Southwestern Europe Regional Contest

November 17, 2013

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

DSIC
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

(This page is intentionally left blank)

Frank lives in London and he really likes games and maths. Lately, he has been playing a game on his mobile phone. It is simple, a sequence of coloured tokens is provided and each turn the player has to combine a pair of adjacent tokens to obtain a new token of certain colour. This process is repeated until the whole sequence results in only one final token. The problem is that not every pair of colours can be merged. There is a set of rules describing the allowed combinations. For example, given the following rules

blue	+	yellow	→	green
yellow	+	red	→	orange
blue	+	orange	→	brown

and the sequence (`blue`, `yellow`, `red`), the game could be finished obtaining a `brown` token after two steps: (`blue`, `yellow`, `red`) → (`blue`, `orange`) → (`brown`).

Frank is now in Valencia attending a famous programming contest and he is waiting for the tram to go to the university. Meanwhile, he is playing this game to fill time but the sun is shining so bright that he can not properly see the screen of his phone. He has some certainty about the possible colour of each token and he is wondering what will be the resulting colour following the most likely play of the game according to his estimation of the input sequence. Given Frank's estimation of two colours `A` and `B`, and the combination `A` + `B` → `C`, the certainty of the obtained colour `C` is computed as $\mathsf{cer}(\mathtt{C}) = \mathsf{cer}(\mathtt{A}) \cdot \mathsf{cer}(\mathtt{B})$.

INPUT

The first line contains R ($0 < R \leq 100$) the number of rules describing the allowed combinations of colours. Each of the following R lines contains three strings s_1, s_2, s_3 representing the combination $s_1 + s_2 \rightarrow s_3$. Next line shows the number of test cases T. For each test case an integer C indicates the length of the input sequence of tokens ($0 < C \leq 500$). Each of the next C lines describes a token, it contains a list of pairs (k, $\mathsf{cer}(\mathsf{k})$) providing a colour k and its corresponding certainty ($0 < \mathsf{cer}(\mathsf{k}) \leq 1.0$). The list ends with the word `END`. The sum of the certainties for a certain token always is 1.0. Every colour in a test case has appeared first in the rules describing the game.

OUTPUT

For each test case, print the resulting colour of the most likely play of the game. If there is no possible combination to finish the game then print `GAMEOVER`.

SAMPLE INPUT

```
7
Blue Yellow Green
Yellow Red Orange
Green Red Yellow
White Red Pink
Pink Black Red
Orange Red Red
Yellow Orange Yellow
3
2
Red 0.7 Orange 0.3 END
Yellow 1.0 END
4
Blue 0.6 Green 0.4 END
Red 0.2 Orange 0.6 Yellow 0.2 END
White 0.9 Yellow 0.1 END
Red 0.5 Black 0.5 END
2
Blue 1.0 END
Orange 1.0 END
```

SAMPLE OUTPUT

```
Orange
Yellow
GAMEOVER
```

EXPLANATION OF THE SAMPLES

First case

In the first case there are only two tokens. Frank is sure that the second token is `Yellow`, but the first token could be either `Red` or `Orange`. There are two possible solutions to the game:

1. (`Red`, `Yellow`) $\rightarrow$ (`Orange`) :

   cer(`Orange`) $= 0.7$

2. (`Orange`, `Yellow`) $\rightarrow$ (`Yellow`) :

   cer(`Yellow`) $= 0.3$

Hence, according to Frank's estimation the most likely play is the number 1 such that the final colour is `Orange`.

Second case

In the second case there are several tokens and estimations. Two possible solutions are:

1. `(Blue,Yellow,Yellow,Red)` $\rightarrow$ `(Blue,Yellow,Orange)` $\rightarrow$ `(Blue,Yellow)` $\rightarrow$ `(Green)`

   cer(`Green`) $= 0.006$

2. `(Green,Red,White,Black)` $\rightarrow$ `(Green,Pink,Black)` $\rightarrow$ `(Green,Red)` $\rightarrow$ `(Yellow)`

   cer(`Yellow`) $= 0.036$

Hence, according to Frank's estimation the most likely play is the number 2 such that the final colour is `Yellow`.

Third case

In this final case Frank is sure that there are two tokens `Blue` and `Orange`. There is no rule in the defined game that allows the combination of those colours, hence, there is no solution for this sequence of tokens.

