2^{O} Aquecimento para Nacional - Foz

23 de Setembro de 2017

Sevidor BOCA:
`http://maratona.c3sl.ufpr.br/`

Instruções Importantes

- A entrada é feita pela entrada padrão (*standard input*), enquanto a saída é feita pela saída padrão (*standard output*).
- Sua solução será compilada ou executada com a seguinte linha de comando:
  - C: `gcc -static -O2 -lm`
  - C++: `g++ -static -O2 -lm`
  - C++11: `g++ -std=c++11 -static -O2 -lm`
  - Java: `javac`
  - Python: `python3`
  - Pascal: `fpc -Xt -XS -O2`
- Sua solução deve processar cada arquivo de entrada no tempo máximo estipulado para cada problema, dado pela seguinte tabela:

Problema	**Nome**	**Tempo Limite (segundos)**
A	Advertising Revenue	1
B	Voter Depression	1
C	Tracking Fake News	1
D	Creating Fake News	1
E	Bot Or Not	1
F	Nothing But The Truth	1

- Todas as linhas, tanto na entrada quanto na saída, terminam com o caractere de fim-de-linha ($\backslash n$), mesmo quando houver apenas uma única linha no arquivo.
- Para submissões em **JAVA**, a classe deverá ter o mesmo nome que o *basename* do problema (leia a linha entre o título e o texto do problema).

A: Advertising Revenue

`Arquivo: revenue.[c|cpp|java|py|pas]`

One of the motivations for producing fake news stories is money, obtained from ads that are shown to or clicked on by the site's visitors. Mysterious, surprising, or scandalous stories tend to attract a lot of readers; hence the often outrageous and vague nature of fake news stories ("You won't believe..."). Online ads typically come in two forms: display ads and clickthrough ads. For the former, the advertiser will pay the site money just for displaying the ad (the idea is to build brand recognition); for the latter, the advertiser only pays when you click on the ad (the idea is that some fraction of clicks will lead to purchases, now or in the future). Prices for online ads range from a few cents per display/click to as much as $ 100 per click[1]. In this problem, you will be presented with a list of ads (clickthrough and display) and a history of visitors, what they saw, and what they clicked. Your goal is to calculate how much money the fake news website made.

Input

The first line contains a number $K \geq 1$, which is the number of input data sets . This is followed by K data sets of the following form:

The first line of each data set contains two integers n, v. $2 \leq n \leq 1000$ is the number of ads the site has as customers, and $0 \leq v \leq 1000$ is the number of visitors to the site.

This is followed by n lines, each describing one ad i with two integers d_i, p_i. The number d_i will always be either 0 or 1; a value of 1 means that the ad is a display ad (the site gets paid just for displaying it), and a value of 0 means that the ad is a clickthrough ad (the site only gets paid when the user clicks on it). $1 \leq p_i \leq 1000$ is the payment for the ad, which will be paid for displaying it (display ad) or clicking it (clickthrough ad).

Next come v lines, each describing one visitor j with three integers $a_{j,1}, a_{j,2}, c_j$. The first two numbers $a_{j,1}, a_{j,2}$ are the two ads the user is shown when visiting the site; they will be distinct and will always be between 1 and n, inclusive. The number $c_j \in \{0, 1, 2\}$ is the ad the user clicked on. A value of 0 means the user did not click on anything, 1 means that the user clicked on $a_{j,1}$, and 2 means that he/she clicked on $a_{j,2}$. Notice that a user might click on a display ad – the site does not get extra payment for this.

Output

For each data set, first output `Data Set x:` on a line by itself, where `x` is its number. Then, output the total payment that the fake news site receives.

Each data set should be followed by a blank line.

