Problem A
Server
Problem ID: server

Time Limit: 1 second

Picture from Wikimedia Commons

You are in charge of a server that needs to run some submitted tasks on a first-come, first-served basis. Each day, you can dedicate the server to run these tasks for at most T minutes. Given the time each task takes, you want to know how many of them will be finished today.

Consider the following example. Assume $T = 180$ and the tasks take $45, 30, 55, 20, 80$, and 20 minutes (in order they are submitted). Then, only four tasks can be completed. The first four tasks can be completed because they take 150 minutes, but not the first five, because they take 230 minutes which is greater than 180. Notice that although there is enough time to perform the sixth task (which takes 20 minutes) after completing the fourth task, you cannot do that because the fifth task is not done yet.

Input

The input consists of a single test case. The first line contains two integers n and T where $1 \leq n \leq 50$ is the number of tasks and $1 \leq T \leq 500$. The next line contains n positive integers no more than 100 indicating how long each task takes in order they are submitted.

Output

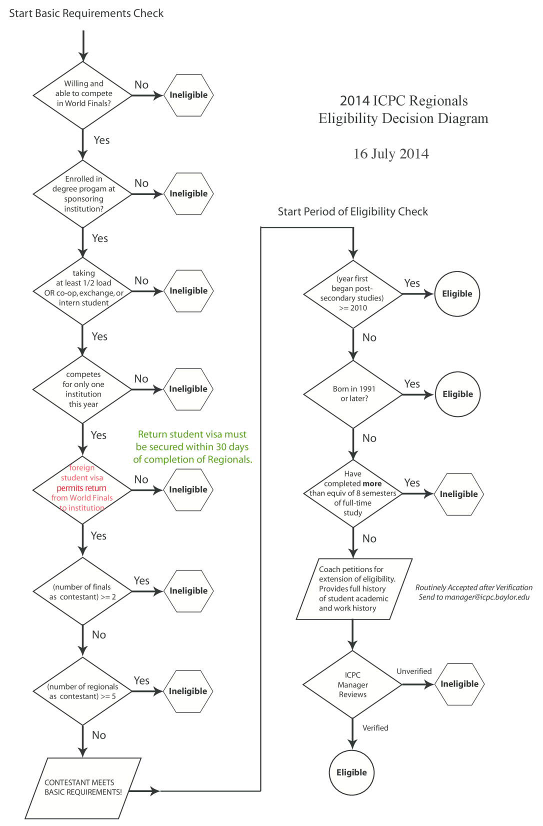
Display the number of tasks that can be completed in T minutes on a first-come, first-served basis.

Sample Input	Sample Output
6 180 45 30 55 20 80 20	4

Sample Input	Sample Output
10 60 20 7 10 8 10 27 2 3 10 5	5

Problem B
Eligibility
Problem ID: eligibility
Time Limit: 2 seconds

Every year, students across the world participate in the ACM ICPC[1]. 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.



Start Basic Requirements Check
Willing and able to compete in World Finals?
No
Ineligible
Yes
Enrolled in degree progam at sponsoring institution?
No
Ineligible
Yes
taking at least 1/2 load OR co-op, exchange, or intern student
No
Ineligible
Yes
competes for only one institution this year
No
Ineligible
Yes
Return student visa must be secured within 30 days of completion of Regionals.
foreign student visa permits return from World Finals to institution
No
Ineligible
Yes
(number of finals as contestant) >= 2
Yes
Ineligible
No
(number of regionals as contestant) >= 5
Yes
Ineligible
No
CONTESTANT MEETS BASIC REQUIREMENTS!
2014 ICPC Regionals
Eligibility Decision Diagram
16 July 2014
Start Period of Eligibility Check
(year first began post-secondary studies) >= 2010
Yes
Eligible
No
Born in 1991 or later?
Yes
Eligible
No
Have completed more than equiv of 8 semesters of full-time study
Yes
Ineligible
No
Coach petitions for extension of eligibility. Provides full history of student academic and work history
Routinely Accepted after Verification
Send to manager@icpc.baylor.edu
ICPC Manager Reviews
Unverified
Ineligible
Verified
Eligible

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:

1. if the student first began post-secondary studies in 2010 or later, the student is eligible;
2. if the student is born in 1991 or later, the student is eligible;
3. 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;
4. 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.

[1]This may be the only problem statement in which these acronyms expand to Association for Computing Machinery International Collegiate Programming Contest.

Input

The 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 1 000 cases.

