THE ACM-ICPC 2015

VIETNAM SOUTHERN PROGRAMMING CONTEST

Host: University of Science, VNU-HCM

September 26, 2015

Contest length: 5 hours

The problem set consists of 11 problems in 16 pages (*excluding the cover page*):

- Problem A: **Lantern**
- Problem B: **Mooncake**
- Problem C: **Space Station**
- Problem D: **Minions**
- Problem E: **Wheel of Fortune**
- Problem F: **Memory Mosaic**
- Problem G: **Lantern Exhibition**
- Problem H: **Arithmetic**
- Problem I: **Buffet**
- Problem J: **Lunar Crystal**
- Problem K: **Business Intelligence**

Problem A
Lantern
Time Limit: 1 second

Mid-Autumn Festival is coming. Kids always love to play with lanterns. Therefore, you decide to make many lanterns to donate for young children.

There are M different types of lanterns and you know the number of lanterns of each type to be made.

You have N robots that can help you make lanterns automatically and you can only assign a robot to make a single type of lanterns. In fact, you are not required to use all N robots to make lanterns.

To quickly have all lanterns ready for young kids, you should minimize the maximum number of lanterns a robot should make.

Input

The first line of input contains two integers N ($1 \leq N \leq 10^9$) and M ($1 \leq M \leq 10^5$). The next M lines each contains a single positive integer in range from 1 to 10^9, which is the number of lanterns of the i^{th} type that you want to make.

Output

Display a single positive integer that is the minimum value of the maximum number of lanterns that a robot makes.

Sample Input	Sample Output
5 2 7 4	3

Problem B
Mooncake
Time Limit: 1 second

A special delicious mooncake has been prepared for Mid-Autumn Festival!

The shape of the cake is a polygon (***not*** necessarily convex). This polygon is ***not*** a degenerate one, that means no vertex lies on an edge joining two other vertices.

The mooncake is expected to serve $K + 1$ visitors.

You use a knife to cut the cake along a horizontal line $y = y_1$ to serve the first visitor. When the second visitor arrives, you cut the cake along another horizontal line $y = y_2 > y_1$ to serve him or her, and so on.

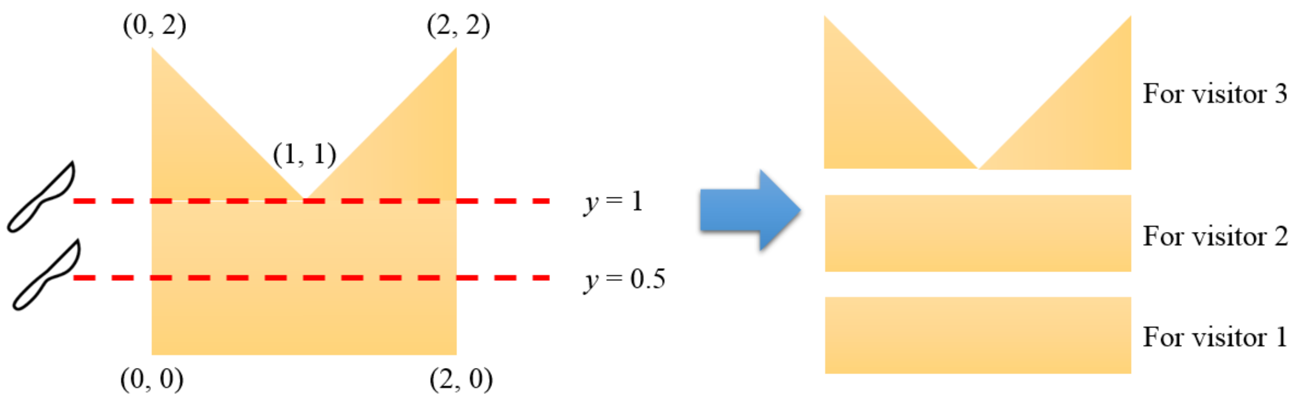
Using K cuts with K distinct horizontal lines, you can split the mooncake into $K + 1$ parts so that $K + 1$ visitors can enjoy the special mooncake. Each part may contain multiple separated pieces of the cake.