[1] Class-action lawyers

Sample Input	**Sample Output**
2 3 3 1 10 1 15 0 100 1 2 0 2 3 2 1 3 0 2 3 1 9 0 13 1 2 0 1 2 1 1 2 2	Data Set 1: 150 Data Set 2: 40

B: Voter Depression

`Arquivo: depress.[c|cpp|java|py|pas]`

Fake news is a powerful tool in the context of elections. Typically, the goal in disinformation campaigns is not so much to increase the vote for one's own candidate as to depress the vote for the other side, getting those voters to stay home. The usual approach is to target specific populations with negative messages about their candidate that will exploit their own biases. For example, if one wanted to harm a Democratic presidential candidate, one could plant targeted "news" stories for the left-leaning wing that appear to confirm pre-conceived notions that the candidate is a war-mongering corporate shill; simultaneously, one could plant "news" stories for the centrist wing that report on the candidate's socialist leanings.

We model this problem as follows: the voters are located at real-valued locations x_j (excluding 0), corresponding to the left–right political spectrum. Voters with negative x_j will vote for the left candidate, and voters with positive x_j for the right candidate. Each voter has an initial propensity to vote p_j. Each fake news story i is characterized by an interval $[l_i, r_i]$ and a factor $0 \leq d_i \leq 1$. This means that for each voter located in the interval $[l_i, r_i]$, his/her propensity to vote if exposed to the story changes from p_j to $p_j \times d_i$. The campaign must select a set of fake news story to deploy such that no voter is exposed to more than one story. The goal is to maximize the difference between the final sum of propensities of people voting for the right candidate minus the sum of propensities of people voting for the left candidate.

Input

The first line contains a number $K \geq 1$, which is the number of input data sets. This is followed by K data sets of the following form:

The first line of the data set contains two integers $1 \leq n \leq 200$ (the number of voters) and $1 \leq m \leq 50$ (the number of fake news stories you have at your disposal).

This is followed by n lines, each describing a voter j with two floating point numbers x_j, p_j with $-1 \leq x_j \leq 1$ and $0 \leq p_j \leq 1$. Voters will be sorted by non-decreasing x_j.

Next come m lines, each describing a fake news story i with three floating point numbers l_i, r_i, d_i, satisfying $-1 \leq l_i \leq r_i \leq 1$ and $0 \leq d_i \leq 1$. The news stories will be sorted by non-decreasing r_i values. To avoid floating point problems, we will guarantee that none of the x_j will be equal to any l_i or r_i.

Output

For each data set, first output `Data Set x:` on a line by itself, where `x` is its number. Then, output the maximum difference you can achieve between the total propensity of people voting for the right candidate, minus the total propensity of people voting for the left candidate, rounded to two decimals.

Each data set should be followed by a blank line.

Sample Input	**Sample Output**
1 5 4 -0.96 0.35 -0.5 0.2 -0.2 0.8 0.3 0.4 0.8 0.8 -0.6 -0.4 0.95 -1 0 0.8 -0.3 0.4 0.5 -0.98 0.9 0	Data Set 1: 0.12

C: Tracking Fake News

`Arquivo: tracking.[c|cpp|java|py|pas]`

When fake news spreads, it often uses social networks as its conduit. A friend posts a story that agrees with your pre-conceived ideas and prejudices (e.g., the presidential candidate of the other party participated in some nefarious activity that you can easily ascribe to them), and you will believe it much more uncritically than if it disagrees with your preferences. As a result, you will post the story, and your friends will see it, etc. A sufficiently juicy and pandering piece of "news" can spread like wildfire. In this problem, you will simulate the spread of a piece of fake news, when you know the social network and the proclivities of the individuals, as well as where in the network the piece of fake news originated.

To keep our model simple (if a bit unrealistic), we will assume that there are a number of different categories i that matter to you, like which political side the story reflects positively on, how much sex it contains, how much violence it contains, etc. For each of those categories, each individual has an integer (positive or negative) number $w_{j,i}$ describing his/her weight for it. Each individual also has a target t_j for total content. Each story also has an integer content c_i for each category. Individual j likes a story exactly iff $\sum_i w_{j,i}c_i = t_j$. (This is a very stringent requirement, but easy to compute.) If they like it, they will repost, otherwise not. We assume that each person sees all stories posted by their friends, and no stories posted by their non-friends (unless friends repost them).