(This page is intentionally left blank)

It Can Be Arranged

Problem B

Every year, several universities arrange inter-university national programming contests. ACM ICPC Dhaka site regional competition is held every year in Dhaka and one or two teams are chosen for ACM ICPC World Finals.

By observing these, MMR (Mission Maker Rahman) has made a plan to open a programming school. In that school, N courses are taught. Each course is taught every day (otherwise, programmers may forget DP while learning computational geometry!). You will be given the starting time A_i and finishing time B_i (inclusive) of each course i ($1 \leq i \leq N$). You will be also given the number of students registered for each course, S_i ($1 \leq i \leq N$). You can safely assume no student has registered to two different courses. MMR wants to hire some rooms of a building, named *Sentinel Tower*, for running that school. Each room of Sentinel Tower has a capacity to hold as much as M students. The programmers (students) are very restless and a little bit filthy! As a result, when $\texttt{course}_i$ is taken in a class room, after the class is finished, it takes $\texttt{clean}_{ij}$ time to clean the room to make it tidy for starting teaching $\texttt{course}_j$ immediately just after $\texttt{course}_i$ in the same room.

Your job is to help MMR to decide the minimum number of rooms need to be hired to run the programming school.

INPUT

Input starts with an integer T ($T \leq 100$) denoting the number of test cases. Each case starts with two integers N ($1 \leq N \leq 100$), number of courses and M ($1 \leq M \leq 10000$), capacity of a room. Next N lines will contain three integers A_i, B_i ($0 \leq A_i \leq B_i \leq 10000000$) and S_i ($1 \leq S_i \leq 10000$), starting and finishing time of a course. Next N lines will contain the clean time matrix, where the i^{th} row will contain N integers $\texttt{clean}_{ij}$ ($1 \leq i \leq N$, $1 \leq j \leq N$, $0 \leq \texttt{clean}_{ij} \leq 10000000$, $\texttt{clean}_{ii} = 0$).

OUTPUT

For each case, print the test case number, starting from 1, and the answer, minimum number of rooms needed to be hired.

SAMPLE INPUT

```
3
1 5
1 60 12
0
4 1
1 100 10
50 130 3
150 200 15
80 170 7
0 2 3 4
5 0 7 8
9 10 0 12
13 14 15 0
2 1
1 10 1
12 20 1
0 2
5 0
```

SAMPLE OUTPUT

```
Case 1: 3
Case 2: 22
Case 3: 2
```

(This page is intentionally left blank)

We want to create a smartphone application to help visitors of a shopping mall and you have to calculate the shortest path between pairs of locations in the mall. Given the current location of the visitor and his destination, the application will show the shortest walking path (in meters) to arrive to the destination.

The mall has N places in several floors connected by walking paths, lifts, stairs and escalators (automated stairs). Note that the shortest path in meters may involve using an escalator in the opposite direction. We only want to count the distance that the visitor has walked so each type of movement between places has a different cost in meters:

- If *walking* or taking the *stairs* the distance is the euclidean distance between the points.
- Using the *lift* has a cost of 1 meter because once we enter the lift we do not walk at all. One lift can only connect 2 points. An actual lift connects the same point of different floors, in the map all the points connected by a lift have the corresponding edge. So you do not need to worry about that. For instance, if there are three floors and one lift at position (1,2) of each floor, the input contains the edges $(0, 1, 2) \rightarrow (1, 1, 2)$, $(1, 1, 2) \rightarrow (2, 1, 2)$ and $(0, 1, 2) \rightarrow (2, 1, 2)$. In some maps it can be possible that a lift does not connect all the floors, then some of the edges will not be in the input.
- The *escalator* has two uses:
  - Moving from `A` to `B` (proper direction) the cost is 1 meter because we only walk a few steps and then the escalator moves us.
  - Moving from `B` to `A` (opposite direction) has a cost of the euclidean distance between `B` and `A` multiplied by a factor of 3.

The shortest walking path must use only these connections. All the places are connected to each other by at least one path.

INPUT

Each input file contains the map of a unique shopping mall and a list of queries.

The first line contains two integers N ($N \leq 200$) and M ($N-1 \leq M \leq 1000$), the number of places and connections respectively. The places are numbered from 0 to *N-1*. The next N lines contain floor and the coordinates x, y of the places, one place per line. The distance between floors is 5 meters. The other two coordinates x and y are expressed in meters.

