Problemset
A - Planting Trees
B - Boiling Vegetables
C - Eligibility
D - Landline Telephone Network
E - Restaurant Ratings
F - Locked Treasure
G - Yet Satisfiability Again!
H - Chemistry
I - Fraud Busters
J - Watermelons

A - Planting Trees

Farmer Jon has recently bought n tree seedlings that he wants to plant in his yard. It takes 1 day for Jon to plant a seedling, and for each tree Jon knows exactly in how many days after planting it grows to full maturity. Jon would also like to throw a party for his farmer friends, but in order to impress them he would like to organize the party only after all the trees have grown. More precisely, the party can be organized at earliest on the next day after the last tree has grown up.

Help Jon to find out when is the earliest day when the party can take place. Jon can choose the order of planting the trees as he likes, so he wants to plant the trees in such a way that the party will be as soon as possible.

Input. The first line contains a single integer n ($1 \le n \le 100000$) denoting the number of seedlings. Then a line with n integers t_i ($1 \le t_i \le 1000000$) follows, where ti denotes the number of days it takes for the i-th tree to grow.

Output. You program should output exactly one line containing one integer, denoting the earliest day when the party can be organized. The days are numbered 1, 2, 3, ... beginning from the current moment.

Sample Input

```
4
2 3 4 3
```

Sample Output

```
7
```

B - Boiling Vegetables

The trick to boiling vegetables is to make sure all pieces are about the same size. If they are not, the small ones get too soft or the large ones get undercooked (or both). Fortunately, you have heard of the kitchen knife, but your parents' warnings of using sharp instruments still echoes in your head. Therefore you better use it as little as possible. You can take a piece of a vegetable of weight w and cut it arbitrarily in two pieces of weight w_{left} and w_{right}, where $w_{left} + w_{right} = w$. This operation constitutes a "cut".

Given a set of pieces of vegetables, determine the minimum number of cuts needed to make the ratio between the smallest and the largest resulting piece go above a given threshold.

Input. Starts with a floating point number t ($0.5 < t < 1$) with 2 decimal digits, and a positive integer n ($n \leq 1000$). Next follow n positive integer weights w_1, w_2, ..., w_n. All weights are less than 10^6.

Output. Output the minimum number of cuts needed to make the ratio between the resulting minimum weight piece and the resulting maximum weight piece be above t. You may assume that the number of cuts needed is less than 500. To avoid issues with floating point numbers, you can assume that the optimal answer for ratio t is the same as for ratio $t + 0.0001$.

Sample Input 1
```
0.99 3
2000 3000 4000
```

Sample Output 1
```
6
```

Sample Input 2
```
0.80 2
1000 1400
```

Sample Output 2
```
3
```

C - Eligibility

Every year, students across the world participate in the ACM ICPC. In order to participate in this contest, a student must be eligible to compete. In this problem, you will be given information about students and you will write a program to determine their eligibility to participate in the ICPC.

We will start by assuming that each student meets the “Basic Requirements” as specified in the ICPC rules – the student is willing to compete at the World Finals, is a registered student with at least half-time load, competes for only one institution in a contest year, and has not competed in two world finals or five regional contests.

The rules to decide if a student is eligible to compete in the contest year 2014 - 2015 are as follows:

- if the student first began post-secondary studies in 2010 or later, the student is eligible;
- if the student is born in 1991 or later, the student is eligible;
- if none of the above applies, and the student has completed more than an equivalent of 8 semesters of full-time study, the student is ineligible;
- if none of the above applies, the coach may petition for an extension of eligibility by providing the student's academic and work history.

For "equivalent of 8 semesters of full-time study", we consider each semester of full-time study to be equivalent to a student completing 5 courses. Thus, a student who has completed 41 courses or more is considered to have more than 8 semesters of full-time study.

Input. Consists of a number of cases. The first line contains a positive integer, indicating the number of cases to follow. Each of the cases is specified in one line in the following format

name YYYY/MM/DD YYYY/MM/DD courses

where *name* is the name of the student (up to 30 alphabetic characters), the first date given is the date the student first began post-secondary studies, and the second date given is the student's date of birth. All dates are given in the format above with 4-digit year and 2-digit month and day. *courses* is a non-negative integer indicating the number of courses that the student has completed. There are at most 1000 cases.

Output. For each line print the student's name, followed by a space, followed by one of the strings **eligible**, **ineligible**, and **coach petitions** as appropriate.

Sample Input

```
3
EligibleContestant 2013/09/01 1995/03/12 10
IneligibleContestant 2009/09/01 1990/12/12 50
PetitionContestant 2009/09/01 1990/12/12 35
```

Sample output

```
EligibleContestant eligible
IneligibleContestant ineligible
PetitionContestant coach petitions
```

D - Landline Telephone Network

The mayor of RMRCity wants to create a secure landline telephone network for emergency use in case of serious disasters when the city is cut off from the outside world. Some pairs of buildings in the city can be directly connected with a wire telephone line and the municipality engineers have prepared an estimate of the cost of connecting any such pair.

The mayor needs your help to find the cheapest network that connects all buildings in the city and satisfies a particular security measure that will be explained shortly. A call from a building A to another building B may be routed through any simple path in the network (i.e., a path that does not have any repeated building). There are also some insecure buildings that one or more persons with serious criminal records live in. The mayor wants only communications intended for these insecure buildings to reach them. In other words, no communication from any building A to any building B should pass through any insecure building C in the network (where C is different from A and B).

Input. The first line contains three integers n, m, p where n ($1 \leq n \leq 1000$) is the number of buildings, m ($0 \leq m \leq 100000$) is the number of possible direct connections between a pair of buildings, and p ($0 \leq p \leq n$) is the number of insecure buildings. The buildings are numbered from 1 to n. The second line contains p distinct integers between 1 and n (inclusive), which are the numbers of insecure buildings. Each of the next m lines contains three integers x_i, y_i and l_i describing one potential direct line, where x_i and y_i ($1 \leq x_i, y_i \leq n$) are the distinct buildings the line connects, and l_i ($1 \leq l_i \leq 10000$) is the estimate of the cost of connecting these buildings. There is at most one direct link between any two buildings in these m lines.

Output. Display the cost of the cheapest network satisfying the security measure if it is possible. Otherwise, display impossible.

Sample Input

```
4 6 1
1
1 2 1
1 3 1
1 4 1
2 3 2
2 4 4
3 4 3
```

Sample Output

```
6
```

E - Restaurant Ratings

A famous travel web site has designed a new restaurant rating system. Each restaurant is rated by one of n ($1 \le n \le 15$) critics, each giving the restaurant a nonnegative numeric rating (higher score means better). Some of these critics are more influential than others.

The restaurants in each city are ranked as follows. First, sum up the ratings given by all the critics for a restaurant. A restaurant with a higher total sum is always better than one with a lower total sum. For restaurants with the same total sum, we rank them based on the ratings given by critic 1. If there is a tie, then we break ties by the ratings by critic 2, etc.

A restaurant owner received the ratings for his restaurant, and is curious about how it ranks in the city. He does not know the ratings of all the other restaurants in the city, so he would estimate this by computing the maximum number of different ratings that is no better than the one received by the restaurant. You are asked to write a program to answer his question.

Input. The input consists of a number of cases. Each case is specified on one line. On each line, the first integer is n, followed by n integers containing the ratings given by the n critics (in order). You may assume that the total sum of ratings for each restaurant is at most 30. The input is terminated by a line containing $n = 0$.

Output. For each input, print the number of different ratings that is no better than the given rating. You may assume that the output fits in a 64-bit signed integer.

Sample input

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

Sample output

```
4
33
10810
```

F - Locked Treasure

A group of *n* bandits hid their stolen treasure in a room. The treasure needs to be locked away until there is a need to retrieve it. Since the bandits do not trust each other, they wanted to ensure that at least m of the bandits must agree in order to retrieve the treasure.

They have decided to place multiple locks on the door such that the door can be opened if and only if all the locks are opened. Each lock may have up to *n* keys, distributed to a subset of the bandits. A group of bandits can open a particular lock if and only if someone in the group has a key to that lock.

Given *n* and *m*, how many locks are needed such that if the keys to the locks are distributed to the bandits properly, then every group of bandits of size at least *m* can open all the locks, and no smaller group of bandits can open all the locks?

For example, if $n = 3$ and $m = 2$, only 3 locks are needed – keys to lock 1 can be given to bandits 1 and 2, keys to lock 2 can be given to bandits 1 and 3, and keys to lock 3 can be given to bandits 2 and 3. No single bandit can open all the locks, but any group of 2 bandits can open all the locks. You should also convince yourself that it is not possible to satisfy the requirements with only 2 locks.

Input. The first line contains the number of cases to follow. Each case is specified by the two integers n ($1 \leq n \leq 30$) and m ($1 \leq m \leq n$) on one line.

Output. For each line print on one line the minimum number of locks needed.

Sample Input

```
4
3 2
5 1
10 7
5 3
```

Sample Output

```
3
1
210
10
```

G - Yet Satisfiability Again!

Alice recently started to work for a hardware design company and as a part of her job, she needs to identify defects in fabricated integrated circuits. An approach for identifying these defects boils down to solving a satisfiability instance. She needs your help to write a program to do this task.

Input. The first line contains a single integer, not more than 5, indicating the number of test cases to follow. The first line of each test case contains two integers n ($1 \leq n \leq 20$) and m ($1 \leq m \leq 100$), where n indicates the number of variables and m indicates the number of clauses. Then m lines follow corresponding to each clause. Each clause is a disjunction of literals in the form X_i or $\sim X_i$ for some i ($1 \leq i \leq n$), where $\sim X_i$ indicates the negation of the literal X_i. The "or" operator is denoted by a v character and is separated from literals with a single space.

Output. For each test case, display satisfiable on a single line if there is a satisfiable assignment; otherwise display unsatisfiable.

Sample Input

```
2
3 3
X1 v X2
~X1
~X2 v X3
3 5
X1 v X2 v X3
X1 v ~X2
X2 v ~X3
X3 v ~X1
~X1 v ~X2 v ~X3
```

Sample Output

```
satisfiable
unsatisfiable
```

H - Chemistry

The chemical formula of a molecule M describes its atomic make-up. Chemical formulas obey the following grammar:

```
M := G | M G

G := S | S C

S := A | '(' M ')'

C := T | N E

E := D | D E

T := '2' | ... | '9'

N := '1' | ... | '9'

D := '0' | .. | '9'

A := U | U L | U L L

U := 'A' | .. | 'Z'

L := 'a' | .. | 'z'
```

The count C represents a multiplier for the subgroup S that precedes it. For example, H2O has two H (hydrogen) and one O (oxygen) atoms, and (AlC2)3Na4 contains 3 Al (aluminum), 6 C (carbon) and 4 Na (sodium) atoms.

Input. Contains multiple test cases. For each test case, there will be one line, containing a valid chemical formula. Each line will have no more than 100 characters.

Output. For each line there will be one line of output which is the atomic decomposition of the chemical in the form of a sum as shown in the sample output. The atoms are listed in lexicographical order, and a count of 1 is implied and not explicitly written. There are no blank spaces in the output. All of the counts in the correct output will be representable in 32-bit signed integers.

Sample Input

```
H2O
AlC2)3Na4
```

Sample Output

```
2H+O
3Al+6C+4Na
```

I - Fraud Busters

The number of cars in Default City that travel to the city center daily vastly exceeds the number of available parking spots. The City Council had decided to introduce parking fees to combat the problem of overspill parking on the city streets. Parking fees are enforced using an automated vehicle registration plate scanners that take a picture of the vehicle registration plate, recognize the sequence of digits and letters in the code on the plate, and check the code against a vehicle registration database to ensure that parking fees are dutifully paid or to automatically issue a fine to the vehicle owner otherwise.

As soon as parking fees were introduced, a parking fee fraud had appeared. Some vehicle owners had started to close one or several digits or letters on their vehicle registration plate with pieces of paper while they park, thus making it impossible for the current version of the automated scanner to recognize their vehicle's registration code and to issue them a fine.

The Default City Council had instituted the Fraud Busters Initiative (FBI) to design a solution to prevent this kind of fraud. The overall approach that FBI had selected is to expand the number of vehicle features that scanners recognize (including features like vehicle type and color), as well as excluding from the list any vehicles that are detected to be elsewhere at this time. This information should help to identify the correct vehicle by narrowing down the search in the vehicle registration database.

You are working for FBI. Your colleagues had already written all the complex pieces of the recognition software that analyses various vehicle features and provides you with a list of registration codes that might potentially belong to a scanned car. Your task it to take this list and a recognized code from the license plate (which may be partially unrecognized) and find all the registration codes that match.

Input. The first line contains 9 characters of the code as recognized by the scanner. Code that was recognized by the the scanner is represented as a sequence of 9 digits, uppercase English letters, and characters “*” (star). Star represents a digit or a letter that scanner could not recognize.

The second line contains a single integer number n ($1 \le n \le 1000$) – the number of vehicle registration codes from the vehicle registration database.

The following n lines contain the corresponding registration codes, one code per line. Vehicle registration codes are represented as a sequence of 9 digits and uppercase English letters. All codes on these n lines of the input are different.

Output. On the first line write a single integer k ($0 \le k \le n$) – the number of codes from the input that match the code that was recognized by the scanner. The code from the scanner matches the code from the database if the characters on all the corresponding positions in the codes are equal or the character from the scanner code is “*”.

On the following k lines write the matching codes, one code per line, in the same order as they are given in the input.

Sample Input

```
A**1MP19*
4
A001MP199
E885EE098
A111MP199
KT7351TTB
```

Sample Output

```
2
A001MP199
A111MP199
```

J - Watermelons

Ahmad Ahmadov came to the market and decided to buy two watermelons: one for himself and another for the wife's mother. It is clear to choose for himself the heaviest watermelon, and for mother-in-law the lightest. But there is one problem: there are many watermelons and he does not know how to choose the lightest and the heaviest one. Help him!

Input

The first line contains the number of watermelons **n** (**n** ≤ **30000**). The second line contains **n** numbers, each number is a mass of corresponding watermelon. All weights of watermelons are positive integers and do not exceed 30000.

Output

Print two numbers: the weight of watermelon that Ahmad Ahmadov will buy for his mother-in-law and the weight of watermelon that he will buy himself, or print the message "**Ooops!**" (without quotes), if someone left without watermelon.

Input example #1

5
5 1 6 5 9

Output example #1

1 9