Input

The first line contains a number $K \geq 1$, which is the number of input data sets . This is followed by K data sets of the following form:

The first line of a data set contains two integers n, r; $1 \leq n \leq 1000$ is the number of individuals in the social network, and $1 \leq r \leq 100$ is the number of categories that matter for stories. The next line is a single line containing exactly r integers, between -100 and 100, inclusive; these are the c_i.

Next come n lines, each describing one individual j in the network. The first r numbers on the line are the $w_{j,i}$; they are integers between -10 and 10, inclusive. Next comes the integer t_j , between -1000000 and 1000000, inclusive. After that comes one integer d_j with $0 \leq d_j \leq n - 1$, which gives you the number of friends of j. This is followed by d_j distinct integers, all between 1 and n, inclusive, and not equal to j. This is the list of j's friends. We will always ensure that if j' is a friend of j, then j will also be a friend of j' .

The fake news story will always start at person 1, though he/she may not even post it if it doesn't appeal to them.

Output

For each data set, first output `Data Set x:` on a line by itself, where `x` is its number. Then, output the total number of people who have posted the fake news story by the time no one new posts.

Each data set should be followed by a blank line.

Sample Input	**Sample Output**
1 5 3 1 -3 7 3 2 1 4 2 2 3 0 0 0 0 3 4 1 3 4 0 0 3 3 2 1 5 -2 1 1 1 2 2 5 10 -1 2 27 2 3 4	Data Set 1: 2

D: Creating Fake News

`Arquivo: creating.[c|cpp|java|py|pas]`

Now that we understand (from the previous problem) how fake news spreads through a social network, we can try to leverage this knowledge to create the perfect piece of fake news to reach everyone in the social network. For the context of this problem, designing a piece of news means choosing the best c_i , which are now allowed to be fractional numbers.

The model will be exactly the same as in the previous problem (so make sure you have at least read that problem), with two exceptions: (1) The content description of the news story you create (the c_i) can be floating point numbers now. (2) The fake news piece can start at multiple individuals, which you (the fake news spammer) get to choose. Note that it must be the *same* piece of fake news that you start at all these individuals.

In some settings, it will be impossible to create a single piece of fake news that everyone will like; you should note when that happens. Otherwise, output the smallest number of individuals you need to start your fake news piece at to reach everyone.

Input

The first line is the number K of input data sets, followed by K data sets, each of the following form:

The input format is the same as for the previous problem, except you will not be given a line describing a piece of fake news (since you are supposed to create it).

Output

For each data set, output `Data Set x:` on a line by itself, where `x` is its number. Then, on a line by itself, output the minimum number of people you need to start your fake news at to reach everyone. If it is impossible to reach everyone, output `Impossible` instead.

Each data set should be followed by a blank line.

Sample Input	**Sample Output**
2 6 3 3 2 1 6 2 2 3 0 0 0 0 3 4 1 3 4 0 0 4 3 2 1 5 -2 1 1 1 2 2 5 9 -1 2 15 2 3 4 0 0 0 0 0 3 2 1 1 1 2 2 3 1 0 1 2 3 1 0 1 1 2 2 1	Data Set 1: 2 Data Set 2: Impossible

E: Bot Or Not

`Arquivo: botornot.[c|cpp|java|py|pas]`

Fake news items are often spread by bots on Twitter and related sites. Fake news bots often try to be mistaken for a human, so that their endorsements of (often fake) news stories can generate traffic for the fake news site. In order to look like human users, bots follow individuals and repost their stories. Twitter would really like to be able to identify and remove such bots. In this problem, you will identify bots by using the fact that the posting patterns of bots differ from those of known humans.