Your task is to find the set of K horizontal lines to split the mooncake into $K + 1$ parts with the same area, each of which will be served for a visitor.

Note: A horizontal line contains all points with the same Y-coordinate.

Example:

To serve three visitors, the mooncake is cut with two horizontal lines, $y = 0.5$ and $y = 1$.



Input

The first line of input contains two integers N and K, $3 \le N \le 65\,535$, $1 \le K \le 65\,535$, where N is the number of vertices of the polygon, and K is the number of lines to cut.

The next N lines contain the vertices of the polygon in clockwise order. Each vertex is defined by its coordinates X and Y not exceeding 10000 in absolute value.

Output

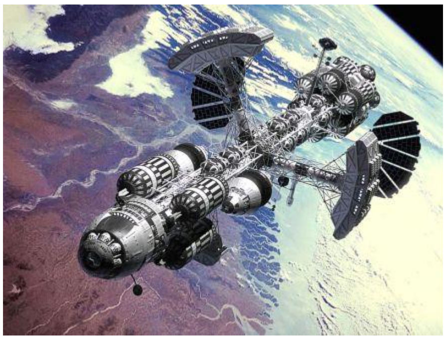
The output file should contain K lines, each of which contains the Y-coordinate of a horizontal line. Those coordinates should be in ascending order.

Your output should have an absolute or relative error of at most 10^{-4}.

Sample Input 1	Sample Output 1
5 2 0 0 0 2 1 1 2 2 2 0	0.5000 1.0000

Sample Input 2	Sample Output 2
4 1 0 0 0 3 3 3 3 0	1.5000

Problem C
Space Station
Time Limit: 2 seconds



The Moon is shining, especially in Mid-Autumn night. Do you want to explore the Moon? It would not be possible for everyone to visit the Moon. However, you can enjoy a virtual trip in a virtual spaceship from the Earth to the Moon. Now, let us board the spaceship. You are an astronaut!

The virtual spaceship flights directly from the Earth to the Moon in a single line. Each point in the path from the Earth to the Moon is labeled with a positive integer that is the distance from the Earth to that point.

Each space station is built at the location whose label is a prime number to provide supply for spaceships. As you are looking for food and water supply, you should find the nearest space station.

Input

The input consists of multiple test cases.

The first line of input contains an integer T ($1 \le T \le 100$), the number of test cases. Each of the next T lines describes one test case. Each test case contains a positive integer A ($2 \le A \le 10^6$) that is the current location of your spaceship.

Output

Display the result of each test case in a separate line. For each test case, display an integer that is the shortest distance (in absolute value) from the current location of your spaceship to the nearest space station.

Sample Input	Sample Output
4 4 122 25 65537	1 5 2 0

Problem D
Minions
Time Limit: 1 second

Do you know that Minions can live on the Moon? Believe it or not! Actually, you can see a lot of Minions wandering around in the Virtual Moon in the Mid-Autumn Festival.

"Bello!" (or "Hello!" in English). Do you also know that Dr. Nefario is the first man on Earth that taught minions to speak? In the Mid-Autumn Festival this year, Dr. Nefario will teach his Lunar minions new and much longer words.

According to his secret research, a minion-word or a minion-string can be represented as a lowercase string and has this particular property: It is lexicographically smaller than each of its proper suffixes (different from the string itself). All strings of length one are minion-strings. A suffix of a string *S* is a substring of *S*, starting somewhere in *S* and finishes at the end of *S*.

As a teaching assistant of Dr. Nefario, you are asked to split a given minion-string in a concatenation of minion-substrings, such that each substring (except the first one) is lexicographically smaller than or equal to the previous one. So that he can estimate the length of his lecture.

Input

The single row of the standard input will contain the given minion-string. It will less than 10^6 characters of Latin alphabet.

Output

Display in a single line the number of substrings, following by the length of each substring, separated by a single space.

Sample Input	Sample Output
`waaw`	`2 1 3`

Explanation: The given minion-string is a concatenation of the following 2 minion-substrings:

`w`

```
aaw
```

Problem E
Wheel of Fortune
Time Limit: 1 second

You are lucky to become an invited player for the game show Wheel of Fortune. The special award in this game show is the free ticket for you and one of your friends to enjoy the 4D Virtual Tour to explore the Moon.

