IDI Open Programming Contest Practice Problems, 2011

Problem Set

Tips

- Tear the problem set apart and share the problems among you.
- Problems are not ordered by difficulty.
- Try solving the easy problems first. Two problems in this set are tagged with "(Easy)" to help point you in the right direction.
- If your solution fails on a problem, you can print your program and debug it on paper while you let someone else work on a different problem on the computer.
- If you need help, contact the judges.

Rules

- Each team consists of one to three contestants.
- One computer is used per team.
- You may not cooperate with persons not on your team.
- You may print your programs on paper to debug them.
- What you may bring to the contest:
  - Any written material (Books, manuals, handwritten notes, printed notes, etc).
  - Pens, pencils, blank paper, stapler and other useful non-electronic office equipment.
  - NO material in electronic form (CDs, USB pen and so on).
  - NO electronic devices (PDAs and so on).
- The only electronic content you may consult during the content is that specified by the organiser (see the web-page). You may not copy source code from web pages, etc.
- Your programs should read from standard in and write to standard out. Writing to standard error will result in a failed submission. C programs should return 0 from `main()`.
- Your program may use at most 100MB of memory.
- Your programs may not:
  - access the network,
  - read or write files on the system,
  - talk to other processes,
  - fork,
  - or similar stuff.
  - If you try, your program will hang or crash. If it hangs, it will take a couple of minutes before others will be able to run their programs. And please do not crack somebody who uses their spare time trying to give you something valuable.
- Show common sense and good sportsmanship.

Problem A

Addition (Easy)

Your little brother is having a hard time learning to add. As an effort to ensure he never learns this, you decide to write a program that adds two numbers for him.

Input specifications

The first line of input contains a single number T , the number of test cases. The following T lines contain two numbers each, A and B, the numbers you need to add.

Output specifications

For each test case, print a single line containing the number that equals A + B.

Notes and Constraints

- $0 < T \leq 100$
- $-1000 \leq A, B \leq 1000$

Sample input

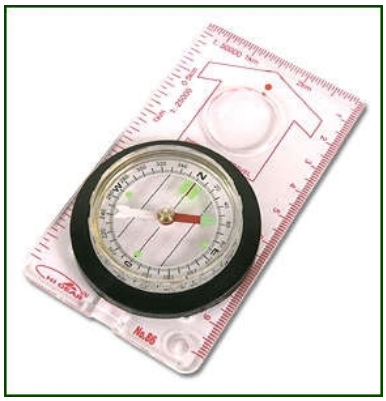
```
2
1 3
-7 0
```

Output for sample input

```
4
-7
```

Problem B

The Traveling Orienteerer (Easy)



Lasse is organizing the orienteering world championship in Trondheim in 2008. Since he has been running in Bymarka for decades, he has a lot of route proposals. He must find a route with just the right length, but he doesn't have time to measure them all by running through the control points in order, as he is too busy parallelizing code. The job is therefore given to Ola, who he thinks spends too much time with schoolwork.

As Ola is slightly less interested in orienteering, he figures out a more time-saving way of measuring the length of all the routes. He brings his GPS, visits all the points of all the routes in the most convenient order, and goes home early to do the rest of the job on his computer.

Input specifications

The first line of input gives $1 \leq n \leq 1000$, the total number of control points. Then follow n lines giving their coordinates, with two floating-point numbers x_i and y_i, with $0.0 \leq x_i, y_i \leq 10000.0$. The number of routes is then given by $1 \leq m \leq 100$. Each route is defined by first a line with $2 \leq p \leq 17$, the number of control points (including start and goal), and then a line with p indexes $0 \leq i < n$, identifying them.

Output specifications

For each test case output the total track distance, rounded, with no decimals.

Sample input

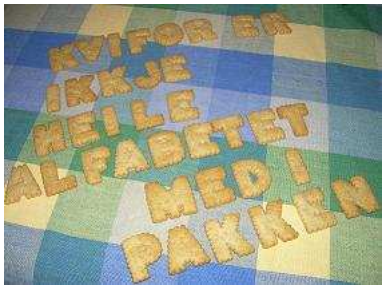
```
5
0.0 0.0
1000.0 1000.0
123.45 0.0
3475.43 7765.4
4325.9865 13.0
2
2
0 1
4
3 1 4 0
```

Output for sample input

```
1414
14999
```

Problem C

Letter Cookies



Sætre has finally decided to put their famous Letter Cookies back on market again. Of course they are just as interesting to play with as to eat. Your little sister is trying to make words out of the letters she found in the box, but you want to be faster than her to decide whether it is possible to make the word or not.

Given the letters in the cookie box, is it possible to spell out the words your little sister knows how to spell? (After creating a word, she scrambles the cookies again and can reuse the letters for later words.)

Input specifications

The first line of the input consists of a single number T, the number of letter cookie boxes your sister has. Each test case starts with a line describing all the letters in this box, in no particular order. Then follows a line with W, the number of words she would like to spell, and then follow the W words to write on a single line each.

Output specifications

For each word, output a line containing `YES` if it is possible to spell the word, or `NO` if it is not possible.

Notes and Constraints

- $0 < T \leq 100$
- $0 < W \leq 100$
- All letters are uppercase letters from the english alphabet (`A-Z`).
- There are at most 1000 letters in each cookie box.
- Each word has at most 100 letters (but is not necessarily an actual English word).

Sample input

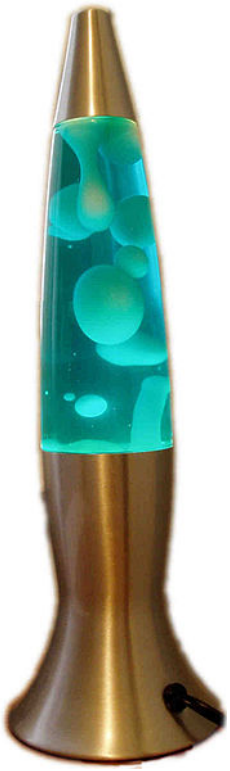
```
1
ABANANACOOKIE
4
BANANA
LETTER
COOKIES
CAN
```

Output for sample input

```
YES
NO
NO
YES
```

Problem D

The Diligent Cryptographer



Halvor is in charge of the Single Sign-On (SSO) login system for Identity Directories, Inc. He has been a passionate supporter of their technology for years, telling anyone who will listen how it makes user authentication simpler and more secure with the encrypted login back-end provided by Trustworthy Enterprises (TE). Last week Halvor got a newsletter from TE, where they introduced their new and highly innovative Open Trust Protection (OTP) system, which was recently implemented and has been used for new accounts and users that changed their password in the last month.

Previously, a user's cryptographic key consisted of a permutation of the first letters of the alphabet, repeated many times so it could be used for long messages. In the new improved system the cryptographic key instead consists of random letters generated by a lava lamp-based sub-contractor.

As an example the string `BCAEDBCAEDBCAED` was a possible key in the old system since this is a repetition of `BCAED`, which is a permutation of the letters from `A` to `E`. The strings `BCDBCD` and `BABBABBABBAB` would not be possible, since the letter `A` is missing from the repeated permutation `BCD`, and `BAB` is not a permutation of `AB` since there are two `B`'s.

Halvor decides to change the keys for the users that have not already been automatically moved to the new system. Luckily he has read and write access to all the keys for his users, and has contracted you to write a program to determine which users need to be updated. To avoid any privacy concerns, you are only given a list of the user's names, last login times and up to the first 1000 letters of their key.

Thus for the old system the end of the key substring you receive might be cut off in the middle of a repetition, but the first letter is guaranteed to be the start of a permutation. For the new system the entire string will be random, including the letters you receive.

Input specifications

The first line of input contains a single number T, the number of test cases to follow. Each test case consists of one line containing a string K, which is the first part of a user's cryptographic key.

Output specifications

For each test case, output a line containing the line "old" if K is definitely from the old system, "new" if K is definitely from the new system, or "unknown" if this cannot be determined from the provided key substring.

Notes and Constraints

- $1 \leq T \leq 1000$
- $1 \leq |K| \leq 1000$
- The entropy can be written as $H(X) = -\sum_{i=1}^{n} p(x_i) \log_b(x_i)$, where p denotes the probability mass function of X.

Sample input

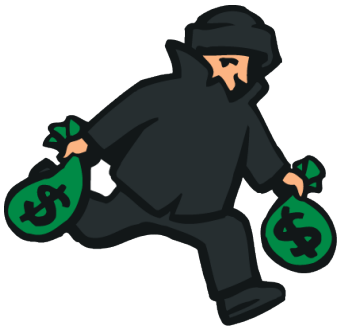
```
4
ABCD
BB
HELP
IAMTRAPPEDINACRYPTOGRAPHICKEYFACTORY
```

Output for sample input

```
unknown
new
unknown
new
```

Problem E

Robberies



The aspiring Roy the Robber has seen a lot of American movies, and knows that the bad guys usually gets caught in the end, often because they become too greedy. He has decided to work in the lucrative business of bank robbery only for a short while, before retiring to a comfortable job at a university.

For a few months now, Roy has been assessing the security of various banks and the amount of cash they hold. He wants to make a calculated risk, and grab as much money as possible. His mother, Ola, has decided upon a tolerable probability of getting caught. She feels that he is safe enough if the banks he robs together give a probability less than this.

Input specifications

The first line of input gives T, the number of cases. For each scenario, the first line of input gives a floating point number P, the probability Roy needs to be below, and an integer N, the number of banks he has plans for. Then follow N lines, where line j gives an integer M_j and a floating point number P_j. Bank j contains M_j millions, and the probability of getting caught from robbing it is P_j.

Output specifications

For each test case, output a line with the maximum number of millions he can expect to get while the probability of getting caught is less than the limit set.

Notes and Constraints

- $0 < T \leq 100$
- $0.0 \leq P \leq 1.0$
- $0 < N \leq 100$
- $0 < M_j \leq 100$
- $0.0 \leq P_j \leq 1.0$
- A bank goes bankrupt if it is robbed, and you may assume that all probabilities are independent as the police have very low funds.

Sample input

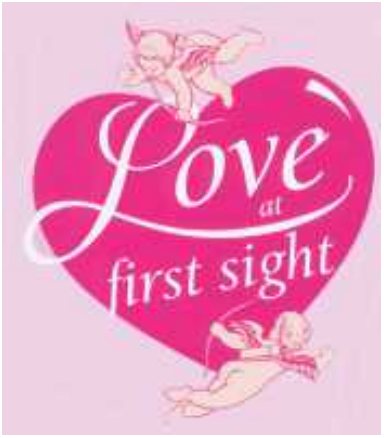
```
3
0.04 3
1 0.02
2 0.03
3 0.05
0.06 3
2 0.03
2 0.03
3 0.05
0.10 3
1 0.03
2 0.02
3 0.05
```

Output for sample input

```
2
4
6
```

Problem F

Party



Kjell Bratbergsengen, head of the computer science department, is concerned about the small number of girls studying computer science at NTNU. Representatives from Norwegian software companies complain constantly, but Kjell believes that the most dangerous possible effect of this problem is low overall recruitment in the future. It is well known that geeks usually do not have parents who studied at Dragvoll, and Kjell wants to do all he can to make sure his current students gets the opportunity to reproduce with decent partners.

While considering how to cope with this problem, Kjell remembers how things were when he studied at Gløshaugen. He recalls a lot of cute nurse students. It was not always easy to connect with them, as they were usually more interested in medical doctors than geeks, but charming young Kjell had his share of success nonetheless. Kjell believes that todays students also have a chance, and decides to arrange a party. Some male students already have girlfriends, but Kjell wants to make sure that all the single students get a date for the party as well. He tries to recruit nurse students, but realizes that it is impossible to get the number he needs, and many of them are picky about who they want date. How can he get a date for all the single geeks?

After some thinking, he comes up with the solution: He can just arrange several parties with the same girls! It is expensive to arrange parties, so there should be as few as possible. Even though Kjell is a talented programmer, he is now very busy with administrative chores, and needs your help writing a program which decides the minimum number of parties he will have to host.

Input specifications

The input has $n \leq 200$ cases, and the first line consists of a positive integer giving n. The first line for each test case consists of two positive integers separated by a single space, $m \leq 100$ and $f \leq 50$, where m denotes the number of male students who need a date, and f the number of nurse students available.

Then follow f lines. Line number i represents nurse number i. The line starts with a positive integer giving the number of male students she is willing to date. Then follows a list of space separated unique integers naming these geeks. The male students are numbered from 0 to $m - 1$.

Output specifications

Output one line for each test case. If it is impossible to make sure all male students get a date, no matter how many parties Kjell arranges, output a line with the text "`impossible`". Otherwise output a line with a single integer giving the minimum number of parties needed to make sure all male students can attend a party with a date.

Sample input

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

Output for sample input

```
2
3
impossible
```