You will be given some "primary" accounts that post stuff, and "secondary" accounts that repost stuff from the primary accounts. For each secondary account, you will be given which primary accounts it follows, as well as whether it is known to be human or unknown. You will also be given a list of all posts that it reposts. The assumption is that each secondary account sees all posts by all primary accounts it follows; so if it does not repost a post, this is a conscious (or bot-ish) decision.

We assume that human reposting patterns are somewhat similar, and bot patterns are different. Specifically, you will be given three integer numbers w_0, w_1, w_2 with $w_1 < 0 < w_0, w_2$. w_j is the similarity score between two accounts if j of them repost a particular post. So if they agree (both of them repost, or neither of them does), they get a positive match score; if they disagree, they get a negative match score w_1. We say that two accounts are "similar" if their total match score exceeds a given threshold t. We judge an account as human if it is similar to a known human account; note that being similar to an account that is similar to a known human does *not* make the account judged as human.

Input

The first line contains a number $K \geq 1$, which is the number of input data sets. This is followed by K data sets of the following form:

The first line of the data set contains six integers p, s, w_0, w_1, w_2, t. $1 \leq p \leq 100$ is the number of primary accounts, and $1 \leq s \leq 100$ the number of secondary accounts. $-100 \leq w_1 < 0 < w_0, w_2 \leq 100$ are the similarity scores, and $0 \leq t < 100000$ is the similarity threshold.

This is followed by a line with p integers $0 \leq m_i \leq 1000$; m_i is the number of posts that primary account i made. The posts are numbered so that the first m_1 posts come from account 1, the next m_2 posts from account 2, etc.

Next come s lines, each describing a secondary account j. Each line first contains an integer h_j which is either 0 (not known if human or bot) or 1 (known to be human). Next is an integer $0 \leq f_j \leq p$, the number of primary accounts that j follows. This is followed by f_j distinct integers between 1 and p, the indices of the primary accounts that j follows. Next is an integer $0 \leq r_j \leq \sum_i m_i$, the number of reposts made by account j. This is followed by r_j distinct integers between 1 and $\sum_i m_i$, the posts that account j reposted. Our input will never have j reposting posts by an account that he/she/it isn't following.

Output

For each data set, first output `Data Set x:` on a line by itself, where `x` is its number. Then, output the number of secondary accounts that are judged (or known) to be human.

Each data set should be followed by a blank line.

Sample Input	**Sample Output**
1 3 3 1 -4 5 8 5 3 6 0 1 2 3 8 7 6 1 2 1 3 6 1 9 10 13 14 5 0 2 2 3 7 6 8 13 14 9 7 12	Data Set 1: 2

F: Nothing But The Truth

`Arquivo: truth.[c|cpp|java|py|pas]`

When you are presented with a "news" story, it can be hard to tell whether it is true or fake, in particular if it has been designed to sound "believable" to you or someone with your biases. However, sometimes, you get lucky and know facts that contradict the story. Here, you will be given several facts about the whereabouts of several people at different times, and then a text that makes claims about where these people were and whom they met. You are supposed to count how many definitely false statements you can find in the text based on the knowledge we provided you.

The background knowledge we give you will be of the form that person X was at place Y from time t_1 until time t_2, for different persons, places, and times. (The same person, place, or time may of course appear multiple times; but we will never claim that a person was in two places at the same time.) When there is a time at which we didn't tell you where a person was, that person could have been anywhere. (It's something you don't know.)

The text will make claims about who met whom and who was where when. (It will also make many other claims, but you will ignore those.) Text consists of letters (upper case and lower case), spaces (including tabs and newlines), periods, and commas. Commas and all kinds of spaces separate words. Periods separate sentences. To simplify your parsing task (natural language is hard to parse), we follow the following rules:

- When the sentence contains the word `met`, then all people whose names are in the sentence are claimed to have met each other, though not necessarily at the same time or in the same place. For instance, the sentence `Flynn, Trump and Putin met.` claims that the pair (Flynn, Trump) met at some point, (Flynn, Putin) met at some (possibly different) point, and (Trump, Putin) met at some point.
- Any sentence that contains the phrase `was at` makes a claim about a location of a person at some time. The next word after this `at` is the claimed location. A sentence containing the phrase `was at` applies to any person whose name occurs in the sentence. The time period that is claimed is given by two phrases denoted with `from` and `until`. Each is followed by an integer (between 0 and 10000) which is the claimed time. We will guarantee that the `from` integer is never larger than the `until` integer. So `Until 50 Bernie was at Philadelphia Hillary from 30.` claims that Bernie and Hillary were both at Philadelphia from time 30 until time 50 (inclusive).
- No sentence contains both a `met` and a `was at`. No sentence contains more than one `was at`. Each sentence with a `was at` contains exactly one `from` and exactly one `until`.
- Case does not matter, so `Jill` and `jiLl` are the same person, and `met` and `mET` mean the same.
- No two people or places will have the same name, and no person and place will have the same name. No person or place will be named `met`, `from`, or any other word that carries special meaning for this problem.

The creators of the fake news story know enough to avoid internal contradictions (like claiming that Bernie was at Philadelphia from 30–50 and at Vermont from 20–30), so you don't need to look for them. Instead, you are supposed to count the false statements implied by the text. In the examples above, if Flynn met Trump, but neither Flynn nor Trump met Putin, that would be two false statements. If neither Bernie nor Hillary was at Philadelphia for the entire interval 30–50 (maybe Hillary was there from 30–45 and Bernie from 40–50), then that would also be two false statements.

We consider a claim that X met Y to be true if based on our background knowledge, the two were in the same place at the same time at least once. (For instance, if Flynn was at MarALago from 10 to 20 and Trump was at MarALago from 20 to 22, then they met.) If we have no such evidence, we consider the statement false. On the other hand, we consider a `was at` statement false only if we have evidence that the person was somewhere else for at least one time unit in the interval.

Input

The first line contains a number $K \geq 1$, which is the number of input data sets. This is followed by K data sets of the following form:

The first line of a data set contains four integers p, l, m, n. $1 \leq p \leq 20$ is the number of people, $1 \leq l \leq 20$ is the number of locations. $0 \leq m \leq 1000$ is the number of facts you are given, and $0 \leq n \leq 30$ is the number of lines of the news story.

This is followed by a single line with the names of the p people, each a string of 1–16 letters, separated by spaces and/or tabs. (Names have no spaces in them.) Next is a single line with the names of the l locations, each also a string of 1–16 letters, separated by spaces and/or tabs.

After that are m lines, each stating a fact i by giving four integers p_i, l_i, s_i, f_i. Here $1 \leq p_i \leq p$ is the person, $1 \leq l_i \leq l$ is the location, and $0 \leq s_i \leq f_i \leq 10000$ is the time interval when person p_i was at l_i . (From s_i until f_i, inclusive.)

Finally, there are n lines of text, as described above. Each line is at most 120 characters long.

Output

For each data set, first output `Data Set x:` on a line by itself, where `x` is its number. Then, on a line by itself, output the number of false statements made in the text.

Each data set should be followed by a blank line.

Sample Input

```
1
6 6 11 7
Trump Flynn Putin Hillary Bernie Jill
Philadelphia Moscow USC MarALago NYC washington
3 2 0 100
4 1 20 40
4 5 42 50
4 6 60 90
5 1 30 50
1 5 30 70
1 4 71 100
2 5 69 69
2 2 80 81
5 6 80 100
6 4 100 101
USC is the best university, much better than UCLA.
Trump and Flynn and Putin met at USC.
was at,philadelphia Bernie not at the same time from 20 HIllAry until 40.
Bernie, Hillary met Jill,Trump.
We hold these truths to be self-evident. Jill until 80,was at,nyc,FROM 3.
until 40 from 39, Government of the people, by the people, for the people
was a moscow shall not perish from the Earth putiN.
```

Sample Output

```
Data Set 1:
4
```