`12 4`
`11 1 2 5`
`15 5`
`1 9 1 1 2`
`10 2`
`3 4`
	`Case #1: 2`
	`Case #2: 4`
	`Case #3: 2`

Explanation for 1st sample case
This sample corresponds to example in the problem statement.

Explanation for 2nd sample case
One configuration with the minimum cost:

- Move 1st coin (from slot 1) counter clockwise to slot 12, cost = 4.
- Move 3rd coin (from slot 1) counter clockwise to slot 0, cost = 1.
- Move 4th coin (from slot 1) clockwise to slot 3, cost = 2.
- Move 5th coin (from slot 2) clockwise to slot 6, cost = 4.

After these movements, the coins are located in slot 0, 3, 6, 9 and 12 which are equidistant to each other. The cost to achieve this configuration is 4.

Problem J

Alien Abduction Again

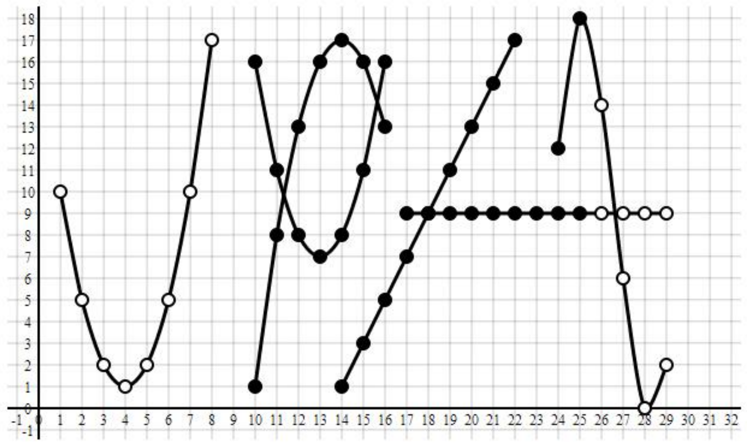
Last year there was an alien ship abducting people on Earth and returning them in bizarre locations. Some of them were returned in the middle of a desert, a jungle, an ocean, or a lake. Fortunately, our brilliant scientist (with some help from last year ICPC contestants) was able to quickly locate the positions in which the humans were returned and saved all the abducted humans. Nevertheless, the Earth government is still afraid that the alien ship will come back and do it again this year, or maybe even worse.

After analyzing last year report, our brilliant scientist now has a better knowledge on how the alien transporter technology works. For simplicity, let's assume our world is in one dimensional space (i.e., each person occupies space only in an interval of the x-axis). Depending on the person's orientation and posture, it requires different energy (at each integral point in the x-axis) to transport the person. To be precise, to transport a person p who lies on position $[x_1,\ x_2]$, it requires energy at each **integral** x position according to some function $f_p(x) = ax^3 + bx^2 + cx + d$.

Here is an example of 6 humans scattered on x-axis:

- Person #1 is at x = [1, 8] with $f_1(x) = x^2 - 8x + 17$
- Person #2 is at x = [10, 16] with $f_2(x) = -x^2 + 28x - 179$
- Person #3 is at x = [10, 16] with $f_3(x) = x^2 - 26x + 176$
- Person #4 is at x = [17, 29] with $f_4(x) = 9$
- Person #5 is at x = [14, 22] with $f_5(x) = 2x - 7$
- Person #6 is at x = [24, 29] with $f_6(x) = x^3 - 80x^2 + 2125x - 18732$

The graph on the right shows where on the x-axis each of the 6 person lies. The y-axis is the energy required for the transporter to operate. Note that there is no energy required for non-integral value of x. Each line (or curve) in the graph is a guideline that signifies a person.



When the alien ship performs the abduction on the human located at $[x_1,\ x_2]$, it requires energy equivalent to the total of energy at each integral x positions in between x_1 and x_2, inclusive. For example, if the alien ship tries to transport human at x = [10, 25], it would require 353 energy (i.e. the sum of the y-value of the black dots in the graph above). Observe that person #5 and person #6 are “partially” transported.

Since the transporter is operating at high energy, it creates a space distortion every time after it finished the abduction. The space distortion somehow equals to an imaginary human being registered to the system at $[\min(r_1, r_2),\ \max(r_1, r_2)]$ with some function $f_p(x)$ where $r_1 = (x_1 \cdot E)\ mod\ 10^6$, $r_2 = (x_2 \cdot E)\ mod\ 10^6$, and E is the energy required by the alien ship to perform the abduction to all humans at $[x_1,\ x_2]$ right before this imaginary human is registered. Note that this E is the non negative energy result after modulo 1,000,000,007 (see the output section).

To prevent mankind from being transported in the future, the brilliant scientist suggests us to emit negative energy with the same amount as soon as the transport operation is in progress.

Knowing this, the Earth government asks the brilliant scientist to build a device to disrupt the alien ship transport operation. However, in order to build this device, the scientist must be able to compute the total energy of a given range in the x-axis very quickly. Since the Earth is so large, even the brilliant scientist cannot compute the total energy by hand, he needs to create a program for this. Well... unsurprisingly, the brilliant scientist cannot program. Knowing that you are one of the best programmers on Earth, the scientist asks for your help again!

Note that your program must be very-very efficient, otherwise it will be too late to counter the transport operation and people will still be abducted!

Input

The first line of input contains an integer T ($T \leq 30$) denoting the number of cases. Each case starts with an integer N. Each of the next N lines contains one of the following command:

- p x_1 x_2 a b c d : a person is registered at position $[x_1, x_2]$ with $f(x) = ax^3 + bx^2 + cx^2 + d$.
- t x_1 x_2 a b c d : an abduction is in progress at position $[x_1, x_2]$. After the abduction, an imaginary person is registered (due to the space distortion) which equals to the command:
  p $\min(r_1, r_2)$ $\max(r_1, r_2)$ a b c d
  where
  - $r_1 = (x_1 \cdot E)\ mod\ 10^6$
  - $r_2 = (x_2 \cdot E)\ mod\ 10^6$
  - E is the non negative energy result after modulo 1,000,000,007.

The constraints are:

- $1 \leq N \leq 100{,}000$
- $0 \leq x_1 \leq x_2 < 1{,}000{,}000$
- a, b, c, d are all signed 32-bit integers.

Output

For each case, output "`Case #X:`" in a line, where `X` is the case number starts from 1. For each transport (abduction) operation in each case in the input, print the total energy used modulo 1,000,000,007. If the result is negative, add it by 1,000,000,007 to make it non negative. This output corresponds to E in any previous explanations.

Sample Input	Output for Sample Input
2 7 p 1 8 0 1 -8 17 p 10 16 0 -1 28 -179 p 10 16 0 1 -26 176 p 17 29 0 0 0 9 p 14 22 0 0 2 -27 p 24 29 1 -80 2125 -18732 t 10 25 0 0 0 0 6 p 4 10 3 4 5 -600 t 2 5 1 0 0 0 t 999000 999900 0 0 0 0 p 3 6 0 0 0 141 t 2 5 3 1 2 5 t 0 999999 0 0 1 2	Case #1: 353 Case #2: 999999583 85385243 1000000006 67775098