The next M lines contain the direct connections between places. Each connection is defined by the identifier of both places and the type of movement (one of the following: `walking`, `stairs`, `lift`, or `escalator`). Check the cost of each type in the description above. The type for places in the same floor is *walking*.

The next line contains an integer Q ($1 \leq Q \leq 1000$) that represents the number of queries that follow. The next Q lines contain two places each `a` and `b`. We want the shortest walking path distance to go from `a` to `b`.

OUTPUT

For each query write a line with the shortest path in walked meters from the origin to the destination, with each place separated by a space.

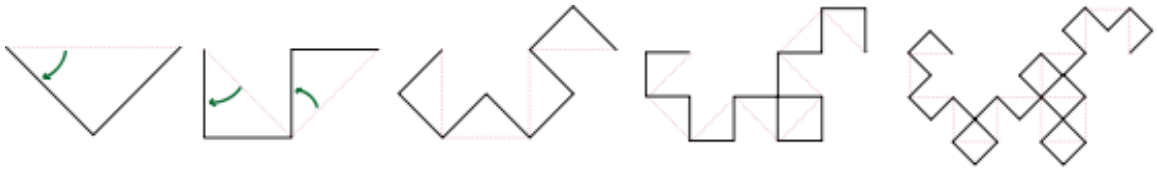
SAMPLE INPUT

```
6 7
3 2 3
3 5 3
2 2 3
2 6 4
1 1 3
1 4 2
0 1 walking
0 2 lift
1 2 stairs
2 3 walking
3 4 escalator
5 3 escalator
4 5 walking
5
0 1
1 2
3 5
5 3
5 1
```

SAMPLE OUTPUT

```
0 1
1 0 2
3 4 5
5 3
5 3 2 0 1
```

Edward is now 21 years old. He has to appear in an exam to renew his *State Alchemist* title. This year the exam is arranged in a bit different way. There will be a long hallway. Each alchemist will enter the hallway from the left side and come out from the right side and he has to do this **n** times. During this tour they have to bend the hallway segments right-left alternatively. Let's describe the process in some pictures:



- First time (First picture): Initially, the hallway is a straight line (soft line in the first picture). So alchemist will bend this segment to right side (he is going from left to right) like the hard line in the first picture above.
- Second time (Second picture): Now he will find two segments in hallway (like soft line in picture). So he will bend the first hallway to right, second one to left (like the hard lines).
- Third time (Third picture): Now he will find four segments in the hallway (like the soft lines) and he will bend them to Right, Left, Right and Left respectively.
- And this goes on for fourth and fifth times in the picture.

Since Full Metal Alchemist Edward is so good, he did it perfectly. Now it is turn of the judges to check the bending if it is correct or not. The judge enters at the left end and comes out from the right end. But during his travel he notes down the turning, R for Right and L for Left. So if $\mathbf{n} = \mathbf{1}$, then the judge would have noted down **L**. If $\mathbf{n} = \mathbf{2}$, it would have been **LLR**. For $\mathbf{n} = \mathbf{4}$, it would have been: **LLRLLRRLLLRRLRR**.

Since this string will grow exponentially with **n**, it will be tough to check whether the bending is correct or not. So the judges have some pre-generated strings and they know whether this string will appear as substring in the final string or not. Unfortunately the judges have lost the answer sheet, can you help them to recover it?

INPUT

First line of the test file contains a positive integer **T** denoting number of test cases (**T** <= **105**). Hence follows **T** lines, each containing an integer and a string: **n S**. **n** is the number of times Edward has passed through the hallway; and **S** is the string the judge is going to check with. You may assume that **S** consists of only the letters **L** and **R**. (**n** <= **1000**, length of **S** <= **100**). Also you may assume that length of **S** will not be greater than the length of the string for **n**.

OUTPUT

For each test case output the case number and **Yes** or **No** denoting whether the string is in the final string as substring.