Similar to the regular Wheel of Fortune, you have to guess a secret phrase S containing only no more than 100 lowercase letters (a→z).

You will sequentially guess a letter that might appear in the secret phrase S. If that character actually exists in S, all of its appearances in S (maybe more than one time) will be displayed.

A guess is `ACCEPTED` if the letter you guess appears in S (one or multiple times) and you have not guessed that letter before. Otherwise, a guess is `REJECTED`.

You can guess as many times as you want, but if the number of `REJECTED` guesses is more than or equal to K, the game is over.

Input

The input consists of multiple test cases.

The first line of input contains an integer T ($1 \le T \le 100$), the number of test cases. The succeeding lines contain the test cases.

Each test case is in a single line and contains the following data, separated by single spaces:

K: the number of `REJECTED` guesses must not exceed this value ($1 \le K \le 20$)

S: the secret phase. The length of S is from 1 to 100.

R: the sequence of letters that you guess. The number of letters in R is from 1 to 100.

Output

For each test case, display in a single line `Correct` if the number of `REJECTED` is smaller than K and all characters of S are displayed. Otherwise, display `Wrong`.

Sample Input

```
4
1 a a
3 hcmus hcipomsu
3 problems prsblemio
2 acmicpc acmicpc
```

Sample Output

```
Correct
Wrong
Correct
Wrong
```

Explanation:

For the second test case, there are three letters you guess but do not exist in *S*, including `'i'`, `'p'`, and `'o'`. Therefore, the number of `REJECTED` guesses is 3.

For the fourth test case, the first time you guess `'c'` is `ACCEPTED`. However, you guess this letter up to three times. Therefore, the last two times you guess `'c'` are `REJECTED`.

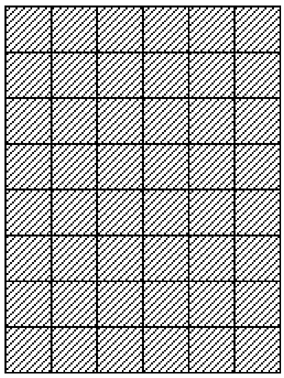
Problem F
Memory Mosaic
Time Limit: 1 second

Mid-Autumn Festival will be a great event with many beautiful scenes. Hugo decides that take a lot of photos at the Festival. Currently, he is preparing a large picture frame *F* of the size *N*×*M* with gridlines to display his favorite pictures taken at the Festival.

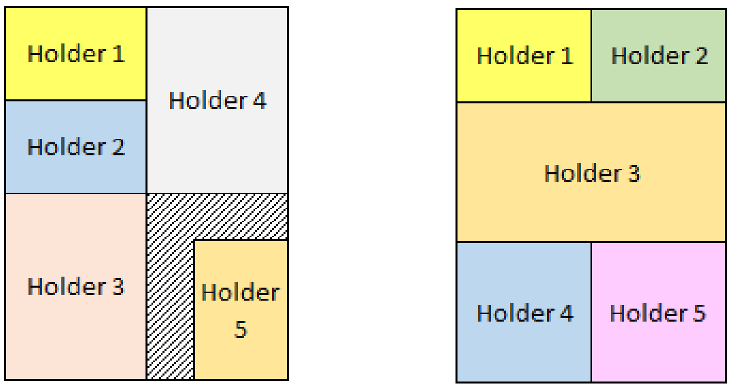
Hugo intends to put *K* picture holders on the picture frame *F*. Each picture holder has a rectangular shape and its dimensions are integers. A picture holder should not be rotated and should be aligned with the gridlines on the frame *F*. There is no occlusion between a picture holders and others.

After selecting his favorite photos, he will put each of them into an appropriate picture holder on the frame. Then he will have a Memory Mosaic with his pictures.

Example:



An empty picture frame *F* of the size 7 × 6



Two different ways to put 5 picture holders on frame *F*.
Regions with diagonal stripes are empty.

The question is how many different layouts of Memory Mosaic that Hugo can make?

Input

The input contains one line with exactly 3 integer numbers *N*, *M*, and *K* ($1 \le N; M \le 100$, $1 \le K \le 5$), separated by a single space.

Output

Display only one integer corresponding to the maximum number of layouts for *K* picture holders to be placed on the picture frame *F*.

Sample Input 1 / Sample Output 1

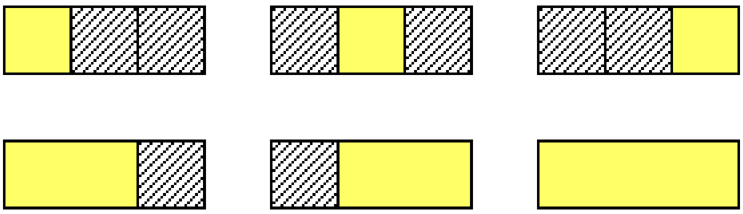
`1 1 2`	`0`

Explanation: There are no ways to stick 2 picture holders on a frame with only one cell.

Sample Input 2 / Sample Output 2

`1 3 1`	`6`

Explanation: There are 6 different ways to put one picture holder on a frame as below (cells with diagonal stripes are empty):



Sample Input 3 / Sample Output 3

`1 2 2`	`2`

Explanation: There are 2 different ways to put two picture holders on a frame as below:

Problem G
Lantern Exhibition
Time Limit: 4 seconds

We are preparing for Lantern Exhibition in Mid-Autumn Festival. There are N lanterns to be displayed in the exhibition. Lanterns will be placed into K exhibition rows. Initially, all exhibition rows are empty.

We know the height of each lantern. Assume that the heights of lanterns are all distinct, and denoted from 1 to N (N is the highest).

For each lantern, from the first to the N^{th} one, we need to decide which row to put it or ignore that lantern. A lantern can be put for display in an exhibition row if there is no lantern already in that row with higher height. By this way, we can ensure that all *selected* lanterns in the exhibition will be at least partially visible to all visitors from the entrance of Lantern Exhibition.

Our task is to determine the maximum number of lanterns that can be displayed in Lantern Exhibition.

Input

The input consists of multiple test cases. The first line of input contains an integer T ($1 \le T \le 100$), the number of test cases. The succeeding lines contain the test cases.

Each test case consists of two lines:

- Line 1: there are two positive integers N and K ($1 \le K \le N \le 5000$).
- Line 2: there are N space-separated integers denoting the heights of N lanterns (in the order from the first to the N^{th} lantern).

Output

The output should contain the solutions to all the test cases, in the order of the test cases in the input. There should NOT be any blank line in the output. For each test case, print a single line containing the maximum number of lanterns to be displayed in Lantern Exhibition.

Sample Input	Sample Output
3 3 1 1 3 2 5 2 2 1 4 5 3 5 1 5 1 2 3 4	2 5 4

Problem H
Arithmetic
Time Limit: 1 second

Vivian will be awarded the most beautiful lantern in the Mid Autumn Festival if she can quickly win the Arithmetic Quiz for Smart Kids.

Vivian is given three positive integers *A*, *B*, and *C*. She should answer quickly if she can create a new positive integer *K* with the three given numbers using common arithmetic operators, including `+` (`add`), `-` (`subtract`), `×` (`multiply`), and `/` (`divide`), and parentheses?

Vivian can freely put the three numbers *A*, *B*, and *C* in any order. Parentheses, including `(` and `)`, can be used to group numbers and operations.

Input

The input consists of multiple test cases. The first line of input contains an integer T ($1 \le T \le 100$), the number of test cases. Each of the following T lines contains a test case. Each test case consists of four positive integers A, B, C, and K ($0 < A, B, C < 10$ and $1 < K < 30$)

Output

The output should contain the solutions to all the test cases, in the order of the test cases in the input. There should NOT be any blank line in the output. For each test case, display `Possible` if Vivian can find a way to create *K* from *A*, *B*, and *C*. Otherwise, display `Impossible`.

Sample Input	Sample Output
4 2 6 1 12 1 2 3 4 7 5 5 5 2 5 8 20	Possible Possible Impossible Possible

Explanation:

$A = 2, B = 6, C = 1, K = 12$:	As $2 \times 6 \times 1 = 12$, the result is `Possible`.
$A = 1, B = 2, C = 3, K = 4$:	As $2 + 3 - 1 = 4$, the result is `Possible`.
$A = 7, B = 5, C = 5, K = 5$:	As we cannot find any way to create $K = 5$ with $A = 7$, B = 5, and $C = 5$, the result is `Impossible`.
$A = 2, B = 5, C = 8, K = 20$:	As $(5 / 2) \times 8 = 20$, the result is `Possible`.

Problem I
Buffet
Time Limit: 3 seconds

Mooncakes are delicious. There are so many kinds of mooncakes. Do you want to enjoy all of them? Let us join the Mooncake Buffet.

The organizers of the Mooncake Buffet prepare N different tables, labeled from 1 to N, in the Garden of the Moon. The i^{th} table can contain at most q_i pieces of mooncakes of the i^{th} type. Initially, all tables are empty.

The Mooncake Buffet begins at time $t = 1$ and will finish at time $t = M + 1$. At a certain time instance t ($1 \leq t \leq M$), several tables will be refilled. When the i^{th} table is refilled, it will contain exactly q_i pieces of mooncakes (regardless of the number of mooncake pieces available on the table). At every time instance t, visitors will take R pieces of mooncakes from every table. Sometimes there are not enough R pieces of mooncake pieces on a table for visitors to take.

Tommy is an organized person and he always wants to prepare a plan for any activity. He has a list of his favorite types of mooncakes. At a time instance t, he will take exactly e_i pieces of mooncakes from the i^{th} table at once (there must be at least e_i pieces of mooncake on the table) and eat these pieces until after time instance $t + K - 1$. Then he can continue to look for another favorite type of mooncakes from time instance $t + K$.

Can Tommy enjoy all of his favorite types of mooncakes? Please remember that all visitors should leave the Mooncake Buffet right after time instance M.

For your convenience, at each time instance t ($1 \leq t \leq M$), the following actions may happen in this sequence:

- Visitors try to take up to R pieces from every table.
- One table may be refilled.
- Tommy tries to take mooncake pieces from one table (if there are enough pieces of his preference).

Input

The input consists of multiple test cases. The first line of input contains an integer T ($1 \le T \le 20$), the number of test cases.

For each test case:

- The first line contains four integers N, M, K, and R ($1 \le N \le 8$, $1 \le M \le 10000$, $1 \le K \le M$ and $1 \le R \le 10^9$)
- The second line contains N integers q_i, the capacity of the i^{th} table ($1 \le q_i \le 10^9$). When the i^{th} is refilled, it will contain exactly q_i mooncake pieces.
- The third line contains M integers a_t, the index of the table that will refilled at time instance t ($0 \le a_t \le N$). $a_t = 0$ if there is no table refilled at time instance t.
- The fourth line contains an integer Q, the number of favorite lists ($1 \le Q \le 100$)
- Each of the next Q lines contains information of a favorite list. Each favorite list has N integers e_i - the number of mooncake pieces that Tommy intends to eat from the i^{th} table ($1 \le e_i \le 10^9$).

Output

For each list among the Q favorite lists, display `Yes` if there is a plan to eat all types of mooncakes in the favorite list. Otherwise, display `No`.

Sample Input	Sample Output
1 2 6 2 1 4 1 1 2 0 0 0 1 2 1 1 3 1	Yes No

Explanation:

There are 2 tables. The first table (with capacity $q_1 = 4$) will be refilled at time instance $t = 1$ and $t = 6$ (the last time instance). The second table (with capacity $q_2 = 1$) will be refilled only one time at time instance $t = 2$. At every time instance, visitors will take $R = 1$ piece of mooncake from every table (if there is any piece on the table).

