ACM ICPC - 7th Sharif Internet Contest & Internal Selection Contest of Sharif

30 Mehr 1388 - 22 Oct. 2009

Contents

Note on all the problems:

Your codes must read from standard input (`stdin`) and write to standard output (`stdout`).

Problem A: **AMaTeuR HaCKeRS!**

HaCKeR are different from crackers in their ethics. They are not always breaking the law. In contrast, they are interested in showing they are *cool*! One of their cool-ness paradigms is to write words in HaCKeR'S STYLe.

In HaCKeR'S STYLe, the first letter of a word is written in uppercase. But for the other letters (2nd, 3rd and so.) vowels (a, e, i, o, u) must be written in lowercase and CONSONANTS in uppercase.

For example, "PooR", "BReaK", "APPLe" and "KiLL" are all in HaCKeR'S STYLe. But none of the words "BaNaNA", "oiL", "MUSTaCHe" or "TRaINS" are in our desired style!

As a recruiter person of an Underground Hacking Group, you are asked to qualify new hackers. The procedure is that you ask any of them to write a full sentence and then you count the number of words they have written correctly in your style!

Input

First line of Input contains $\mathbf{1} \leq \boldsymbol{N} \leq \mathbf{100}$ that is number of participants. In the i-th line below that, the sentence from the i-th participant is written ($1 \leq i \leq N$).

A *sentence* is defined as a sequence of at least one and at most 100 words, separated by one single space between any two adjacent words. A *word* is a string of at least one and at most 20 uppercase or lowercase English letters.

Output

For each participant, write his grade in the format shown in the sample output. Write the grades right in the order they come in the input.

Sample Input	Sample Output
3 WeLCoMe to SHaRiF HaCK TeaM CAPSLOCK IS ON Polites Are Not Good Hackers Honey	4 out of 5. 2 out of 3. 0 out of 6.

Problem B: **Bahman's disapproval!**

Ali is a first year student in your university. In the first day of the new semester, he found that his student-identification-number (or shortly id), which was 88209689, is a prime number! He got happy and offered his friend Bahman, to have a tea-break together.

After a while, Bahman turned back to Ali and said "Hey bro., that's not a rare thing!". Ali asked "why?". Bahman replied "While the range of new students' id is $[88200000, 88209999]$ (inclusive), with a possibility of 6%, one's id is prime!"

Now Ali (who has not passed even one programming course!), asks you to find out the chance of being lucky in the university (i.e. having a prime id).

Input

First line of Input contains $\mathbf{1} \le \boldsymbol{T} \le \mathbf{100}$ that is number of the tests.

Then, in each of the following T lines, comes two 8-digit integers A and B. It's guaranteed that $A \le B$ and any of these two numbers are either starting with 882 or starting with 881.

Output

For each test, print the percentage that a random number in range $[A, B]$ is prime, in its own line. This percentage must be rounded to the nearest integer, followed by a percent sign (it is %).

Sample Input	Sample Output

3
88200000 88209999
88209688 88209690
88199999 88200000
	6%
	33%
	0%

Problem C: **Captivity of causality**

Saeed is a brilliant student of University. University was a nice place for him until problems started to show up. Some of the problems are too simple and can be easily solved; but there are some amusing problems that *even* Saeed is not able to solve them by hand.

These problems are in a wide range but they have one thing in common -- there is a method to describe them as logical expressions. Recently one of these problems (which you can see in the first sample input) has happened for him and after days of inspection and calculation, finally he made a list to describe his problem as logical expressions.

The list was a set of accurate information in the following format:

"*expression1* `because` *expression2* `.`" (quotes for clarity)

We know that no expression is direct cause of itself and every expression has at most one cause. Now Saeed wants to find the source of his problems. So you are to write a program for him.

Input

Input consist of several test cases.

First line of each test case contains ($0 \leq n \leq 50$) and ($1 \leq m \leq 50$) the number of information lines and the number of questions that Saeed has prepared for you, respectively. Each of next n lines contains one sentence per line. It is guaranteed that there is exactly one " **`because`** " in each sentence and at-least one non-space character in every expression. Next m lines contains a question in format "`Why` *expression*?" (Quotation marks are for clarity).