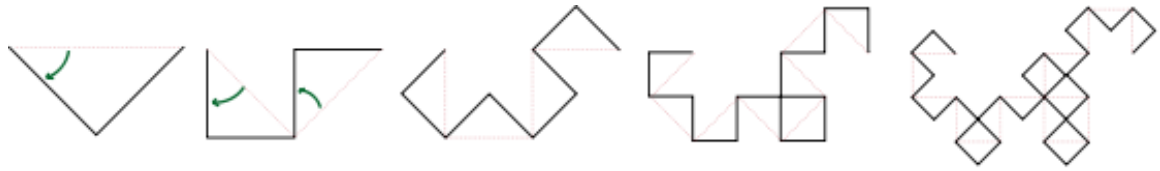
SAMPLE INPUT

```
2
1 R
4 LRRLL
```

SAMPLE OUTPUT

```
Case 1: No
Case 2: Yes
```

Edward is now 21 years old. He has to appear in an exam to renew his "State Alchemist" title. This year the exam is arranged in a bit different way. There will be a long hallway. Each alchemist will enter the hallway from the left side and come out from the right side and he has to do this **n** times. During this tour they have to bend the hallway segments right-left alternatively. Let's describe the process in some pictures:



- First time (First picture): Initially, the hallway is a straight line (soft line in the first picture). So alchemist will bend this segment to right side (he is going from left to right) like the hard line in the first picture above.
- Second time (Second picture): Now he will find two segments in hallway (like soft line in picture). So he will bend the first hallway to right, second one to left (like the hard lines).
- Third time (Third picture): Now he will find four segments in the hallway (like the soft lines) and he will bend them to Right, Left, Right and Left respectively.
- And this goes on for fourth and fifth times in the picture.

Since Full Metal Alchemist Edward is so good, he did it perfectly. Now it is turn of the judges to check the bending if it is correct or not. The judge enters at the left end and comes out from the right end. But during his travel he notes down the turning, R for Right and L for Left. So if **n = 1**, then the judge would have noted down **L**. If **n = 2**, it would have been **LLR**. For **n = 4**, it would have been: **LLRLLRRLLLRRLRR**.

Since this string will grow exponentially with **n**, it will be tough to check whether the bending is correct or not. So the judges have some pre-generated strings and they know whether this string will appear as substring in the final string or not. Unfortunately the judges have lost the answer sheet, can you help them to recover it?

INPUT

First line of the test file contains a positive integer **T** denoting number of test cases (**T <= 105**). Hence follows **T** lines, each containing an integer and a string: **n S**. **n** is the number of times Edward has passed through the hallway; and **S** is the string the judge is going to check with. You may assume that **S** consists of only the letters **L** and **R**. (**n <= 1000**, length of **S <= 100**). Also you may assume that length of **S** will not be greater than the length of the string for **n**.

OUTPUT

For each test case output the case number and **Yes** or **No** denoting whether the string is in the final string as substring.

SAMPLE INPUT

```
2
1 R
4 LRRLL
```

SAMPLE OUTPUT

```
Case 1: No
Case 2: Yes
```

Joe is learning to speak

Problem E

Joe is a 4-year-old child learning to speak his mother tongue. He tried to memorize every possible sentence in the language but he soon realized that the number of different sentences is unbounded. He decided to concentrate on memorizing the subsequences of size up to n and their meaning. Joe *understands* a sentence if he has previously memorized all its subsequences of size $\leq n$.

Every afternoon, Joe starts reading sentences one by one. He remembers everything he has learned in the previous days. After reading a sentence, he asks for every word he doesn't know and learns it. If he can't fully understand the sentence he asks for the meaning of the whole sequence. After this, he memorizes every subsequence of size up to n and reads the next sentence.

For example: supose that Joe memorizes only subsequences of 1 or 2 words and he already knows the sentences *"I live in a house"* and *"This is a green house"*. He will fully understand the sentence *"I live in a green house"*, since he already knows all the subsequences of size 1 and 2. However, if he decided to memorize subsequences up to size 3, he will ask for the meaning of the whole sentence because *"in a green"* is not a subsequence of size 3 in any previous sentence.

Obviously, Joe always knows his name.

INPUT

Input consists of several cases, each case begins with a line indicating the maximum size n ($2 \leq n \leq 5$) that Joe can memorize.

The next lines contain the sentences with the previous knowledge Joe memorized in the previous days. The character * indicates the end of the previous knowledge. Below the *, there is a text containing several sentences that Joe is going to learn today. The symbol # indicates the end of the sentences that Joe is going to read today.

Both sections have up to 20000 different words. The sentences are separated with any of the following characters that will appear as a separate token in the input: . , : ; ? !

Each word has size at most 20. The maximum number of words per sentence is 100. The maximum number of sentences per section is 1000. Sentences may appear more than once.

All the words contain only roman letters, no numbers or other symbols are used except the sentence separators. Words that differ only on the case of the letters are considered the same. For example: *Joe*, *JOE* and *joE* represent the same word.