Output

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

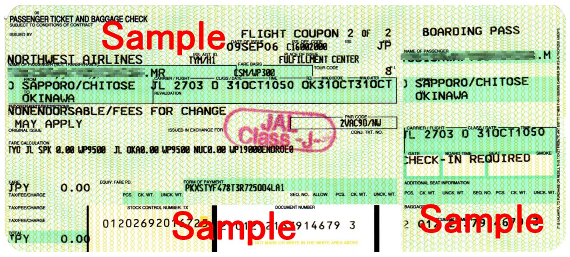
Sample Input

```
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
```

Problem C
Plane Ticket Pricing
Problem ID: seats
Time Limit: 2 seconds



Picture from Wikimedia Commons

Plane ticket prices fluctuate wildly from one week to the next, and their unpredictability is a major source of frustration for travellers. Some travellers regret buying tickets too early when the prices drop right after they purchase the tickets, and some travellers regret buying tickets too late when prices rise right before they are about to make the purchase. At the end, no one is happy, except the airlines, of course.

Surely there is some reason to this madness. It turns out that airlines price their tickets dynamically, based on how many seats are still available and how close the flight is. For example, if there are very few seats left on a flight then the tickets may be expensive until the last few weeks before the flight, at which point the prices may decrease to fill the empty seats. Ultimately, the airlines wish to maximize revenue from each flight.

You have been hired by the International Contrived Pricing Corporation (ICPC) to set ticket prices each week for airlines. The airlines have collected and analyzed historical data, and have good estimates on the number of seats that will be sold at a particular ticket price with a particular number of weeks before the flight. Given the number of seats left on a flight as well as the number of weeks left before the flight, your job is to set the ticket price for the current week, in order to maximize the total revenue obtained from ticket sales from the current week to the time of the flight. You may assume that the number of tickets sold is exactly the same as the estimates, unless there are not enough remaining seats. In that case, all remaining seats will be sold. You may also assume that the optimal ticket prices will be chosen for the remaining weeks before the flight.

Note that higher prices do not necessarily mean fewer tickets will be sold. In fact, higher prices can sometimes increase sales as travellers may be worried that the prices will rise even higher later.

Input

The input consists of one case. The first line contains two integers, N and W, the number of seats left and the number of weeks left before the flight ($0 < N \leq 300$, $0 \leq W \leq 52$). The next $W + 1$ lines give the estimates for W weeks, $W - 1$ weeks, …, and down to 0 weeks (i.e. last week) before the flight. Each of these lines starts with an integer K ($0 < K \leq 100$), the number of different prices to consider that week. This is followed by K integers $0 < p_1 < \cdots < p_K < 1000$ giving the prices in dollars. Finally, this is followed by K additional integers $s_1, \ldots, s_K$ ($0 \leq s_i \leq N$) indicating the number of tickets that will be sold for the corresponding prices.

Output

On the first line, print the maximum total revenue the airline can obtain from ticket sales from the current week to the time of the flight. On the second line, print the ticket price to set for the current week (W weeks before the flight) to achieve this maximum.

If there are multiple sets of ticket prices achieving this maximum, choose the smallest ticket price for week W.

Sample Input

```
50 2
1 437 47
3 357 803 830 13 45 46
1 611 14
```

Sample Output

```
23029
437
```

Sample Input

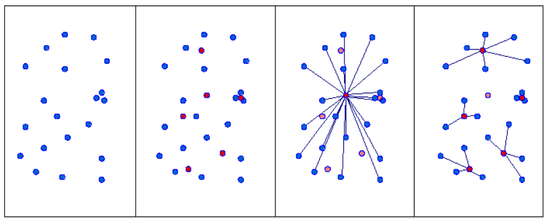
```
100 3
4 195 223 439 852 92 63 15 1
2 811 893 76 27
1 638 3
1 940 38
```

Sample Output

```
83202
852
```

Problem D
Facility Locations
Problem ID: facility

Time Limit: 1 second


Picture from Wikimedia Commons

The HDWBP Inc. has n clients and needs to service these clients by opening k facilities. Each opened facility can serve any number of clients and each client must be served by an open facility. There are m potential locations for these k facilities. The cost of serving client j at potential location i is a non-negative integer c_{ij}. These costs satisfy a locality property: for two clients j and j' and two facilities i and i', we have $c_{ij} \leq c_{i'j} + c_{i'j'} + c_{ij'}$. Given the costs, the CEO of HDWBP Inc. ultimately wants to know the cheapest way to open k facilities and assign clients to these open facilities. For now, he needs your help to determine if it is possible to do this task without any cost (i.e. with cost zero).

Input

The input consists of a single test case. The first line contains three integers m, n, k where $1 \leq m \leq 100$, $1 \leq n \leq 100$ and $1 \leq k \leq m$. Each of the next m lines contains n non-negative integers where the jth integer in the ith line is $c_{ij} \leq 10\,000$.

Output

Display `yes` if it is possible to do the task with cost zero; otherwise, display `no`.

Sample Input	Sample Output

3 2 2
0 2
1 1
2 0
	yes

Sample Input	Sample Output

3 3 2
0 2 2
1 1 1
2 2 0
	no

Problem E
Repeated Substrings
Problem ID: substrings
Time Limit: 5 seconds

Picture from Wikimedia Commons

String analysis often arises in applications from biology and chemistry, such as the study of DNA and protein molecules. One interesting problem is to find how many substrings are repeated (at least twice) in a long string.

In this problem, you will write a program to find the total number of repeated substrings in a string of at most 100 000 alphabetic characters. Any unique substring that occurs more than once is counted. As an example, if the string is "aabaab", there are 5 repeated substrings: "a", "aa", "aab", "ab", "b". If the string is "aaaaa", the repeated substrings are "a", "aa", "aaa", "aaaa". Note that repeated occurrences of a substring may overlap (e.g. "aaaa" in the second case).

Input

The input consists of at most 10 cases. The first line contains a positive integer, specifying the number of cases to follow. Each of the following line contains a nonempty string of up to 100 000 alphabetic characters.

Output

For each line of input, output one line containing the number of unique substrings that are repeated. You may assume that the correct answer fits in a signed 32-bit integer.

Sample Input

```
3
aabaab
aaaaa
AaAaA
```

Sample Output

```
5
4
5
```

Problem F
Landline Telephone Network
Problem ID: landline

Time Limit: 2 seconds

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 input consists of a single test case. The first line contains three integers n, m, p where $1 \leq n \leq 1\,000$ is the number of buildings, $0 \leq m \leq 100\,000$ is the number of possible direct connections between a pair of buildings, and $0 \leq p \leq n$ 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 $(1 \leq \ell_i \leq 10\,000)$ 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	Sample Output

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

Sample Input	Sample Output

4 3 2
1 2
1 2 1
2 3 7
3 4 5
	impossible

Problem G
Aquarium Tank
Problem ID: tank
Time Limit: 1 second

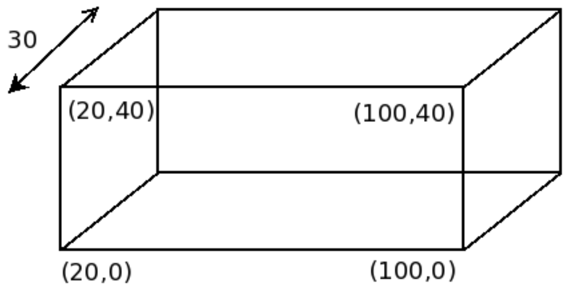
Picture from Wikimedia Commons

You just bought an "artistic" aquarium tank that has an interesting shape, and you poured L litres of water into the tank. How high is the water in the tank?

When you look at this tank from one side, it has the shape of a convex polygon. This polygon has exactly two vertices on the table (y-coordinates are 0), and all other vertices have positive y-coordinates. There are also exactly two vertices with maximum y-coordinates, and water is poured into the opening between these two vertices. This aquarium tank has a depth of D centimetres. The tank is glued to the table, so no matter what shape it has, it keeps its position and does not tip over.

All coordinates and lengths in this problem are given in centimetres. It should be noted that each cubic metre is equivalent to 1 000 litres.

An illustration showing the configuration of the tank of the first sample input is given below:



Input

The input consists of a single test case. The first line contains an integer N ($4 \leq N \leq 100$) giving the number of vertices in the polygon. The next line contains two integers D and L, where $1 \leq D \leq 1\,000$ is the depth of the aquarium tank and $0 \leq L \leq 2\,000$ is the number of litres of water to pour into the tank. The next N lines each contains two integers, giving the (x, y) coordinates of the vertices of the convex polygon in counterclockwise order. The absolute values of x and y are at most $1\,000$. You may assume that the tank has a positive capacity, and you never pour more water than the tank can hold.

Output

Print the height of the water (in centimetres) in the aquarium tank on a line to 2 decimal places.

Sample Input

```
4
30 50
20 0
100 0
100 40
20 40
```

Sample Output

```
20.83
```

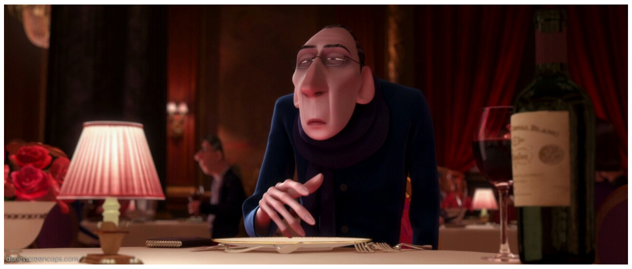
Sample Input

```
9
30 70
110 70
100 80
80 80
-10 60
-40 30
-40 25
20 0
100 0
120 10
```

Sample Output

```
19.74
```

Problem H
Restaurant Ratings
Problem ID: ratings
Time Limit: 1 second



A famous travel web site has designed a new restaurant rating system. Each restaurant is rated by one of n ($1 \leq n \leq 15$) 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	Sample Output

1 3
2 4 3
5 4 3 2 1 4
0
	4
	33
	10810

Problem I
Locked Treasure
Problem ID: locks
Time Limit: 1 second

Picture from Wikimedia Commons

A group of n ($1 \leq n \leq 30$) 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 ($1 \leq m \leq n$) 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 of input contains a positive integer indicating the number of cases to follow. Each case is specified by the two integers n and m on one line.

Output

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

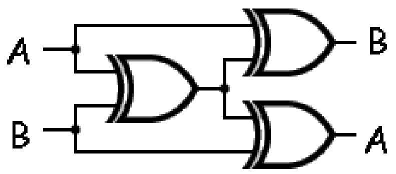
Sample Input	Sample Output

4
3 2
5 1
10 7
5 3
	3
	1
	210
	10

Problem J
Yet Satisfiability Again!
Problem ID: sat

Time Limit: 5 seconds

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.



Picture from Wikimedia Commons

Input

The first line of input 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 and m where $1 \leq n \leq 20$ indicates the number of variables and $1 \leq m \leq 100$ indicates the number of clauses. Then, m lines follow corresponding to each clause. Each clause is a disjunction of literals in the form Xi or $\sim Xi$ for some $1 \leq i \leq n$, where $\sim Xi$ indicates the negation of the literal Xi. The "or" operator is denoted by a 'v' character and is seperated 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
```

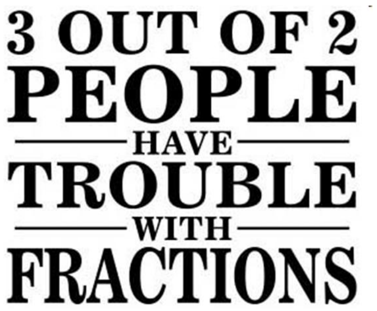
Sample Output

```
satisfiable
unsatisfiable
```

This page is intentionally left blank.

Problem K
Continued Fraction
Problem ID: fraction

Time Limit: 1 second



The (simple) continued fraction representation of a real number r is an expression obtained by an iterative process of representing r as the sum of its integer part and the reciprocal of another number, then writing this other number as the sum of its integer part and another reciprocal, and so on. In other words, a continued fraction representation of r is of the form

$$r = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \dots}}}$$

where $a_0, a_1, a_2, \dots$ are integers and $a_1, a_2, \dots > 0$. We call the a_i-values *partial quotients*. For example, in the continued fraction representation of 5.4, the partial quotients are $a_0 = 5, a_1 = 2, a_2 = 2$. This representation of a real number has several applications in theory and practice. If r is a rational number, the partial quotients are eventually all zero, so we only need to consider a finite number of partial quotients.

Given two rational numbers in continued fraction representation, your task is to perform the four elementary arithmetic operations on these numbers and display the results in continued fraction representation.

Input

The input consists of a single test case. The test case consists of three lines. The first line contains two integers n_1 and n_2, where $1 \le n_i \le 9$ is the number of partial quotients of rational number r_i for $1 \le i \le 2$. The second line contains the partial quotients of r_1 and the third line contains the partial quotients of r_2. The absolute values of the quotients are not more than 10 and you may assume that $r_1 > r_2 > 0$.

Output

Display the partial quotients of the continued fraction representations of $r_1 + r_2$, $r_1 - r_2$, $r_1 \times r_2$, and r_1 / r_2, in order, each in a line. Consecutive partial quotients on each line are separated by a single space. Do not print any trailing zero partial quotients.

Sample Input

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

Sample Output

```
11
0 5
30 4 6
1 27
```