Input terminates with two zeros for n and m.

Output

For each question output one of these sentences:

"`No reason.`", "`Too complicated.`", "`Because` *expression*`.`" (Quotes are for clarity).

See sample input and sample output for more details. Print a blank line after each case.

Sample Input
```
8 1
Saeed should spend his whole next term studying physics because Saeed failed
to pass physics last term.
Saeed's TA could not remember Saeed's face because Saeed's TA never looked
at Saeed in TA classes.
Saeed failed to pass physics last term because Saeed's physics mark was less
than 10.
```

```
Saeed's physics mark was less than 10 because edu operator made a mistake
while adding Saeed's physics marks.
edu operator made a mistake while adding Saeed's physics marks because
Saeed's TA released Saeed's mark in the last moment.
Saeed's TA released Saeed's mark in the last moment because Saeed's TA was
suspicious about Saeed.
Saeed's TA was suspicious about Saeed because Saeed's TA could not remember
Saeed's face.
Saeed's TA never looked at Saeed in TA classes because there was another
nice student in Saeed's TA classes.
Why Saeed should spend his whole next term studying physics?

0 1
Why so serious?

4 5
a because b.
b because a.
c because b.
d because e.
Why a?
Why b?
Why c?
Why d?
Why e?

0 0
```

(Note: Long lines are wrapped. A complete line ends with a single dot.)

Sample Output

```
Because there was another nice student in Saeed's TA classes.

No reason.

Too complicated.
Too complicated.
Too complicated.
Because e.
No reason.
```

Problem D: **Dr. B and the pack of FANs**

Dr. B is head of department of CE of university and he is interested in FANs. A FAN is defined as set of smart tricks that makes the life of students harder and more unpleasant.

These FANs are infinitely various but after some researches, some students found some information about one kind of FANs which is called CSC (changing student's classes). Each term after the students select their classes, Dr. B starts to apply his FAN. This FAN allows him to change students classes in the following procedure:

1- He chooses a student id to apply his FAN.
2- He removes one of his/her classes. (optional)
3- He adds a new class instead of removed one.
4- Then he removes all the classes that conflict with the added class.
5- He adds a new class instead of every removed classes in the previous level.
6- While there is conflict a between classes repeat 4th and 5th level (step).

This FAN has some restrictions:

1- After applying this FAN, number of classes should be the same as number of classes before applying it.
2- Each class can be removed at most once.
3- Each class can be added at most once.
4- Dr.B can not remove a class that he added before.
5- Dr.B can not add a class that he removed before.

CSC_FANITY of a student is the maximum number of classes Dr. B can remove from his classes due to the rules and restrictions. In another words, CSC_FANITY is the number of classes that exist in the final schedule (after FANs of Dr. B) but were not selected in the initial schedule (prepare by the student).

Little Aideen is new student of university and he is afraid of FANs. Your task is to write a program to compute *CSC_FANITY* of Aideen.

Notice that Dr. B can apply his FAN only once for each student and you have to find out the FAN which brings maximum sadness to the student!

Input

Input consists of several test cases.

first line of each test case comes $(1 \leq n \leq 15)$ and $(1 \leq m \leq n, m \leq 5)$ being number of all classes (labeled from 1 to n) and number of classes that Aideen choose for his term.

Next line consists of m integers being label of Aideen classes.

Next n lines contains 2 numbers s_i and f_i that shows start time and finish time of class number i. These numbers show that class number i starts in first second of s_ith hour of week and ends in last second of f_ith hour of week. ($0 \leq s_i \leq f_i \leq 6 \times 24$).

Input terminates with two zeros for n and m, this case should not be processed.

Output

For each case output an integer, *CSC_FANITY* of Aideen.

Sample Input	Sample Output

2 1
1
1 3
4 6
 	
3 2
1 3
1 2
2 3
3 4
 	
0 0
	1
	0

Problem E: **Ehem, Ehem, Natural-Born-Losers!**

Nima has always been a lucky boy in his life. But in ACM ICPC competitions, he is out of look! -- he has got two rank of first-medalless-team in World Final competitions! However, this problem is not all about him. ;-)

Considering "*luck*" is a characteristical property of a person, It's believed that an ACM-ICPC team, which has exactly three members, will get the first place in a competition if and only if all 3 members of that team be lucky. That means if we show lucky persons with L and others with U, a team of "LLL" will certainly get the 1st rank; but not any team in the format of {ULL, LUL, LLU, LUU, ULU, UUL, UUU}.

Unfortuantely, it is quite rough to ask directly from a contestant "Are you lucky in ACM-ICPC?". So you are hired to solve this puzzle! Given some sort of information about contests result (in the last competitions), you are asked to find out if an specific team **has** chance to get 1st rank or not?

Input

$\mathbf{1} \le \boldsymbol{T} \le \mathbf{50}$, the number of test cases comes in the first line. After that, comes $\boldsymbol{T}$ blocks, describing one test case each. Each block begins with a number $\mathbf{0} \le \boldsymbol{N} \le \mathbf{50}$ that is the number of results we have about contestants. Following N lines contain one result per line.

A result begins with a single character `+` or `-`, which denotes the mentioned team has won (i.e. got the 1st place) or lost in the history. After that symbol, comes the name of 3 different contestants of that time separated by a single space. The names are between one and twenty characters in length and from lowercase or uppercase letters of English alphabet. The names are case insensitive (i.e. Nima and niMa are the same person).

Finally in the last line of the block (which is line $N + 2$, regard to the block's size) comes the query. The query is the name of your willing team, to find if it possible that this team gets 1st place or not?

Output

For each test, if the results are having contradiction to themselves, print "`Inconsistent data!`" in one line and advance to the next tests. Otherwise (the results are O.K.), if the specified team can get the 1st place without any contradiction with the belief of luck, print "`Yes, they can win!`", in a single line. And finally, if they have no chance to get the 1st place, print in one line: "`No, they are natural-born-losers!`".

Sample Input	Sample Output
2 1 - Nima Saeed Aideen Nima Saeed Hessam 3 - Simul Segment Matching + Simul AliBaba Sorting + Nima Saeeed Aideen Matching Segment Sorting	Yes, they can win! No, they are natural-born-losers!

Problem F: **Factorials Again!**

You may know how to compute number of zeros at the end of $n!$ in base10, but we want number of zeroes at the end of $n!$ in base k.

We define $n! = \prod_{i=1}^{n} i = 1 \times 2 \times \ldots \times n$. For example $5! = 1 \times 2 \times 3 \times 4 \times 5 = (120)_{10} = (1111000)_2 = (11110)_3 = (1320)_4$.

So number of zeroes at the end of 5! in base 3 is 1 and in base 2 is 3.

Input

Input consists of several test cases (at most $1{,}000$).

Each test case has two numbers n, k in a single line. $1 \leq n \leq 10^{18}$ and $2 \leq k \leq 500{,}000{,}000$.

The last line of input contains two zeros which should not be processed.

Output

For each n, k in a test case, print the number of zeroes at the end of $n!$ in base k in a single line.

.

Sample Input	Sample Output
5 2	3
5 3	1
5 10	1
123456789 1234	200416
0 0	

Problem G: **Glamorous Polygons**

Two polygons (with the same number of vertices) are given to you. The polygon has distinct vertices but may be degenerate in that the line segments could intersect. Because these polygons are generally created from remote images, there is some uncertainty as to their scale and rotation. Your job is to determine whether or not these two polygons are similar; that is, can they be made equal by repositioning, rotating and magnifying them?

Input

Input consists of multiple test cases.

The first line of each test case is $3 \le N \le 100{,}000$ that is the number of vertices in both polygons. Next N lines in that case describes the coordinate pairs of first polygon in counter-clockwise order. Next N lines contain the coordinate pairs of second polygon in counter-clockwise order. The coordinates of all vertices are less than 10^6 in absolute value.