OUTPUT

For each test case, a line with the text *Learning case* C where C is the number of the case starting by 1. Outputs for different test cases should be separated by a blank line.

For every unknown word W he asks for the meaning with the following question: *What does the word "W" mean?* For every sentence S not understood by Joe, he asks *What does the sentence "S" mean?* Output both W and S the same case as they appear. The words in S must be separated with exactly 1 space.

SAMPLE INPUT

```
2
hello sam . say hello to everybody . good bye .
*
say hi to everybody .
hi to everybody .
hello joe .
swerc
#
4
Joe is my name . I have a dog . My dog is
a
cinnamon coloured golden retriever
. The name of my father is Ben and the name
of my mother is Linda .
*
Hello   Joe , how   are   you ?  Are  you   my    friend ?
Last day you told me : the name of my mother is Linda .
You also told me : the name of my father is Ben .
  How  are   you   my   friend ?
#
```

SAMPLE OUTPUT

```
Learning case 1
What does the word "hi" mean?
What does the sentence "say hi to everybody" mean?
What does the sentence "hello joe" mean?
What does the word "swerc" mean?

Learning case 2
What does the word "Hello" mean?
What does the sentence "Hello Joe" mean?
What does the word "how" mean?
What does the word "are" mean?
What does the word "you" mean?
What does the sentence "how are you" mean?
What does the word "friend" mean?
What does the sentence "Are you my friend" mean?
What does the word "Last" mean?
What does the word "day" mean?
What does the word "told" mean?
What does the word "me" mean?
What does the sentence "Last day you told me" mean?
What does the word "also" mean?
What does the sentence "You also told me" mean?
What does the sentence "How are you my friend" mean?
```

In mathematics, the factorial of a positive integer number n is written as $n!$ and is defined as follows:

$$n! = 1 \times 2 \times 3 \times 4 \times \cdots \times (n-1) \times n = \prod_{i=1}^{n} i$$

The value of 0! is considered as 1. $n!$ grows very rapidly with the increase of n. Some values of $n!$ are:

$0! = 1$	$5! = 120$
$1! = 1$	$10! = 3628800$
$2! = 2$	$14! = 87178291200$
$3! = 6$	$18! = 6402373705728000$
$4! = 24$	$22! = 1124000727777607680000$

You can see that for some values of n, $n!$ has odd number of trailing zeroes (eg 5!, 18!) and for some values of n, $n!$ has even number of trailing zeroes (eg 0!, 10!, 22!). Given the value of n, your job is to find how many of the values 0!, 1!, 2!, 3!, ... , $(n-1)!$, $n!$ has even number of trailing zeroes.

INPUT

Input file contains at most 1000 lines of input. Each line contains an integer $n (0 \leq n \leq 10^{18})$. Input is terminated by a line containing a -1.

OUTPUT

For each line of input produce one line of output. This line contains an integer which denotes how many of the numbers 0!, 1!, 2!, 3!, ..., $n!$, contains even number of trailing zeroes.

SAMPLE INPUT

```
2
3
10
100
1000
2000
3000
10000
100000
200000
-1
```

SAMPLE OUTPUT

```
3
4
6
61
525
1050
1551
5050
50250
100126
```

(This page is intentionally left blank)

Vivoparc is a zoological park located in Valencia. It has recently added a new area formed by a big plane savanna grassland divided into several enclosures.

Our purpose is to assign one animal of the 4 different species (lions, leopards, tigers, and panthers) to each of the new VivoParc enclosures. These animals are very territorial therefore we must be sure that no animal can see other animal of its own species from its enclosure. The Vivoparc manager has sent us a file with the visibility from the different enclosures and we have to assign one species to each enclosure. At the end of the process, all the enclosures must have an assigned species.

INPUT

The first line of the input file contains the number of enclosures ($N \leq 100$). Each of the following lines contain a visibility restriction: 1-3 means that animals in enclosure #1 can see animals in enclosure #3 and animals in enclosure #3 can see animals in enclosure #1. Note that the Vivoparc manager is not a very well organized person and therefore, some data appearing in the file may be redundant.

OUTPUT

The output file contains one of all the possible species assignation. The file consists of a line per enclosure and each line contains the number of the enclosure followed by the assigned species (1= Lion, 2= Leopard, 3= Tiger, 4= Panther). Enclosures assignation must appear in ascending order.