Here is the states of all tables at each time instance (without Tommy's actions)

Time Instance	Number of mooncake pieces		Explanation
	1st table	2nd table	
$t = 1$	4	0	Tables are initially empty. Visitors cannot take any mooncake piece. Table 1 is refilled.
$t = 2$	3	1	Table 2 is refilled. Visitors take 1 piece from the table 1
$t = 3$	2	0	No table is refilled. Visitors take 1 piece from each table.
$t = 4$	1	0	No table is refilled. Visitors take 1 piece from table 1. Nothing is left on table 2.
$t = 5$	0	0	No table is refilled. Both tables are empty.
$t = 6$	4	0	Table 1 is refilled

For the first favorite list, Tommy wants to eat 1 piece of mooncake from each table. If he decides to take 1 piece from the first table at time instance $t = 1$, he should eat that piece until time instance $t = 3$. Then he cannot take any piece from the second table as another visitor already takes the only piece on the second table right at time instance $t = 2$. His plan should be like this: Tommy first takes 1 piece of mooncake from the second table at time instance $t = 2$, eats it until $t = 4$, then takes 1 piece of mooncake from the first table, eats it until $t = 6$, the leave the Mooncake Buffet at $t = 7$ (on time!).

For the second favorite list, Tommy wants to eat 3 pieces from the first table, and 1 piece from the second table. It can be verified that he cannot fulfill his dream!

Problem J
Lunar Crystal
Time Limit: 1 second

In the Mid-Autumn Festival, there are a lot of attractions and games. Quinlan is playing the game Lunar Crystal. This is actually a treasure hunt game and it is very easy:

Given a 2D grid of size $M \times N$, with cells with Lunar Crystal (denoted by '.') or cells with bombs (denoted by 'x'). Quinlan can start at any cell that contains no bomb; and at a cell he can move in any of the directions, north, west, south or east, if the destination cell contains no bomb.

He can pass through any cell only once. Quinlan has to visit and collect all Lunar Crystal in order to win. Please compute the number of all possible routes.

Routes are considered different even when: (i) same starting cell, different traversal paths; or (ii) same traversal path, different starting cells.

Input

The input consists of multiple test cases.

The first line of input contains an integer T $(1 \le T \le 30)$, the number of test cases.

Each test case starts with a line containing two integers, M and N $(1 \le M \le 100,\ 1 \le N \le 8)$. Each of the next M lines contain a string of N characters describing the configuration of the grid. 'x' denotes a cell with bombs and '.' denotes a cell with Lunar Crystal.

Output

For each test case, print one line containing the total number of possible routes for the corresponding case. As this number can be quite large, it should take modulo 500 000 003.

Sample Input	Sample Output
2 2 2 xx .x 2 2 x. ..	1 2

Problem K
Business Intelligence
Time Limit: 4 seconds

In Mid-Autumn Festival, visitors not only enjoy attractions, buffet, or game shows, but also spend a lot of money to buy different products, gifts, souvenirs, etc. It would be wise to analyze transactions from previous years to predict which product a visitor may buy together with another product to increase the revenue for Festival Organizer.

In annual festival, there are no more than 10000 products. Each product has a unique product identifier, which is a positive integer that is no more than 10^9.

There are N customers ($1 \le N \le 10000$). For the i^{th} customer, we have his or her history list L_i of all products that he or she purchased in previous Mid-Autumn Festivals. There are no more than 1000 items in each history list.

Giving a threshold T ($0 < T < 10^6$), two distinct products $\{x, y\}$ are considered to be a "common pair" if there are at least T customers bought these two products.

It is our task to identify all "common pairs" from the data on transactions in previous Mid-Autumn Festivals provided by the Festival Organizer.

Input

The first line of input contains two positive integers N and T ($1 \le N \le 10000$, $1 \le T \le 10^6$).

The i^{th} line in the following N lines contains an integer k_i ($0 \le k_i \le 1000$), the number of products bought by the i^{th} customer, following by k_i positive integers that are the identifiers of these products. The identifier of a product is a positive integer with the value from 1 to 10^9.

Output

Display a single integer that is the total number of "common pairs" found in the transactions in previous Mid-Autumn Festivals provided by the Festival Organizer.

Sample Input	Sample Output
5 2 3 100 987 1 2 987 1 0 1 100 5 4 987 12 4 2015	1

Explanation: there is only one "common pair" of products, that is $\{1, 987\}$.
The two pairs of products $\{x, y\}$ and $\{y, x\}$ are identical and should be counted only once.