A line containing $N = 0$ defines the last test case; which should not be processed.

Output

For each case, output a line "YES" or "NO" as appropriate (i.e. "YES" where they are similar and "NO" when they are not). The two **polygons are similar** if, after some combination of translation, rotation, and scaling (but not reflection) both vertices corresponding to each feature are in the same position.

Sample Input	Sample Output

3
0 0
1 0
0 1
2 0
-1 3
-1 0
3
0 0
1 0
0 1
0 0
0 1
0 5
0
	YES
	NO

Problem H: **Horribly-Fast Calculation**

You may know how to compute sum of all positive divisors for a positive integer n, but we want sum of this number for 1 to n as fast as possible. We define:

$$\sigma(n) = \text{sum of all positive divisors of } n$$

For example $\sigma(12) = 1 + 2 + 3 + 4 + 6 + 12 = 28$

And you should compute $\sum_{k=1}^{n} \sigma(k)$ for all given inputs.

Input

Input consists of several test cases (at most 10,000). Each test case is a number n in a single line that fits in $1 \le n \le 5{,}000{,}000$.

The last line of input is a single zero, which should not be processed.

Output

For each n in input you should write $\sum_{k=1}^{n} \sigma(k)$ in a single line.

.

Sample Input	Sample Output
5 10 0	21 87

Problem I: **Intergalactic Souvenirs**

E.T. (Extra Terrestrial -- an alien creature, who has come to the earth), has decided to go back to his planet. But before leaving the earth, he wants to pick two souvenirs -- one for his older brother S.T. (Super-extra Terrestrial) and one for his younger sister N.T. (not-necessarily-extra Terrestrial)!

E.T.'s best friend, Elliot, has offered E.T. a pack of N marbles. But since E.T. is too shy to accept all the marbles, he just wants to pick the heaviest marble for S.T. and the lightest one for N.T. You can consider no two marbles have the same weight, and they are indexed from 1 to N.

In order to find out which one is the heaviest and which one is the lightest, Elliot has brought a beam balance (scale) to E.T., and has already compared some pairs of marbles. The result of Elliot's scale actions are logged in the format "i-th marble is heavier than j-th marble" or "i-th marble is lighter than j-th marble" where $\mathbf{1 \leq \boldsymbol{i} \neq \boldsymbol{j} \leq \boldsymbol{N}}$.

With the help of these results, E.T. wants to find the heaviest and lightest marbles with the minimum number of additional weightings. We know each weighting action is to compare exactly two marbles with each other -- no mass comparison is possible. Please help E.T., and tell him the minimum number of weightings he needs from now to find his proper souvenirs!

Input

$\mathbf{1 \leq \boldsymbol{T} \leq 40}$, the number of test cases comes in the first line. After that, comes $\boldsymbol{T}$ block, describing one test case each. Each block begins with two numbers N $\mathbf{(2 \leq \boldsymbol{N} \leq 50,000)}$ and then K $\mathbf{(0 \leq \boldsymbol{K} \leq 50,000)}$ where N is the number of marbles and K is the number of comparisons (weightings) Eliot has already done. Then in the following $\boldsymbol{K}$ lines, comes one string (entered by user) per line in the format of "`i < j`" or "`i > j`" that means i-th marble is strictly lighter than j-th, or i-th marble is strictly heavier than j-th, respectively. It is guaranteed that $1 \leq i, j \leq N$. Please attend that there exists a whitespace before and after < and > characters in these lines.

Output

For each test case, print **`Implausible`** in a separate line, if there is any contradiction in Elliot's results. Otherwise, print the minimum number of weightings E.T. has to do to find out heaviest and lightest marbles in its own line. Thus, your output must have exactly T lines.

Sample Input	Sample Output

```
4
2 1
1 < 2
3 1
3 > 2
5 3
1 > 2
3 < 2
3 > 1
4 0
```
	```
	0
	2
	Implausible
	4
	```