SAMPLE INPUT

```
8
1-2
3-1
4-5
4-8
1-7
1-4
7-1
2-4
1-8
6-7
2-3
1-5
1-6
7-6
7-8
2-5
7-1
3-4
5-6
7-8
```

SAMPLE OUTPUT

```
1 4
2 2
3 1
4 3
5 1
6 2
7 1
8 2
```

(This page is intentionally left blank)

Binary Tree is a tree data structure where each node has at most two children, usually they are distinguished as **left** and **right** child. And the node having the children are called parent of those children.

An instruction string is a string consisting of the letters *L*, *R* and *U*. *L* stands for *Left*, *R* for *Right* and *U* for *Up*. Meaning of these will be clear shortly.

One day I have drawn an infinitely large Binary Tree. In this tree each node has exactly two children (*left child* and *right child*) and each of them has a parent. For this problem, we will consider **the parent of the root is root itself**. I put a pen in the root and follow the instruction string **S**. That is, we look at the first character if it says *L* we go to *left child*, if it says *R* we go to *right child* and if it says *U* then to *the parent*. If we receive a *U* instruction at root we just end up at root since we assumed parent of the root is root itself.

Now we have another instruction string **T**. Starting from the node where we are after following the instruction string **S**, we will follow the instruction string **T**. But this time, if we wish we may skip any instruction in the string **T** (possibly discarding all of them). You have to tell me how many different nodes I can end up after following instruction string T (skipping as many instructions as I wish).

For example:

Suppose: **S** = **L** and **T** = **LU**. Our answer is 3. Following **S** we will end up at the left child of the root. Now, when we follow **T**, there may be 4 cases:

i Skipping all letters: we will be at the **same node** where we are.

ii Skipping L and following U: we will be at the **root**.

iii Following L and Skipping U: we will be at the **left child** of current node.

iv Following both L and U: we will be at the **same** node as in case *i*.

Since 3 different nodes we can end up after following **T**, the answer is 3.

INPUT

First line of the test file contains an integer **N** (<= 15) denoting number of test cases. Hence follow **N** test cases. Each test case consists of two non empty strings. First line will contain instruction string **S** and the second line will contain the instruction string **T**. You may assume that there will not be any letter other than L, R or U in these strings. Length of the strings will not be greater than 100000.

OUTPUT

For each test case print the case number followed by the number of nodes we can end up finally. Since the answer may be large, you have to give the answer modulo **21092013**.

SAMPLE INPUT

```
2
L
LU
L
L
```

SAMPLE OUTPUT

```
Case 1: 3
Case 2: 2
```

(This page is intentionally left blank)

Trending Topic

Problem I

Imagine you are in the hiring process for a company whose principal activity is the analysis of information in the Web. One of the tests consists in writing a program for maintaining up to date a set of trending topics. You will be hired depending on the efficiency of your solution.

They provide you with text from the most active blogs. The text is organised daily and you have to provide the sorted list of the N most frequent words during the last 7 days, when asked.

INPUT

Each input file contains one test case. The text corresponding to a day is delimited by tag `<text>`. Queries of top N words can appear between texts corresponding to two different days. A top N query appears as a tag like `<top 10 />`. In order to facilitate you the process of reading from input, the number always will be delimited by white spaces, as in the sample.

Notes:

- All words are composed only of lowercase letters of size at most 20.
- The maximum number of different words that can appear is 20000.
- The maximum number of words per day is 20000.
- Words of length less than four characters are considered of no interest.
- The number of days will be at most 1000.
- $1 \leq N \leq 20$

OUTPUT

The list of N most frequent words during the last 7 days must be shown given a query. Words must appear in decreasing order of frequency and in alphabetical order when equal frequency. **There must be shown all words whose counter of appearances is equal to the word at position *N*. Even if the amount of words to be shown exceeds *N*.**

SAMPLE INPUT

```
<text>
imagine you are in the hiring process of a company whose
main business is analyzing the information that appears
in the web
</text>

<text>
a simple test consists in writing a program for
maintaining up to date a set of trending topics
</text>

<text>
you will be hired depending on the efficiency of your solution
</text>

<top 5 />

<text>
they provide you with a file containing the text
corresponding to a highly active blog
</text>

<text>
the text is organized daily and you have to provide the
sorted list of the n most frequent words during last week
when asked
</text>


<text>
each input file contains one test case the text corresponding
to a day is delimited by tag text
</text>

<text>
the query of top n words can appear between texts corresponding
to two different days
</text>

<top 3 />

<text>
blah blah blah blah blah blah blah blah blah
please please please
</text>

<top 3 />
```

SAMPLE OUTPUT

```
<top 5>
analyzing 1
appears 1
business 1
company 1
consists 1
date 1
depending 1
efficiency 1
hired 1
hiring 1
imagine 1
information 1
main 1
maintaining 1
process 1
program 1
simple 1
solution 1
test 1
that 1
topics 1
trending 1
whose 1
will 1
writing 1
your 1
</top>
<top 3>
text 4
corresponding 3
file 2
provide 2
test 2
words 2
</top>
<top 3>
blah 9
text 4
corresponding 3
please 3
</top>
```

(This page is intentionally left blank)

Cleaning the Hallway

Problem J

We have **N** persons and **N** vacuum cleaners. How much area can we clean? If you are thinking- "Isn't it a silly question? We can clean the entire hallway!" then you are wrong! You are forgetting that we need some electricity point in the floor to work with the vacuum cleaner. So it might not be possible to cover entire ground. Again the electrical wire of a vacuum cleaner is limited one as well. If a vacuum cleaner has wire of **D** units long, then the person using cleaner can go at most **D** unit distance away from the electrical point. But to make things worse, the cleaner have to be exactly **D** unit distance away. There is some glitch in the wire, so if the wire becomes loose at any moment (becomes less than **D** unit distance from the electrical point), the vacuum cleaner loses electricity. On top of this, the handle of the vacuum cleaner is too heavy, so the person cannot move the cleaner handle more than **d** unit distance from him. So to sum up, each cleaner has to be exactly **D** unit distance from the electric point and it can clean everything within **d** unit distance from it.

Now for each vacuum cleaner you are given **D** and **d**. Also you are given the co-ordinate of the electrical points where the vacuum cleaner is attached at. Find out the area of the ground that can be cleaned with the given setup. Please note that, some area can be covered by multiple persons but we are interested in the union of the area not sum of the area covered by individuals.

INPUT

In the first line of the input file number of test cases are given, **T** ($\leq$ **30**). Hence follow **T** test cases. Each test case starts with a positive integer **N** ($\leq$ **500**). In the next **N** lines you will be given description for the vacuum cleaners. Each line will contain, **4** integers: **x y D d**. **(x, y)** is the co-ordinate of the electrical point, **D** is the wire length and **d** is the distance of vacuum head from the person. ($|\mathbf{x}|$, $|\mathbf{y}| \leq$ **1000**, **0** $<$ **D**, **d** $\leq$ **200**).

OUTPUT

For each case print the case number and the area of the ground that can be covered. Error up to 10^{-2} will be ignored.

SAMPLE INPUT

```
2
1
0 0 10 1
2
0 0 7 1
13 0 10 1
```

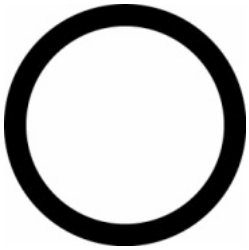
SAMPLE OUTPUT

```
Case 1: 125.663706
Case 2: 205.474931
```

EXPLANATION OF THE SAMPLES

First Case

Only one cleaner. Electrical point is at **(0, 0)** and $\mathbf{D} = \mathbf{10}$, $\mathbf{d} = \mathbf{1}$. So the person handling this cleaner will be moving around a circle with **10** unit radius centering at **(0, 0)**. Since $\mathbf{d} = \mathbf{1}$, the cleaner will be able to clean any dirt with in **1** unit distance from him. The region cleaned by the cleaner is denoted by black region in the following picture:



Here the black region is bounded by two circles both centering at co-ordinate **(0, 0)**. Radii of the circles are **11** and **9**. So the area covered $= \pi \times (11^2 - 9^2) =$ **125.663706**.

Second Case

Two cleaners. Electrical point of first cleaner is at **(0, 0)** and the point for second cleaner is at **(13, 0)**. For first cleaner $\mathbf{D} = \mathbf{7}$, $\mathbf{d} = \mathbf{1}$ while for second cleaner $\mathbf{D} = \mathbf{10}$, $\mathbf{d} = \mathbf{1}$. The region cleaned by the cleaners is denoted by black region in the following picture: