

MARCH09	<a href="#">March 2009 (Contest I)</a>	28 Feb 2009 22:00:00	14 days 2 hours	347
---------	--	-------------------------	-----------------	-----

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » Paying up

## Paying up Problem Code: A1

In the mysterious country of Byteland, everything is quite different from what you'd normally expect. In most places, if you were approached by two mobsters in a dark alley, they would probably tell you to give them all the money that you have. If you refused, or didn't have any - they might even beat you up.

In Byteland the government decided that even the slightest chance of someone getting injured has to be ruled out. So, they introduced a strict policy. When a mobster approaches you in a dark alley, he asks you for a specific amount of money. You are obliged to show him all the money that you have, but you only need to pay up if he can find a subset of your banknotes whose total value matches his demand. Since banknotes in Byteland can have any positive integer value smaller than one thousand you are quite likely to get off without paying.

Both the citizens and the gangsters of Byteland have very positive feelings about the system. No one ever gets hurt, the gangsters don't lose their jobs, and there are quite a few rules that minimize that probability of getting mugged (the first one is: don't go into dark alleys - and this one is said to work in other places also).

### Input

The first line contains integer  $t$ , the number of test cases (about 100). Then  $t$  test cases follow. Each test case starts with  $n$ , the number of banknotes in your wallet, and  $m$ , the amount of money the muggers asked of you. Then  $n$  numbers follow, representing values of your banknotes. Your wallet does not hold more than 20 banknotes, and the value of a single banknote is never more than 1000.

### Output

For each test case output a single line with the word 'Yes' if there is a subset of your banknotes that sums to  $m$ , and 'No' otherwise.

### Example

**Input:**

3 3

1

1

1

5 11

1

2

4

8

16

5 23

1

2

4

8

16

5 13

1

5

5

10

10

20 132

17

6

4

998

254

137

259

153

154

3

28

19

123

542

857

23

687

35

99

999

**Output:**

Yes

Yes

Yes

No

Yes

**Explanation:** For example, in the last case you have to pay up, since:  $6+3+123=132$ .

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » Johnny and the Beanstalk

## Johnny and the Beanstalk Problem Code: A2

One evening Johnny found some funny looking beans in his grandfather's garden shed, and decided to plant one of them. Next morning, to his surprise he found an enormous beanstalk growing in his back yard. Undaunted by its size, he decided to count its leaves.

You must know that beanstalks in Byteland grow in a very special way. At the lowest (1st) level, there is exactly one stem. At any level(including the 1st), a stem can end (forming exactly one leaf), or branch into exactly two stems which grow into the next level, following the same rules.

Johnny believes he has managed to count the number of leaves at each of the levels of the beanstalk. However, you must know that before he began to count, Johnny ate one or two of the other beans he found in his grandfather's shed, and that's why he is not quite sure of his results. Please verify whether Johnny's results may possibly be correct, at least in theory.

### Input

The input starts with a line containing integer  $t$ , the number of test cases ( $1 \leq t \leq 20$ ). The descriptions of exactly  $t$  test cases follow.

Each test case starts with an integer  $k$ , representing the number of levels of the beanstalk ( $1 \leq k \leq 10^6$ ). The next  $k$  non-negative space-separated integers (not greater than  $10^6$ ) represent the number of leaves of the beanstalk at successive levels, starting from level 1.

### Output

For each test case, output a line containing exactly one of the words 'Yes' or 'No', depending on whether a beanstalk having the stated leaf counts can grow in accordance with the Bytelandian rules.

### Example

**Input:**

2

3

0 1 2

3

0 0 3

**Output:**

Yes

No

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » The Rise and Fall of Power

## The Rise and Fall of Power Problem Code: A4

Johnny was asked by his math teacher to compute  $n^n$  ( $n$  to the power of  $n$ , where  $n$  is an integer), and has to read his answer out loud. This is a bit of a tiring task, since the result is probably an extremely large number, and would certainly keep Johnny occupied for a while if he were to do it honestly. But Johnny knows that the teacher will certainly get bored when listening to his answer, and will sleep through most of it! So, Johnny feels he will get away with reading only the first  $k$  digits of the result before the teacher falls asleep, and then the last  $k$  digits when the teacher wakes up.

Write a program to help Johnny to compute the digits he will need to read out.

### Input

The first line contains  $t$ , the number of test cases (about 30000). Then  $t$  test cases follow.

Each test case consists of one line containing two numbers  $n$  and  $k$  ( $1 \leq n \leq 10^9$ ,  $1 \leq k \leq 9$ ). It is guaranteed that  $k$  is not more than the number of digits of  $n^n$ .

### Output

For each test case, print out one line containing two numbers, separated by a space, which are the first and the last  $k$  digits of  $n^n$ .

### Example

#### Input

2

4 2

9 3

**Output**

25 56

387 489

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » Some More Homework

## Some More Homework Problem Code: A6

In the computer science class at the school in Byteland, the teacher handed out the following assignment as homework:

"For an integer  $n$ , let  $b_n$  denote the bit parity of the binary representation of  $n$ , i.e.  $b_n=0$  if  $n$  has an even number of ones when written down in the binary system, and  $b_n=1$  otherwise. The numbers  $b_n$ , for  $n \geq 0$ , form an infinite sequence of bits (zeros and ones). Given a sequence  $c=(c_0, \dots, c_{p-1})$  of  $p$  bits, find the first occurrence of sequence  $c$  as a subsequence of  $b$  (i.e., the smallest value of index  $k$  such that for all  $i$  between 0 and  $p-1$ , we have  $c_i = b_{i+k}$ )."

And the teacher gave his students several short sequences  $c$ , asking them to provide the answers next day. Most, as expected, wrote programs to solve the task. Only Johnny computed the results by hand, claiming (quite correctly) that it was quicker that way. The teacher, slightly exasperated, decided to teach Johnny a lesson, and prepared a harder assignment, just for him.

"Given a sequence  $c=(c_0, \dots, c_{p-1})$  of  $p$  bits, for each  $s$  between 0 and  $p-1$ , compute the first occurrence of the prefix  $(c_0, \dots, c_s)$  of sequence  $c$  as a subsequence of  $b$ ."

And to be doubly sure that Johnny does his homework using a computer, the teacher gave him some really long sequences to deal with. Now, Johnny is in a bit of a spot, because he has never bothered to learn to program. Please help him out!

**Input**

The first line of input contains a positive integer  $t < 10$ , describing the number of tests. Exactly  $t$  test cases follow.

Each test case is given in two lines. The first line contains integer  $p$  ( $1 \leq p \leq 10^6$ , the length of sequence  $k$ ). The next line contains exactly  $p$  space-separated numbers (0 or 1), denoting successive elements of sequence  $c$ .

## Output

For each test case, print a line containing exactly  $p$  space-separated numbers, corresponding to the indexes of the first occurrence of successive prefixes of  $c$  as subsequences of  $b$ . All indexes are numbered starting from zero. If there is no such occurrence, output -1.

## Example

### Input:

```
1
9
1 0 0 1 0 1 1 1 0
```

### Output:

```
1 2 4 4 8 8 8 -1 -1
```

### Explanation:

The first 16 elements of  $b$  are: 0 1 1 0 1 0 0 1 1 0 0 1 0 1 1 0

The prefix '1' first appears in  $b$  at index 1,

The prefix '1 0' first appears in  $b$  at index 2,

The prefixes '1 0 0' and '1 0 0 1' first appear in  $b$  at index 4,

... and so on.

There are no occurrences of '1 0 0 1 0 1 1 1' in sequence  $b$

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » The Guessing Game

## The Guessing Game Problem Code: A3

Alice and Johnny are playing a simple guessing game. Johnny picks an arbitrary positive integer  $n$  ( $1 \leq n \leq 10^9$ ) and gives Alice exactly  $k$  hints about the value of  $n$ . It is Alice's task to guess  $n$ , based on the received hints.

Alice often has a serious problem guessing the value of  $n$ , and she's beginning to suspect that Johnny occasionally cheats, that is, gives her incorrect hints. After the last game, they had the following little conversation:

- [Alice] Johnny, you keep cheating!
- [Johnny] Indeed? You cannot prove it.
- [Alice] Oh yes I can. In fact, I can tell you with the utmost certainty that in the last game you lied to me at least \*\*\* times.

So, how many times at least did Johnny lie to Alice? Try to determine this, knowing only the hints Johnny gave to Alice.

## Input

The first line of input contains  $t$ , the number of test cases (about 20). Exactly  $t$  test cases follow.

Each test case starts with a line containing a single integer  $k$ , denoting the number of hints given by Johnny ( $1 \leq k \leq 100000$ ). Each of the next  $k$  lines contains exactly one hint. The  $i$ -th hint is of the form:

operator  $l$  logical\_value

where operator denotes one of the symbols  $<$ ,  $>$ , or  $=$ ;  $l$  is an integer ( $1 \leq l \leq 10^9$ ), while logical\_value is one of the words: Yes or No. The hint is considered correct if logical\_value is the correct reply to the question: "Does the relation:  $n$  operator  $l$  hold?", and is considered to be false (a lie) otherwise.

## Output

For each test case output a line containing a single integer, equal to the minimal possible number of Johnny's lies during the game.

## Example

Input:

3

2

$< 100$  No

$> 100$  No

3

$< 2$  Yes

$> 4$  Yes

= 3 No

6

< 2 Yes

> 1 Yes

= 1 Yes

= 1 Yes

> 1 Yes

= 1 Yes

**Output:**

0

1

2

**Explanation:** for the respective test cases, the number picked by Johnny could have been e.g.  $n=100$ ,  $n=5$ , and  $n=1$ .

[Home](#) » [Compete](#) » [March 2009 \(Contest I\)](#) » Place the mines!

## Place the mines! Problem Code: AX

The following a tie breaker problem. The best answer will receive one point. All other successful answers will be scored on a curve and receive a fraction of a point based on how close they come to the best answer.

You are given a large square area with an edge length of  $2^n$ , subdivided into unit squares (fields). Some mines have already been planted on certain fields, and we would like to plant as few more mines as possible, so as to destroy all of the fields of the area (assuming all the mines explode simultaneously). For each mine, the fields it destroys are given as follows. For any  $k$ ,  $0 \leq k \leq n$ , we can partition the considered area into  $2^n$  rectangles of size  $2^k \times 2^{n-k}$ . The mine is assumed to destroy a square if and only if the mine and the square belong to the same rectangle in the considered partition, for some value of  $k$ .

For example, if  $n=3$ , the mine "\*" placed on the square at position (1,1) will destroy all the fields which are not marked with dots below:

```

.#.....
.#.....
.#.....
.#.....
##.....
##.....
#*#####
####....
```

## Input

The first line will contain integer  $8 \leq n \leq 16$ , (where  $2^n$  is the size of the area). The second line will contain  $0 \leq m < 2^n$ , the number of already planted mines. The next  $m$  lines will contain the 0-based x and y coordinates of the existing mines.

## Output

The first line should contain  $t$  - the number of mines to be added. The next  $t$  lines should contain the coordinates of the added mines.

## Scoring

For each test case, you will receive  $(2^n - m)/t - 1$  points, provided you destroy all the fields of the area.

## Example

Input:

2

1

3 3

**Output:**

2

0 1

2 2

You will receive  $(4-1)/2 - 1 = 0.5$  pts for such a mine placement.

**Note:** There will be some tests with no initially placed mines, and some tests with initial placements covering most of the area pretty well. Good luck!

APRIL09	<a href="#">April 2009 (Contest II)</a>	01 Apr 2009 15:00:00	14 days	248
---------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » Squash the Bugs

## Squash the Bugs Problem Code: B5

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

Bugs have gotten into The Chef's kitchen! Help him trap them all and he'll make you a batch of his famous chocolate chip cookies. You are given a given a square map of the kitchen divided into tiles, and in each tile sits some known number of bugs. You also have a square trap, which can be dropped to cover a certain number of tiles (the trap may only cover tiles from within the map, and must be aligned to the borders of the kitchen). However, the trap only catches bugs from one of the tiles which it has covered, having a minimum number of bugs on it. For all possible positions at which the trap can be dropped, determine number of bugs that will be caught.

### Input

Two numbers,  $0 < n \leq 1000$  (size of the map), and  $0 < k \leq n$  (size of the trap), followed by  $n$  rows with  $n$  numbers, determining the number of bugs on each tile. The number of bugs on each tile will fit in an signed 32-bit integer.

### Output

You should output  $n-k+1$  rows with  $n-k+1$  numbers in each row.

### Example

**Input:**

4 2

0 1 2 3

4 5 6 7

8 9 0 1

2 3 4 0

**Output:**

0 1 2

4 0 0

2 0 0

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » Battleship V

## Battleship V Problem Code: B3

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

In the game of "BattleShip V", you control a cannon which is attacking a large enemy battleship, armed with many guns. Your goal is to destroy as many of the guns as possible.

The battle field is a 2D Cartesian grid, where your cannon is located at the origin.

The enemy battleship is a horizontal line segment located from the coordinates (X1 , Y) to (X2 , Y). There are exactly (X2 - X1 + 1) guns on the ship, located at the integer points (X1, Y), (X1+1, Y), ..., (X2, Y).

However, the problem is, you cannot always fire at a gun. There are supernatural rocks located at all points of the battlefield whose X and Y coordinates are both integers. In order to fire successfully at an enemy's gun, the line connecting your cannon and that gun must not go through any rocks.

How many guns you successfully destroy?

### Input

The first line contains t, the number of test cases (about 100). Then t test cases follow.

Each test case consists of one line containing three numbers Y, and X1, X2 ( $0 < |Y| \leq 2100000000$ ,  $-2100000000 \leq X_1 \leq X_2 \leq 2100000000$ ).

## Output

For each test case, output the number of the enemy battleship's guns that your cannon can destroy.

## Example

### Input

1

2 -2 1

### Output

2

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » Spaghetti Monsters

## Spaghetti Monsters Problem Code: B1

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

Spaghetti Monsters have stolen The Chef's golden spoon! The Chef has a map, on which are marked: his location, the location of the golden spoon, and the locations of the spaghetti monsters. The map is in fact rectangular, consisting of square fields. Each field on the map, except for those adjacent to the boundary, is adjacent to 8 other fields -- fields which share a side or corner are assumed to be at a distance of 1 from each other, and it is possible to move between them directly. The Chef is a bit scared of spaghetti monsters and would prefer not to approach them too closely... Help him compute the minimum distance up to which he must approach some spaghetti monster, so as to find the golden spoon.

### Input

The first line of input contains two numbers  $1 \leq n, m \leq 1000$ , where n denotes the height of the map, and m its width.

The map is given in the next n lines, each of which consists of m characters:

The unique field represented by the character '@' is the location of The Chef.

The unique field represented by the character '\$' is the location of the golden spoon.

A field represented by the character 'D' is the location of a spaghetti monsters.  
All the remaining fields are represented by the character '.' and denote empty positions.

## Output

The output should contain exactly one number, equal to the minimum distance, at which  
The Chef must find himself from one of the spaghetti monsters in order to reach the golden  
spoon.

## Example 1

### Input:

7 5

....\$

.....

.....

D...D

.....

.....

@....

### Output:

2

## Example 2

### Input:

7 5

....\$

.....

.....

DDDDD

.....

.....

@....

#### Output:

0

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » [Bubbles](#)

## Bubbles Problem Code: BX

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

The following a tie breaker problem. The best solution will receive one point. All other successful answers will be scored on a curve and receive a fraction of a point based on how close they come to the best answer.

Let's have some fun with soap bubbles... We drive several sticks vertically into a flat surface (so that they look like points from above), and spread a thin film of soap over all of them. The soap film tends to minimize the area of its surface, and eventually becomes very thin, with surfaces turning into lines spread over all points. When everything has settled down, we can assume that the "bubble" we have is in fact a set of line segments, which connect all of our sticks into one network (possibly via some intermediate points which can also be the endpoints of lines).

## Input

First,  $1 \leq n \leq 1000$ , the number of starting points to connect. Then,  $n$  pairs of numbers follow, representing the coordinates of sticks which the soap film will connect.

## Output

First, you should output the number  $m$  of intermediate points you create. Then, the next  $m$  pairs of numbers are coordinates of intermediate points. Then, you should output  $e$ , the number of soap-bubble lines you create. Then, the next  $2 \cdot e$  numbers represent the indexes of points which are connected by the given line. Each index  $i$  should be a number  $0 \leq i < n+m$ . If  $i < n$ , then it represents the  $i$ -th input point (stick). If  $i \geq n$ , then it corresponds to the  $(i-n)$ -th intermediate point.

## Scoring

Your goal is to minimize the score obtained in this problem.

As longer lines tends to get thinner, for each line segment of length  $t$ , you will receive  $t / \max(1, \ln(t))$  penalty points, where " $\ln$ " is the logarithm in the natural base. You will additionally receive  $\ln(m+1)$  penalty points if your solution uses  $m$  intermediate points.

If the soap film given at output does not span all the input points (connecting them into a network), your solution will be judged as incorrect.

## Example

### Input:

3

0.0 0.0

0.0 3.0

4.0 0.0

### Output:

1

1.0 1.0

3

0 3

1 3

2 3

**Score:**

7.090148

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » A Christmas Pizza Puzzle

## A Christmas Pizza Puzzle Problem Code: A7

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

Your grandfather is known in the family for his unconventional sense of humour. Since he made his fortune on the network of pizza parlours<sup>a)</sup>, he has been giving expensive gifts to the members of the family every Christmas. Receiving the gift, however, always involves solving a more or less fancy mathematical puzzle.

This year, your gift was locked in a safe, self-made by your grandfather. The safe was secured with a two digit code: in order to open it you had to provide the correct integer number from the range [0,99]. A bit disgruntled that this year's puzzle would apparently be solved with the force (and not even too brute), you were just about to start examining the numbers, when you heard grandfather's voice - "Beware, kid! After two unsuccessful attempts the content will be annihilated!" "Then how am I supposed to solve this, Grandpa?" - you asked. "Go and have a better look at the package" - grandfather replied.

Indeed, at the bottom of the package you found a city map on which were indicated all the pizza parlours belonging to your grandfather, as well as a long sheet of paper with a sequence of positive integers. After analysing the data for a while you realized that the number of integer values is equal to the number of pizza restaurants... Recognizing that you still have too little input, you went to ask grandfather for further guidance.

"Ah, all right, I'll solve the riddle for you" - grandfather relented. "For each pizza parlour you have to draw such an axis-aligned square that it is centered at the restaurant and half of the length of its side is equal to the corresponding value from the sheet. Then you have to count all the streets on the map which intersect with the square, and write down the **result of calculations**. The result for the last pizza parlour will be your magic number."

You were a bit surprised that grandfather ordered you to draw squares and count streets for all the restaurants (and there were many), since the code would only depend on the last one. Nonetheless you willingly took on the job, drew the square for the last pizza parlour, counted the streets and entered the number. Unfortunately the safe didn't open.

You rushed to grandfather and told him about the first failed attempt. "Oh dear, did I forget to mention?" - the old man was slightly taken aback. "For each restaurant but the first one,

you have to modify both its coordinates by bit-xor'ing them with the result of calculations for the directly preceding restaurant, before drawing the square!". Will you be able to claim the gift now? You have only one attempt left!

## Input

The first line of input contains two integers  $n$  and  $m$  ( $0 \leq n \leq 5000$ ,  $1 \leq m \leq 50000$ ), the number of streets in the capital city and the number of pizza parlours, respectively. The following  $n$  lines contain four integers  $x_1$ ,  $y_1$ ,  $x_2$  and  $y_2$ , each ( $0 \leq x_1, y_1, x_2, y_2 \leq 1000$ ). Each of these lines defines the location of one of the streets, with  $(x_1, y_1)$  representing one of the endpoints and  $(x_2, y_2)$  the other one. The streets can be assumed to be straight line segments. Eventually there are  $m$  lines with three integers  $x$ ,  $y$  and  $r$  each ( $0 \leq x, y \leq 1000$ ,  $1 \leq r \leq 20$ ). Each of these lines defines  $x$  and  $y$  coordinates of one of the pizza parlours and the corresponding number from the sheet attached to your gift, respectively.

You may find it useful to know that:

- A street can have identical endpoints<sup>b)</sup>.
- Any two distinct streets have at most one common point.
- When counting intersecting streets, you should assume that the boundary of the square belongs to the square.
- The configuration of the streets guarantees that any square satisfying  $0 \leq x, y \leq 1000$ ,  $1 \leq r \leq 20$  will contain at most 99 intersecting streets.
- Your grandfather is 78 and claims that Fortran is the best language for solving his Christmas puzzles.

## Output

The output should contain a single integer value - the code that opens the safe.

## Example

**Input:**

5 2

2 2 4 6

2 2 7 4

4 6 7 4

7 4 10 3

7 4 10 5

10 4 2

0 0 1

**Output:**

2

## Notes

- a) This should not surprise anyone. As the IT sector is extremely well-developed in Byteland, pizza network owners are among the richest people there.
- b) For some inscrutable reasons such streets are called squares. Not to be mixed up with the squares you draw!

[Home](#) » [Compete](#) » [April 2009 \(Contest II\)](#) » Packing the Boxes

## Packing the Boxes Problem Code: A5

This problem was part of the [CodeChef April Challenge](#). All user submissions for this contest problem are publicly available [here](#).

Shaheen has bought some gifts for a friend, which are wrapped up in several boxes of different sizes (all of which are full). She will need to carry the gifts a long way to her friend's house, so she would prefer to add some extra packing, and accommodate everything in one extra box. Moreover, to avoid damaging the contents, she does not wish to place more than two boxes directly within any box; however, boxes can be placed within boxes which contain other boxes, etc. A box which is used for holding two boxes of sizes  $a$  and  $b$  will have size  $a+b$ , and will cost Shaheen  $a+b$  coins at the local store. Please help Shaheen determine the minimum cost required to achieve the complete packing, and the number of different possible packings (arrangements of boxes within each other) having such a minimal cost.

## Input

The first line of input is  $n \leq 2000$ , the number of boxes with Shaheen's gifts. The next  $n$  lines of input contain one positive integer each, not greater than  $10^6$ , representing the sizes of the successive boxes.

## Output

Print to output exactly 2 numbers separated by spaces: the cost of the optimal solution, and the number of distinct ways of achieving this solution (modulo  $10^9$ ).

## Example

**Input:**

5

3

2

3

5

1

**Output:**

313

**Explanation:** The three solutions leading to cost 31 are as follows:

1) pack the 2nd and the 5th box together, pack the resulting box together with the 1st box, and pack the result together with an additional box used for packing the 3rd and 4th boxes,

or

2) pack the 2nd and the 5th box together, pack the resulting box together with the 3rd box, and pack the result together with an additional box used for packing the 1st and 4th boxes,

or

3) pack the 2nd and the 5th box together, pack the resulting box together with the 4th box, and pack the result together with an additional box used for packing the 1st and 3rd boxes.

MAY09	<a href="#">May 2009 (Contest III)</a>	01 May 2009 15:00:00	10 days	230
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » Healthy dinner party

## Healthy dinner party Problem Code: C4

The Chef is having a dinner party and invited over all his friends. His guests being fairly health conscious have **exact protein requirements**, and The Chef wishes to oblige them all.

The Chef will cook dishes for each individual guest using the ingredients in his kitchen. Each ingredient has a specific amount of protein. The complete dish will have a protein content value equal to the sum of the protein contents of the individual ingredients. To cook a dish, The Chef can use any of the ingredients which appear on his shelf, **but only in the order which they appear on the shelf**. The same ingredient may appear multiple times, and can also be used as many times as it appears.

There are multiple ways to choose ingredients following the rules given above. However, The Chef is only interested in choosing the set of ingredients that appear first in a **lexicographically ordered** list of ingredients which satisfy the protein constraints. Your job is to write a program that helps The Chef figure out which dish to serve!

## Input

The first line of input contains  $t$ , the number of guests invited by The Chef (about 200).

Each test consists of three lines:

- The first line consists of one integer  $1 \leq k \leq 26$  (the number of **unique** ingredients on the shelf) and then  $k$  space-separated positive integers from the set  $\{1, 2, \dots, 15\}$  describing the protein content for each ingredient in an alphabetically sorted list of unique ingredients. (the first protein value corresponds with ingredient  $a$ , the second corresponds with the protein value for ingredient  $b$ , and so on).
- The second line contains  $L$  - a sequence of lower-case letters of the Latin alphabet (at most 1000) which signify the name of the ingredient.
- The third line contains one positive integer  $S$  which specifies the exact protein requirement of this guest ( $1 < S < 500$ ).

## Output

For each testcase either output the sequence of ingredients as described above, or the word '**IMPOSSIBLE**' if no such subsequence exists.

## Example

**Input:**

3

5 12 1 12 4 4

acccdadceb

2

3 5 4 6

abcbacbabcc

15

2 3 4

baba

7

**Output:**

IMPOSSIBLE

aaa

ab

**Comments:**

For the first guest we have five ingredients: a, b, c, d, e with protein values 12 1 12 4 4 respectively. To achieve a total protein value equal to 2 we need two ingredients b. But there is only one, thus the answer is IMPOSSIBLE.

For the second guest we can achieve a total protein value of 15 with the ingredients taken as: abc, bca, acb, cab, cba, bac, or aaa. Of these, the first according to lexicographic order is aaa.

For the third guest, out of the two possibilities, ab is the correct answer.

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » The powerful sum

## The powerful sum Problem Code: C2

Let us calculate the sum of k-th powers of natural numbers from 1 to n. As the result can be quite large, output the result modulo some integer p.

### Input

First  $t \leq 10$  - the number of test cases. Each test case consist of numbers:  $1 \leq n \leq 1000000000$ ,  $1 \leq k \leq 100$ ,  $1 \leq p \leq 1000000000$ .

### Output

For each test case, output the value:  $(1^k+2^k+\dots+n^k) \bmod p$ .

## Example

For input:

4

10 1 1000000

10 2 1000000

10 3 1000000

10 4 1000000

the correct output is:

55

385

3025

25333

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » Prime words

## Prime words Problem Code: C3

In the magic land of Mathtopia, the words of the language are written only using two symbols: ones and zeros.

A given word  $w$  is called "prime" if it cannot be written in the form of the concatenation of several copies of some shorter word. So, for example the words '100', '1100', and '001100' are prime, while the words '0101', '100100', '1111', and '101010' are not prime.

Your task is to calculate the number of prime words which can be built from exactly  $a$  ones and  $b$  zeros.

## Input

$t$  - the number of test cases. For each test case, two integers:  $1 \leq a \leq 10^9$ ,  $1 \leq b \leq 10^9$ .

## Output

For each test case, output the required answer modulo 531169.

## Example

**Input:**

1

2 2

**Output:**

4

Explanation: the 4 words from the example are: '0011','1100','0110','1001'.

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » Swapping mismatches

## Swapping mismatches Problem Code: C1

You are given a sequence  $w$  of integers. A mismatch is any such pair of neighbouring elements of sequence  $w[i]$  and  $w[i+1]$ , that  $w[i] > w[i+1] + 1$ . As long as there is any mismatch, you solve it by swapping the mismatching numbers. Given an input sequence, calculate one of the possible output mismatch-less sequences obtained by successively solving mismatches by swapping.

### Input

First -  $1 \leq t \leq 10$  - the number of tests. For each test: first -  $1 \leq n \leq 100000$ . Then,  $n$  nonnegative integers.

### Output

For each test, you should output exactly  $n$  integers.

## Example

**Input:**

2

4

4 3 2 1

4

4 3 1 2

**Output:**

4 3 2 1

1 4 3 2

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » Best board fill

## Best board fill Problem Code: CX

The following a tie breaker problem. The best solution will receive one point. All other successful answers will be scored on a curve and receive a fraction of a point based on how close they come to the best answer.

Given a large board (with holes) and set of tetris-like figures (an unlimited source, capable of generating any number of them), try to cover the board as exactly as possible. The source is capable of generating the following pieces:

1)

###

.#.

2)

##.

.##

3)

#.#

###

4)

###

#..

5)

..#

###

#..

6)

#.#

###

.#.

You are allowed to flip and rotate pieces before placing them on the board.

## Input

The first line contains two numbers -  $100 \leq n, m \leq 1000$  - the height and the width of the board. The next  $n$  lines, each containing  $m$  space-separated numbers, are the board description. The symbol '0' is a square which should be filled, the symbol '1' is a square which should not be filled.

## Output

First output the number of pieces used,  $k < 10^6$ . Then write  $k$  successive descriptions of the used pieces. Each description should be of the form:  $t$  - the number of full squares of the piece, followed by  $t$  pairs of integers denoting the coordinates of the respective squares (using 0-based offsets, with the top-left of input written as  $(0,0)$  ).

## Example

Input:

3 3

0 0 0

0 1 0

0 0 0

**Output:**

2

4

0 0

0 1

1 1

0 2

4

0 2

1 2

2 2

2 1

**Scoring**

For each square you will receive a penalty, calculated in the following way: Let  $a$  be the number a square should be covered by (either 0 or 1), and let  $b$  be the actual number of times the square has been covered by a # piece (pieces **may overlap**). If  $a > b$  (the square should have been covered, but was not), the penalty is **3**. If  $b > a$  (the square should not have been covered and was covered, or should have been covered, but was covered more than once), the penalty is  **$b-a$** . The total penalty is the sum of individual penalties, taken over all squares.

In the example the penalty is  $3+3+1+1 = 8$ .

The program will be run several times for different data sets and the overall score will be the mean of scores received.

**Tests**

All tests have been randomly generated, with 1 covering less than 40% of the total number of squares.

[Home](#) » [Compete](#) » [May 2009 \(Contest III\)](#) » An interesting subsequence

## An interesting subsequence Problem Code: C5

For two given sequences of integers, find the longest possible integer sequence which is a subsequence of both these sequences, and whose elements are integers in a (strictly) increasing order.

### Input

The first number is  $n$  ( $1 \leq n \leq 5000$ ), the length of the first sequence.

The next  $n$  integers are elements of the first sequence.

The next is  $m$  ( $1 \leq m \leq 5000$ ), the length of the second sequence.

The next  $m$  integers are elements of the second sequence.

### Output

On the first line, print a single number  $k$ , the length of the longest possible common subsequence that is increasing.

In the next line, print  $k$  space separated integers which is the answer. If there are multiple possible subsequences, any one would do.

### Example

For the input:

5

2 3 1 4 0

6

10 3 4 1 0 0

A correct answer is:

2

3 4

JUNE09	<u>June 2009 (Contest IV)</u>	01 Jun 2009 15:00:00	10 days	381
--------	-------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » Product of divisors

## Product of divisors Problem Code: D1

**A tutorial for this problem is now available on our blog. Click [here](#) to read it.**

Being in love with number theory, Johnny decided to take a number theory course. On the first day, he was challenged by his teacher with the following problem: given a number  $N$ , compute the product of its positive divisors. Johnny is desperate to impress his new teacher so he asks you for help.

In this problem, the divisors of  $N$  do not include the number  $N$  itself. For example, if  $N=12$ , the divisors of  $N$  (excluding  $N$ ) are 1, 2, 3, 4, and 6. Thus, the product of divisors is  $1 \times 2 \times 3 \times 4 \times 6 = 144$ . Since the result may be very large, if the result has more than 4 decimal digits, Johnny only needs to compute the **last 4 digits** of it.

### Input

The first line contains  $t$ , the number of test cases (about 300,000). Then  $t$  test cases follow.

Each test case contains a single integer  $N$  ( $1 \leq N \leq 500,000$ ) whose product of divisors needs to be computed.

### Output

For each test case, print a single line containing the corresponding result of that test case.

### Example

#### Input

6

3

4

12

25

957

10000

### Output

1

2

144

5

7493

0000

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » Count the squares

## Count the squares Problem Code: D6

Everyone knows what a square looks like. Mathematically, a square is a regular quadrilateral. This means that it has four equal sides and four equal angles (90 degree angles).

One beautiful day, Johnny eagerly examined the interesting properties of squares. He did not forget you, his best friend and a talented programmer and thus made a problem about squares to challenge your programming ability. The problem is: given a set of  $N$  points in the plane, how many squares are there such that all their corners belong to this set?

Now let's show Johnny your skill!

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case has the following form:

- The first line contains an integer  $N$ , the number of points in the given set ( $4 \leq N \leq 500$ ).
- Then  $N$  lines follow, each line contains two integers  $X, Y$  describing coordinates of a point ( $-50 \leq X, Y \leq 50$ ).

## Output

For each test case, print in a single line the number of squares that have vertices belong to the given set.

## Example

### Input:

```
1
7
0 0
0 1
1 0
1 1
1 2
2 1
2 2
```

### Output:

```
3
```

### Output details:

The three squares are:  
 $(0\ 0)$ ,  $(0\ 1)$ ,  $(1\ 1)$ ,  $(1\ 0)$   
 $(1\ 1)$ ,  $(1\ 2)$ ,  $(2\ 2)$ ,  $(2\ 1)$   
 $(0\ 1)$ ,  $(1\ 0)$ ,  $(2\ 1)$ ,  $(1\ 2)$

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » The Lucky Draw

## The Lucky Draw Problem Code: D2

The Chef is planning a buffet for the [DirectiPlex inauguration party](#), and everyone is invited. On their way in, each guest picks up a sheet of paper containing a random number (this number may be repeated). The guests then sit down on a round table with their friends.

The Chef now decides that he would like to play a game. He asks you to pick a random person from your table and have them read their number out loud. Then, moving clockwise around the table, each person will read out their number. The goal is to find that set of numbers which forms an increasing subsequence. All people owning these numbers will be eligible for a lucky draw! One of the software developers is very excited about this prospect, and wants to maximize the number of people who are eligible for the lucky draw. So, he decides to write a program that decides who should read their number first so as to maximize the number of people that are eligible for the lucky draw. Can you beat him to it?

## Input

The first line contains  $t$ , the number of test cases (about 15). Then  $t$  test cases follow. Each test case consists of two lines:

- The first line contains a number  $N$ , the number of guests invited to the party.
- The second line contains  $N$  numbers  $a_1, a_2, \dots, a_n$  separated by spaces, which are the numbers written on the sheets of paper in clockwise order.

## Output

For each test case, print a line containing a single number which is the maximum number of guests that can be eligible for participating the the lucky draw.

## Constraints

- $1 \leq N \leq 10000$
- You may assume that each number number on the sheet of paper;  $a_i$  is **randomly generated**, i.e. can be with equal probability any number from an interval  $[0, U]$ , where  $U$  is some upper bound ( $1 \leq U \leq 10^6$ ).

## Example

Input:

3

2

0 0

3

3 2 1

6

4 8 6 1 5 2

Output:

1

2

4

## Output details

Case 1: No matter where who reads there number first, just one person is eligible for the lucky draw.

Case 2: The person in the last position could read their number first to obtain the sequence 1 3 2. We could then pick 2 sheets with numbers 1 and 2.

Case 3: The person in the fourth position could read their number first to obtain the sequence 1 5 2 4 8 6. We could then pick 4 sheets with numbers 1 2 4 6.

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » The battlefield

## The battlefield Problem Code: D3

The Ministry of Defence (MoD) of Mighty Mouse Kingdom (MMK) needs to arrange defence forces in a battlefield to prevent possible attacks of Catland's army. The battlefield is a square grid of  $N$  rows and  $N$  columns. The cell in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column is denoted by  $(i, j)$ , with  $0 \leq i, j \leq N-1$ .

The MoD wants to put exactly  $N$  cannons in cells of the grid. Each cannon can guard all the cells that are in the same column or same row as it is. To minimize the cost, the MoD decides that each row and each column should be guarded by exactly one cannon.

Of course not all ways of arranging the cannons are equally good strategically. A cannon in cell  $(i, j)$  is said to be in a **safe** position if:

- $1 \leq j \leq N-2$
- The row coordinates of the cannons in the  $j-1^{\text{st}}$  and  $j+1^{\text{st}}$  columns lie on the same side of the row coordinate  $i$  of the cannon  $(i, j)$ . In other words, if we denote  $r_k$  to be the row position of the cannon in the  $k^{\text{th}}$  column, we should have either  $r_{j-1} < i$  and  $r_{j+1} < i$ , or  $r_{j-1} > i$  and  $r_{j+1} > i$ .

The Safety Index (SI) of the battlefield is the number of cannon in safe positions. Military strategists have pointed out that the SI must be at least  $K$  ( $K$  may vary depending on the weather, opponent's forces, etc.) in order to have a good defence of this important battlefield.

Compute the number of ways of arranging the cannons so that the SI is at least  $K$ .

### Input

The first line contains  $t$ , the number of test cases (about 1000). Then  $t$  test cases follow. Each test case is described in a single line containing two numbers  $N$  and  $K$  ( $3 \leq N \leq 2000$ ,  $0 \leq K \leq N-2$ ).

### Output

For each test case, print a single line containing the number of ways of arranging the cannons so that the Safety Index is at least K. Since the result may be very large, you only need to print the remainder of the result when dividing by 30041975.

## Example

**Input:**

```
3
3 1
4 1
8 5
```

**Output:**

```
4
22
13102
```

## Output details

Case 1: There are four ways of arranging the cannons: put the cannon at positions  $\{(1,1), (2,3), (3,2)\}$ ,  $\{(1,2), (2,3), (3,1)\}$ ,  $\{(1,2), (2,1), (3,3)\}$ , or  $\{(1,3), (2,1), (3,1)\}$ . The cannon in the middle column is always in the safe position.

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » Pack the balls in a box!

## Pack the balls in a box! Problem Code: DX

Suppose we have a set of balls and a large cuboid box, with a rectangle as its base. The box has a fixed size at the base, but we can choose its height. We would like to place all the balls within the box, and at the same time try to minimize its height.

### Input

First, 2 integers,  $10 \leq a, b \leq 100$  - the dimensions of the rectangular base of the box. Then, an integer  $1 \leq n \leq 10000$ , representing the number of balls. The following  $n$  values  $1 \leq r_i \leq 5$  are the radii of the respective balls.

### Output

You should write to output  $n$  triples of floating-point numbers, the  $i$ th triple being the  $x, y$ , and  $z$  coordinates of the center of the  $i$ th ball.

If we want to be precise, the coordinates of the points written to output must fulfill the following constraints for the  $i$ -th point:  $x_i - r_i \geq 0$ ,  $y_i - r_i \geq 0$ ,  $z_i - r_i \geq 0$ ,  $x_i + r_i \leq a$ ,  $y_i + r_i \leq b$ . Moreover, for each  $i \neq j$ ,  $(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \geq (r_i + r_j)^2$  (no two balls are allowed to overlap).

## Scoring

The goal is to minimize the height  $h$  of the box, where  $h = \max_i (z_i + r_i)$ . For each data set, your program will be scored by the proportion of the box volume actually used by the balls:

$$\text{score} = 4/3 * \pi * (r_1^3 + \dots + r_n^3) / (a * b * h).$$

The program is run independently for a number of data sets, and the displayed score is the mean of scores obtained for individual data sets.

## Example

### Input:

5 5

2

1.0

2.0

### Output:

4.0 4.0 3.0

2.0 2.0 2.0

### Score:

$$37.6991118 / 100.0 = 0.376991118$$

## Primes in the GCD table Problem Code: D4

Johnny has created a table which encodes the results of some operation -- a function of two arguments. But instead of a boring multiplication table of the sort you learn by heart at prep-school, he has created a GCD (greatest common divisor) table! So he now has a table (of height  $a$  and width  $b$ ), indexed from  $(1,1)$  to  $(a,b)$ , and with the value of field  $(i,j)$  equal to  $\text{gcd}(i,j)$ . He wants to know how many times he has used prime numbers when writing the table.

### Input

First,  $t \leq 10$ , the number of test cases. Each test case consists of two integers,  $1 \leq a,b < 10^7$ .

### Output

For each test case write one number - the number of prime numbers Johnny wrote in that test case.

### Example

#### Input:

2

10 10

100 100

#### Output:

30

2791

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » Alien Clock

## Alien Clock Problem Code: D7

The Chef has been kidnapped by aliens (they want to utilize his excellent cooking skills :P) and has been transported to the alien planet "ZX532". Interestingly, "ZX532" uses a different time scale which has  $Y$  AlienHours on the face of a clock (instead of the 12 hours that we have). They also have ' $X*Y$ ' AlienMinutes (instead of the 60 that we have).  $X$  is defined as the number of minute divisions between each hour (the earth clock has  $X=5$ ).

The Chef has been forced to cook a lot of dishes by the aliens. He is short of ingredients and hence needs to go out to the market to get them. However when he returns, he makes an interesting observation. The hands of AlienClock have exactly swapped position from the time he left (the big hand and the little hand have switched)!

Given the times between which The Chef might have left and the times between which The Chef may have returned, you are supposed to find out the actual time when he left. If there is more than one possibility, print the one that maximizes the amount of time that The Chef is out.

### Details of the Clock

The AlienClock is circular and has exactly 2 hands : the hour hand and the minute hand. The whole circumference of the AlienClock is divided into  $Y$  divisions , each of which is further divided into  $X$  subdivisions [  $X$  and  $Y$  are both positive integers ]. The clock is not discrete but continuous (the hands move continuously instead of moving at certain intervals such as every second, etc.... ).

### Input

First line of input contains  $t$  ( $1 \leq t \leq 100$ ) - the number of test cases. Each test case contains 2 lines each. The first line contains 2 space separated integers ,  $X$  and  $Y$  (same as those specified in the problem statement;  $X \leq 10^9$  and  $Y \leq 10^9$  ).

The next line contains 8 space separated integers ( $a,b,c,d,e,f,g,h$ ;  $0 \leq a \leq c < Y$ ;  $0 \leq e \leq g < Y$ ;  $a \leq e$ ;  $0 \leq b,d,f,h < X*Y$  ): specifying the times between which The Chef might have left ( $a:b \Rightarrow a$  hours and  $b$  minutes TO  $c:d \Rightarrow c$  hours and  $d$  minutes) and the times between which The Chef may have returned (The Chef returns sometime between  $e:f$  and  $g:h$ ).

The interval at which The Chef re-enters the kitchen will never start prior to the interval at which he first leaves the kitchen to purchase the ingredients.

### Output

The output should contain one line for each test case specifying the time when The Chef left the kitchen. This should be in the form ( $h:m \Rightarrow h$  hours and  $m$  minutes).  $h$  will be an integer, and  $m$  should be rounded to 2 decimal places. In case, there is no solution possible, output -1 for that test case.

### Example

Input:

3

5 12

1 0 2 0 4 0 5 0

5 12

3 3 0 4 0 5 0 6 0

5 12

3 0 4 0 5 0 6 0

**Output:**

1:20.56

-1

3:26.43

**Note:** Be careful with precision.

[Home](#) » [Compete](#) » [June 2009 \(Contest IV\)](#) » The Bytelandian Union

## The Bytelandian Union Problem Code: A8

Byteland is a strange country, with many cities, but with a poorly developed road network (in fact, there is exactly one route from each city to any other city, possibly leading through other cities). Until recently, the cities of Byteland were independently governed by proud Mayors, who chose not to integrate too tightly with their neighbours. However, recent opinion polls among Bytelandian computer programmers have shown a number of disturbing trends, including a sudden drop in pizza consumption. Since this had never before happened in Byteland and seemed quite inexplicable, the Mayors sought guidance of the High Council of Wise Men of Byteland. After a long period of deliberation, the Council ruled that the situation was very serious indeed: the economy was in for a long-term depression! Moreover, they claimed that tighter integration was the only way for the Bytelandian cities to survive. Whether they like it or not, the Mayors must now find a way to unite their cities as quickly as possible. However, this is not as easy as it sounds, as there are a number of important constraints which need to be fulfilled:

- Initially, each city is an independent State. The process of integration is divided into steps.
- At each step, due to the limited number of diplomatic envoys available, a State can only be involved in a unification process with *at most one* other state. At each step two States can unite to form a new State, but only if there exists a road directly connecting some two cities of the uniting States.
- The unification process is considered to be complete when all the cities belong to the same, global State.

The Mayors have asked you to arrange a schedule for the diplomatic talks, so that unification can be completed in as few steps as possible. Can you handle this delicate task?<sup>a)</sup>

## Input

The first line contains  $t$ , the number of test cases (less than 1000). The descriptions of  $t$  test cases follow.

Each test case contains the description of the cities of Byteland, given in two lines. The first line contains a single integer  $k$ , representing the number of cities in Byteland ( $2 \leq k \leq 600$ ); we assume that the cities are numbered  $0, \dots, k-1$ . The second line contains exactly  $k-1$  integers, and the  $i$ -th integer having a value of  $p$  represents a road connecting cities having numbers  $i+1$  and  $p$  in Byteland.

## Output

For each test case, output a separate line containing one number, equal to the minimum number of steps required to perform the unification.

## Example

### Input:

```
3
4
0 1 2
8
0 1 2 0 0 3 3
9
0 1 1 1 1 0 2 2
```

### Output:

```
2
4
5
```

a) Some conspiracy theorists claim that this task has in fact nothing to do with unification, and that it was proposed by pizza parlour lobbyists simply to boost their direct revenue at your expense. But don't worry, in any case, you are helping Byteland out of depression!

JULY09

[July 2009 \(Contest V\)](#)01 Jul 2009  
15:00:00

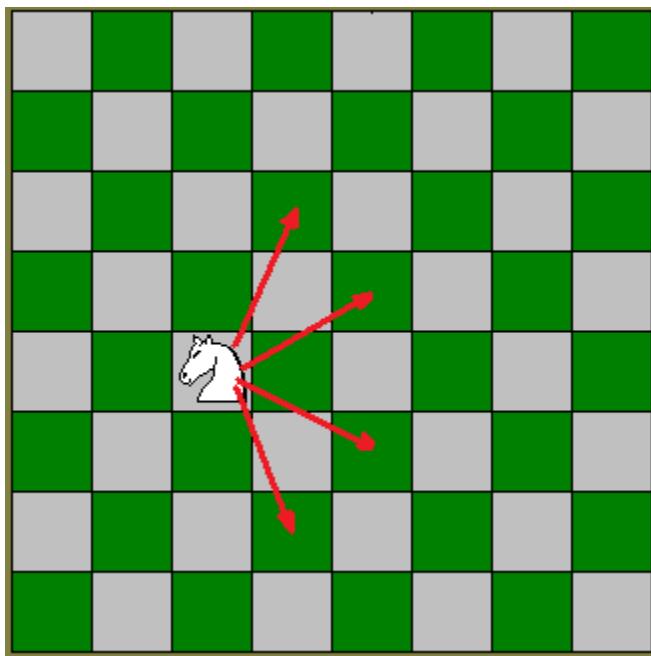
12 days

405

[Home](#) » [Compete](#) » [July 2009 \(Contest V\)](#) » [The White Knight](#)

## The White Knight Problem Code: E1

You are given a chessboard of size  $N \times N$ . There is a white knight and several black pawns located on the board. The knight can move similarly to the normal knight in the game of chess; however it can only move towards the rightmost column of the board (see the below figure).



The mission of the knight is to capture as many black pawns as possible. Its journey ends when it moves to the rightmost column of the board.

Compute the maximum number of black pawns the white knight can capture.

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case has the following form:

- The first line contains  $N$ , the size of the chessboard ( $4 \leq N \leq 1000$ ).

- Then  $N$  lines follow, each line containing  $N$  characters which may be '.', 'K' or 'P', corresponding to the empty cell, the white knight, and the black pawn, respectively. There is exactly one 'K' character in the whole of the board.

## Output

For each test case, print in a single line the maximum number of black pawns that can be captured.

## Example

**Input:**

```
1
5
K.....
..P..
.P...
...P.
....
```

**Output:**

```
2
```

[Home](#) » [Compete](#) » [July 2009 \(Contest V\)](#) » Sums in a cuboid

## Sums in a cuboid Problem Code: CUBESUM

Suppose there is a  $X \times Y \times Z$  3D matrix  $A$  of numbers having coordinates  $(i, j, k)$  where  $0 \leq i < X$ ,  $0 \leq j < Y$ ,  $0 \leq k < Z$ . Now another  $X \times Y \times Z$  matrix  $B$  is defined from  $A$  such that the  $(i, j, k)$  element of  $B$  is the sum of all the the numbers in  $A$  in the cuboid defined by the  $(0, 0, 0)$  and  $(i, j, k)$  elements as the diagonally opposite vertices. In other word  $(i, j, k)$  in  $B$  is the sum of numbers of  $A$  having coordinates  $(a, b, c)$  such that  $0 \leq a \leq i$ ,  $0 \leq b \leq j$ ,  $0 \leq c \leq k$ . The problem is that given  $B$ , you have to find out  $A$ .

## Input

The first line of input will contain the number of test cases ( $\leq 10$ ). That many test cases will follow in subsequent lines. The first line of each test case will contain the numbers X Y Z ( $0 \leq X, Y, Z \leq 100$ ). After that there will be  $X \times Y$  lines each containing Z numbers of B. The first line contains the numbers  $(0, 0, 0), (0, 0, 1) \dots, (0, 0, Z-1)$ . The second line has the numbers  $(0, 1, 0), (0, 1, 1) \dots, (0, 1, Z-1)$  and so on. The  $(Y+1)^{\text{th}}$  line will have the numbers  $(1, 0, 0), (1, 0, 1) \dots, (1, 0, Z-1)$  and so on.

## Output

For each test case print the numbers of A in exactly the same fashion as the input.

## Example

### Input:

2

3 1 1

1

8

22

1 2 3

0 9 13

18 45 51

### Output:

1

7

14

0 9 4

18 18 2

## Lights Off Problem Code: E2

Lights Off is a puzzle game consisting of an  $n \times n$  grid of lights. At the beginning of the game, some of the lights are switched on. When a light is pressed, this light and the four adjacent lights are toggled, i.e., they are switched on if they were off, and switched off otherwise. The purpose of the game is to switch all the lights off.

The following figure illustrates the game:

Johnny has become addicted to playing the Lights Off game on his new iPhone. Yesterday he got stuck at a difficult level. He asked you, the talented programmer, for help. Please write a program to help Johnny solve the Lights Off game, not only for the regular 5x5 board size, but also for much larger dimensions!

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case has the following form:

- The first line contains  $n$ , the size of the board ( $3 \leq n \leq 30$ ).
- Then  $n$  lines follow, each line contains  $n$  characters '0' or '1'. The character in the  $i^{\text{th}}$  line and  $j^{\text{th}}$  column is '0' if the corresponding light is off and '1' if it is on.

## Output

For each test case, in the first line, print  $k$ , the number of times Johny must press a light. Any valid solution in which  $k$  does not exceed 5000 is accepted.

Then  $k$  lines follow, each line containing two numbers  $i$  and  $j$  ( $1 \leq i, j \leq n$ ), describing the position of a light to be pressed. Note that  $(i,j)$  means the light in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column; the rows are numbered 1 to  $n$  from top to bottom, and the columns are numbered 1 to  $n$  from left to right.

## Example

**Input:**

1

3

000

110

010

**Output:**

5

1 1

2 1

2 2

3 2

3 3

**Output details:**

The states of the game after pressing each light are:

000	110	010	010	000	000	000
110	(1,1)	010	(2,1)	100	(2,2)	011
010		010		110		011

[Home](#) » [Compete](#) » [July 2009 \(Contest V\)](#) » Quadratic Equations

## Quadratic Equations Problem Code: E4

**A tutorial for this problem is now available on our blog. Click [here](#) to read it.**

Knowing Johnny's mathematical talent, our teacher has prepared a new interesting problem for him, hoping he will enjoy solving it. The problem description is given below.

"There is a rectangular room of length  $l$  and width  $w$  ( $l$  and  $w$  are integers). The length and width of the room fulfill the relation  $l=Aw+B$ , where  $A$  and  $B$  are given integer constants. The room is divided into square cells of unit dimensions. You have observed that, after adding an integer  $C$  to the number of cells in the room, the number of cells becomes divisible by the prime number  $P$ . Find all the possible values of the width of the room."

### Input

The first line contains  $t$ , the number of test cases (about 10000). Then  $t$  test cases follow. Each test case is given in one line containing 4 integers  $A$ ,  $B$ ,  $C$  and  $P$  ( $2 \leq P < 10^6$ ,  $0 < A < P$ ,  $0 \leq B, C < P$ ).

$P$  is always a prime number.

### Output

For each test case, write the result in one line. The first number  $K$  is the number of solutions. Then  $K$  numbers  $X_1, X_2, \dots, X_K$  follow ( $0 \leq X_i < P$ ) in increasing order, which are the solutions to the corresponding problems.

### Example

Input:

2

1 1 0 2

1 2 2 3

Output:

2 0 1

0

[Home](#) » [Compete](#) » [July 2009 \(Contest V\)](#) » Lights Off, Revisited!

## Lights Off, Revisited! Problem Code: EX

Johnny is, by now, a little bored of the original Lights Off game, and he has proposed his own variant. This time, the task which awaits him is much, much harder...

Imagine we have a large square board, containing an  $n \times n$  grid (matrix) of lightbulbs. Initially, some of them are on, and the rest are off. We can perform some special operations on the whole matrix, and we would like to switch off as many lightbulbs as possible. At each step, the only operation we can perform is defined as follows: we choose one particular lightbulb, one direction (North, East, West, or South), and starting from this lightbulb and going in the specified direction, till the end of matrix, one by one, we toggle the states of the lightbulbs - from on to off, or from off to on.

### Input

First,  $1 \leq n \leq 1000$ , the size of the board. Then  $n$  rows of  $n$  numbers follow, representing the initial state of the lightbulbs, where the  $j$ -th number in the  $i$ -th row represents the state of the lightbulb at coordinates  $(i,j)$ : 0 if it is off, 1 if it is on. All coordinates are 0-based, i.e., the bulb in the top left hand corner has coordinates  $(0,0)$ .

### Output

First, you should output  $k$ , the number of operations you perform. The following  $k$  rows should describe a sequence of operations to be performed on the board, in the order in which they should be applied. Each operation is described by three space separated numbers  $i, j, c$ , where  $0 \leq i, j < n$  are 0-based coordinates of the lightbulb at the starting point ( $i$  stands for the row number,  $j$  stands for the column number), and  $c$  is one character from the set  $\{N, E, W, S\}$ ,  $N$  meaning moving to smaller row coordinates,  $S$  to larger row coordinates,  $W$  to smaller column coordinates, and  $E$  to larger column coordinates.

### Scoring

For each lightbulb than remains on after all the operations, you will receive one penalty point. For each operation performed you will receive one penalty point.

The total score is averaged over 9 data sets.

### Example

Input:

3

1 1 1

1 1 1

1 1 1

Output:

2

0 0 E

0 0 S

Score:

7

[Home](#) » [Compete](#) » [July 2009 \(Contest V\)](#) » Johny the farmer

# Johny the farmer Problem Code: E3

Johny the Farmer has decided to plant potatoes. He quickly realized that he must build a net fence to keep animals from eating his precious plantation. But he only has 3 sticks which can act as vertical poles supporting fence; on the other hand, his supply of net is unlimited. Moreover, since the ground is extremely rocky, the poles can only be positioned in some special places on the field. So, he has decided that the fence must be triangular in shape, with vertices in the distinguished points. Quite naturally, he would also like to maximize the area of the field inside fence -- please help him achieve this goal!

## Task

Write a program which:

- reads from the input the coordinates of the points which can hold the poles,
- calculates the maximal possible area of the potato plantation.

## Input

First, an integer  $t$ , representing the number of test cases.

Each test case starts with an integer  $n$  ( $n \leq 10^6$ ). In the next line there are  $n$  pairs of integers separated by spaces,  $x_i, y_i$ , which are the coordinates for  $i$ -th point where a pole can potentially be located ( $-10000 \leq x_i, y_i \leq 10000$ ).

## Output

For each test case output a single integer - the area of the field surrounded by the fence, **multiplied by 2**.

## Example

### Input

```
1
5
0 0 1 0 0 1 1 2 1 1
```

### Output

```
2
```

## Buggy algorithm Problem Code: E5

Lukas was asked to solve the following problem: 'given a permutation of numbers 1,..,n, calculate the number of inversions in it, i.e., the number of pairs of numbers i, j from 1..n, such that  $i < j$  and i comes after j in the permutation'. After thinking about the problem for some time, he has invented the following algorithm: 'for each number in the permutation, if it is located to the right of its correct position in sorted sequence, add to the output value the distance between this two positions'. Formally, Lukas's answer is given as:  $\text{Sum} (\max \{0, i - \text{pi}[i]\})$ , where  $\text{pi}[i]$  is the i-the element of the permutation .

So, for permutation 4,1,2,3, the algorithm would give the answer  $0+1+1+1 = 3$ , and for the permutation 3,4,5,1,2, the answer  $0+0+0+3+3 = 6$ .

Obviously, this algorithm is wrong (it gives wrong results e.g. for permutation 3,2,1), but... sometimes it gives correct results.

Lucas would like to calculate what are the chances that this algorithm gives the correct result. Specifically, he wants to know how many different permutations of fixed length, which start with a given prefix (several starting numbers), lead to a correct answer to this problem.

### Input

First, an integer  $t < 100$ , representing the number of test cases. Each test case consists of two integers  $n$  and  $m$ , where  $n, 1 \leq n \leq 30$ , is the length of permutation, and  $m, 0 \leq m \leq n$ , is the length of the given prefix; then,  $m$  distinct integers between 1 and  $n$  follow, describing the prefix of the permutation.

### Output

For each test-case, output in how many ways we can augment the prefix to a complete permutation so that the algorithm gives correct results.

### Example

Input:

3

3 0

5 2 1 4

5 1 3

Output:

5

3

9

AUG09	<a href="#">August 2009 (Contest VI)</a>	01 Aug 2009 15:00:00	10 days	235
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [August 2009 \(Contest VI\)](#) » Golf course

## Golf course Problem Code: F1

A company wants to invest into building a golf course. They could choose the land to build the course from a rectangle area of size  $M \times N$ . The rectangle area is divided into  $M \times N$  cells. They measured the elevation of each cell and recorded it into an integer number.

The company decided that the golf course should be a  $K \times K$  square ( $K \leq \min(M, N)$ ). Thus, there are  $(M-K+1)(N-K+1)$  possible  $K \times K$  squares that can be used to build the golf course. For each such square, the company wants to know the highest elevation of a cell inside the square. They also need to know how many cells that have such highest elevations.

Write a program that help the company do the above task.

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains three integers  $M, N, K$  ( $1 \leq M, N \leq 500, 1 \leq K \leq \min(M, N)$  ).
- There are  $M$  lines follow. Each line contains  $N$  nonnegative integers not exceeding 10000. Each integer represents the elevation of a cell.

There is a blank line after each test case.

### Output

For each test case, in the first line print "Case d:" where  $d$  is the number of the test case. Then print  $M-K+1$  lines and in each line print  $N-K+1$  entries. Successive entries should be separated by spaces. The entry in the  $i^{\text{th}}$  line and the  $j^{\text{th}}$  column has the following form:

- The first number is the highest elevation of a cell in the KxK square whose top left corner is (i,j).
- If there is more than one cell in the square that has the highest elevation, print (c), where c is the number of cells inside the square that have the highest elevation.

Print a blank line after each test case.

## Example

**Input:**

2

3 3 2

5 5 5

5 5 5

5 5 5

3 3 2

4 2 1

3 5 7

2 8 8

**Output:**

Case 1:

5 (4) 5 (4)

5 (4) 5 (4)

Case 2:

5 7

8 8 (2)

## Curry Stained Napkin Problem Code: F3

The Chef has a huge square napkin of size  $2^n \times 2^n$ . He folds the napkin  $n-3$  times. Each time he folds its bottom side over its top side, and then its right side over its left side. After each fold, the side length of the napkin is reduced by half. The Chef continues folding until there remains a  $8 \times 8$  sheet, lying flat on a table.

Oh, did I forget to mention that the Chef was cooking a new brown colored curry while folding the napkin. He drops some brown colored gravy onto some cells in the folded  $8 \times 8$  napkin. When he drops the gravy, it soaks through all the cells below it.

Now the Chef unfolds the napkin to its original size. There are now many curry stained brown colored cells in the napkin. They form several separate regions, each of which is connected. Could you help the Chef count how many regions of brown cells are there in the napkin?

Note that two cells are adjacent if they share a common edge (they are not considered adjacent if they only share a corner). Two cells are connected if we can go from one cell to the other via adjacent cells. A region is a maximal set of cells such that every two of its cells are connected.

Please see the example test case for more details.

### Input

The first line contains  $t$ , the number of test cases (about 50). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains  $N$  ( $3 \leq N \leq 10^9$ )
- Then, 8 lines follow. Each line is a string of 8 characters, 0 or 1, where 1 denotes a stained brown cell in the folded napkin.

### Output

For each test case, print a single number that is the number of disconnected brown regions in the unfolded napkin. Since the result may be a very large number, you only need to print its remainder when dividing by 21945.

### Example

**Input:**

3

3

01000010

11000001

00000000

00011000

00011000

00010100

00001000

00000000

4

01000010

11000001

00000000

00011000

00011000

00010100

00001000

00000000

1000000000

11111111

11111111

11111111

11111111

11111111

11111111

11111111

11111111

**Output:**

6

22

1

## Output details

Case 1 and 2: There are 6 brown regions in the 8x8 napkin. If we unfold it once, it has 22 brown regions: 11 regions in the top half and 11 regions in the bottom half (as shown in the figure above).

Case 3: All cells of the napkin are stained, so there is always one brown region, no matter how many times the Chef unfolds the napkin.

[Home](#) » [Compete](#) » [August 2009 \(Contest VI\)](#) » Bytelandian Robots

## Bytelandian Robots Problem Code: F2

The Bytelandian Robotic Department has just successfully made a robot that can be programmed to move. Our robot can perform two type of movements: moving one step to the left or one step to the right.

Our robot moves according to the instruction from its control sequence. A control sequence of the robot is a string consisting of only two type of characters: '+' and '-' symbols, in which '+' means a rightward step and '-' means a leftward step. For example, '+--' is a control sequence that instruct the robot to go one step to the right followed by two steps to the left.

Our robot also has a master control sequence (MCQ), which is a string of  $N$  characters '+' and '-'. **Any valid control sequence of the robot must be a subsequence of the MCQ.** Recall that a subsequence is a sequence obtained by removing some characters of the initial sequence. An empty sequence is also a subsequence of the MCQ.

Our robot is currently placed on a runway of length  $L$ , in which the leftmost point has coordinate 0 and the rightmost point has coordinate  $L$ . Initially our robot stands at the point at coordinate  $A$ . It needs to move to the point at coordinate  $B$ . Of course, it cannot go outside the runway.

How many different control sequences are there that make the robot go from A to B?

## Input

The first line contains  $t$ , the number of test cases (about 50). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains four integers  $N, L, A, B$  ( $1 \leq N \leq 500, 1 \leq L \leq 10^9, 0 \leq A, B \leq L$ ).
- The second line contains the MCQ which is a string of exactly  $N$  characters. The MCQ contains only symbols '+' and '-'.

## Output

For each test case, print a single number that is the number of different control sequences that could make the robot to go from A to B. Since the result may be a very large number, you only need to print the remainder of the result when dividing by 7051954.

## Example

### Input

5

6 6 0 0

+-+-+-

6 6 3 3

+-+-+-

6 1 0 0

+-+-+-

6 6 1 4

+-+-+-

6 6 2 6

+-+-+-

### Output

5

9

4

1

0

## Output details

Case 1: the different control sequences are +-, +--+, +---, +---- and the empty sequence.

Case 2: same as case 1 together with 4 additional sequences: -+, -+++, -++-, +---. These additional sequences no longer make the robot go outside the runway as in case 1.

Case 3: same as case 1 but without the sequence +--- since it will make the robot go outside the runway

Case 4: there is only one possible control sequence: +++

Case 5: there is no way for the robot to go to coordinate 6

[Home](#) » [Compete](#) » [August 2009 \(Contest VI\)](#) » Crystals

## Crystals Problem Code: FX

The Evil Magician of Byteland was performing evil magical experiments, and he has left you with an impressive collection of evil magical crystals which he produced. Honestly, you would be overjoyed to dispose of (or in other words: destroy) these crystals, but destroying a magical crystal is not so easy. To achieve thus, you need to connect three different crystals (red, green and blue) and cast some magic spell to destroy this particular triplet. Each magical crystal has its own mana level. You need a certain amount of your own mana to destroy the triplet, precisely equal to the product of the mana levels of the crystals in the triplet you are destroying. Fortunately, your crystals are all already grouped in triplets, and there are no leftovers (so it is possible to actually destroy all of them); unfortunately, the composition of the triplets is not necessarily optimal from the point of view of mana consumption. However, you are allowed to choose two crystals of the same color, and swap them within triplets (crystals become very unstable and hazardous when not part of a triplet, so you cannot perform any operations more complicated than swapping). But there is a catch (there always is, isn't there?): swapping crystals requires a magic spell -- a spell with a significant mana cost, and what makes matters worse, using this spell (as any other spell) on the crystals being swapped makes them accumulate more mana into their mana level (exactly by 1). Try to minimize the amount of mana you need to use to destroy all the crystals!

## Input

First,  $2 \leq n \leq 10^5$ , the number of crystals of each color. Then,  $0 \leq c \leq 10^4$ , the mana cost of the swapping spell. After that,  $n$  triplets of integers follow, the  $i$ th triplet consisting of  $0 \leq r_i \leq 100$ ,  $0 \leq g_i \leq 100$ ,  $0 \leq b_i \leq 100$ , representing the initial mana levels of successive Red, Green and Blue crystals, respectively.

## Output

First  $0 \leq t \leq 10^6$ , the number of swaps. Then,  $t$  descriptions of the swaps in the order in which they are applied, each of the form:  $1 \leq p \leq 3$ ,  $1 \leq x \leq n$ ,  $1 \leq y \leq n$ , meaning a swap between crystals of the  $x$ th and  $y$ th triplets ( $p=1$  stands for Red, 2 for Green, 3 for Blue).

## Example

Input:

3 10

1 1 1

5 5 5

10 10 10

Output:

2

1 1 3

3 1 2

Score:

$11*1*6 + 5*5*2 + 2*10*10 + 10 + 10 = 336$

[Home](#) » [Compete](#) » [August 2009 \(Contest VI\)](#) » Magic sequence

## Magic sequence Problem Code: F5

Johnny has invented a magic sequence. Each element of the sequence is defined by the same recursive definition - take some linear combination of previous elements (whose coefficients are fixed) and add to them the  $n$ -th powers of some integers. Formally:  $X_n = X_{n-1}$

$a_1 + \dots + X_{n-i}a_i + b_1d_1^n + \dots + b_jd_j^n$ , for some integer constants  $p, q, a_1, \dots, a_p, b_1, \dots, b_q, d_1, \dots, d_q$ . Of course, as the values can quickly grow, he computed them modulo a fixed value:  $10^6$ . He wrote many consecutive values of the sequence, but then he lost most of his work. All he has now, is 10 consecutive values taken from somewhere in the sequence (he doesn't know at what  $n$  they begin), and the recursive rule. And he would like to recover the sequence, or at the very least, to be able to write the next 10 values taken from the sequence.

## Input

First, two integers,  $0 \leq p \leq 4$ ,  $0 \leq q \leq 4$ . Then come the descriptions of the coefficients,  $-100 \leq a_1, \dots, a_p, b_1, \dots, b_q, d_1, \dots, d_q \leq 100$ . Then, the following 10 integers are  $X_n, X_{n+1}, \dots, X_{n+9}$  for some unknown  $n$ .

## Output

Output 10 integers -  $X_{n+10}, X_{n+11}, \dots, X_{n+19}$

## Example

Input:

1 1

1

1

1

11 12 13 14 15 16 17 18 19 20

Output:

21 22 23 24 25 26 27 28 29 30

Explanation:

$$x_n = x_{n-1} + 1$$

Input:

1 1

1

1

2

1 3 7 15 31 63 127 255 511 1023

Output:

2047 4095 8191 16383 32767 65535 131071 262143 524287 48575

Explanation:

$$x_n = x_{n-1} + 2^n = \dots = 2^n + 2^{n-1} + \dots + 1 = 2^{n+1} - 1$$

Input:

2 0

1 1

1 1 2 3 5 8 13 21 34 55

Output:

89 144 233 377 610 987 1597 2584 4181 6765

Explanation:

$x_n = \text{Fib}(n)$  is the n-th Fibonacci number

Input:

2 1

2 -1

2

1

0 1 4 9 16 25 36 49 64 81

Output:

100 121 144 169 196 225 256 289 324 361

Explanation:

$$x_n = n^2$$

[Home](#) » [Compete](#) » [August 2009 \(Contest VI\)](#) » Divide and conquer

## Divide and conquer Problem Code: F6

The Chef has one long loaf of bread of length 1. He wants to cut it into as many little loaves as he can. But he wants to adhere to the following rule: At any moment, the length of the longest loaf which he possesses may not be larger than the length of shortest one, times some constant factor. Every time, he is only allowed to cut exactly one loaf into two shorter ones.

### Input

One floating-point number,  $1 \leq k \leq 1.999$ , meaning the stated constant factor. The number will have at most 3 digits after the decimal point.

### Output

First, you should output one number  $n$ , the maximal achievable number of loaves for the given value of the constant factor. Then, you should output any proof that this number of loaves is in fact achievable:  $n-1$  descriptions of cutting, using the following notation. At each step, you print two numbers: first, the index of the loaf that you want to cut into two parts; second, the length of the newly created loaf (cut off from the original one). It is assumed that the starting loaf has index 0. Each newly created loaf will be given the lowest possible free integer index (so, at the  $i$ th step this will be  $i$ ). Each time, the size of the original loaf will be decreased by the size of the newly created loaf.

### Example

Input:

1.5

Output:

4

0 0.4

0 0.3

1 0.2

SEP09

September 2009  
(Contest VIII)01 Sep 2009  
15:00:00

10 days

219

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » A Coin Game

## A Coin Game Problem Code: G3

Once again, the smart friends Johnny and Mary have invented a new game. This time, the game is played with their coins and a strip of paper divided into cells. The cells are numbered from the left, starting from 1.

Before the game starts, Johnny and Mary put some coins at some random cells on the strip, each coin in a cell. They alternatively take their turn to play the game. At each step, a player must take a coin and move it to a cell to the left of it. There is at most one coin in a cell at any time of the game. Of course, a coin cannot jump over another coin.

The rule is simple: whoever cannot move loses the game. Mary takes her turn first.

Could you tell them who will win the game, provided that the winner will play the perfect strategy? If Mary could win the game, show her a winning move. If there are several winning moves, show her the winning move that uses the leftmost possible coin. If there are still several moves, show her the move that moves the coin as far as possible to the left.

### Input

The first line contains  $t$ , the number of test cases (about 100). Each test case has the following form:

- The first line contains an integer  $N$  ( $1 \leq N \leq 10000$ ) describing the number of coins on the strip.
- The second line contains  $N$  integers  $p_1, p_2, \dots, p_n$  ( $1 \leq p_1 < p_2 < \dots < p_n \leq 10^9$ ) representing the initial position of the coins.

Each test case is separated by a blank line.

### Output

For each test case, print either the string "Mary wins" or "Johnny wins" depending on who will win the game. If Mary could win the game in the next line, print the string "Move  $a$  to  $b$ " describing the desired winning move, where  $a$  is replaced by the position of the coin to move and  $b$  is replaced by the new position of that coin.

Remember to print a blank line after the output for each test case.

### Example

**Input:**

3

2

2 3

3

1 3 5

4

2 4 6 9

**Output:**

Johnny wins

Mary wins

Move 5 to 4

Mary wins

Move 2 to 1

**Output details**

Case 1: Mary's only possible move is to move the coin at position 2 to position 1. Johnny then moves 3 to 2 and wins the game.

Case 2: After Mary moves 5 to 4, the only possible move for Johnny is to move 3 to 2. Mary then finishes the game by moving 4 to 3.

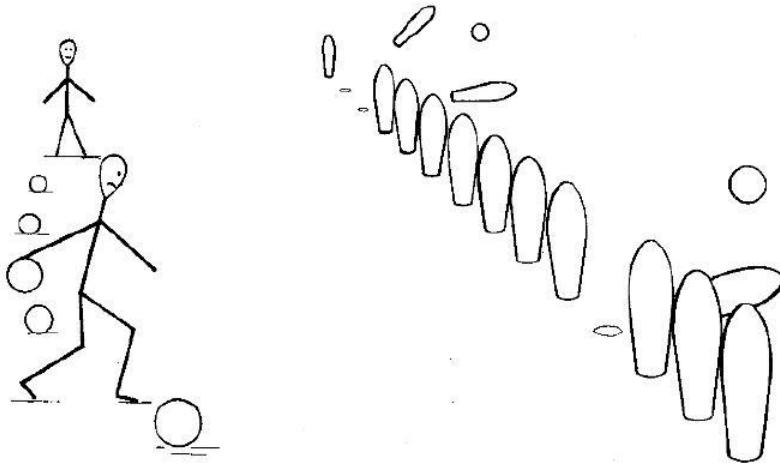
Case 3: (see the above figure) Among the winning moves, moving 2 to 1 uses the leftmost possible coin.

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » A Bowling Game

## A Bowling Game Problem Code: G1

Mary and Johnny are playing a game with bowling pins. There is a row of  $N$  bowling pins. Mary and Johnny alternatively take their turn to bowl. They are such skillful bowling players that at each player's turn, any of them can take out any row of adjacent bowling pins, provided that the number of pins taken out is no more than  $K$ . It is important to notice that all the pins taken out must be adjacent. It is impossible to knock down pins separated by a greater distance. Whoever knocks out the last bowling pin wins the game.

You have just come down to watch Mary and Johnny playing the game. Some pins are already removed and now it is Mary's turn. With a good knowledge of mathematical games, you know that it is possible to compute in advance whether Mary or Johnny will win the game, provided that the winner always plays the perfect strategy. So why not impress your two friends by telling them who will win? If Mary could win the game, show her a winning move. Since Mary always wants to win the game as fast as possible, if there are more than one possible winning moves, show her the move that has the most number of pins taken out. If there are still several solutions, show her the move in which the pins taken out are as near to the left as possible.



### Input

The first line contains  $t$ , the number of test cases (about 15). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains two numbers  $N$  and  $K$  ( $1 \leq N \leq 2000$ ,  $1 \leq K \leq 20$ )
- The second lines contains a string of  $N$  characters 0 or 1 (without spaces) representing the row of  $N$  bowling pins. A character 0 means the bowling pin at

the corresponding position is already taken out whereas a character 1 means the bowling pin is still in its position.  
Each test case is separated by a blank line.

## Output

For each of the test case, print either the string "Mary wins" or "Johnny wins" depending on who will win the game. If Mary could win the game, in the next line print N characters 0 or 1 representing the row of N bowling pins after Mary's move. Use the same format as that of the input data.

Remember to print a blank line after the output for each test case.

## Example

**Input:**

3

3 2

111

8 3

10110111

8 4

11111111

**Output:**

Mary wins

101

Johnny wins

Mary wins

11000011

## Output details

Case 1: Mary knocks down the middle pin. Regardless of which pin Johnny takes out in the next move, Mary always takes out the remaining pin and wins the game.

Case 2: Mary will lose the game regardless of which move she plays. For example if she takes out the rightmost pin, it is easy to verify that Johnny will always win if he then takes out the leftmost pin.

Case 3: There is more than one move that allows Mary to win the game. The four middle pins make the most number of pins that Mary can take out.

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » Smart Frog

## Smart Frog Problem Code: G2

Bibi the Smart Frog is playing a jumping game on an  $m \times n$  rectangular board. There is a number written in each cell of the board (Bibi can read these numbers since he is very smart!)

Bibi starts the game by picking any cell on the board and stays there. At each step, Bibi will jump to another cell. He can either:

- Jump to the **right** to a cell in the **same row**, provided that the number written in that cell is **not smaller than** the number written in the current cell.
- Jump **downwards** to a cell in the **same column**, provided that the number written in that cell is **not greater than** the number written in the current cell.

To win the game, Bibi needs to jump through as many cells as possible. But no, he is not so extraordinarily smart that he could compute the optimal way to play the game. After all, he is only a frog, you know! Write a program to help Bibi compute the maximum number of cells that he can jump into.

## Input

The first line contains  $t$ , the number of test cases (about 10). Each test case has the following form:

- The first line contains two integers  $m, n$  ( $1 \leq m, n \leq 500$ ).

- Then,  $m$  lines follow, each line containing  $n$  numbers representing the rectangular board. It is guaranteed that no number exceeds  $10^6$ .

Each test case is separated by a blank line.

## Output

For each test case, print a single number that is the maximum number of cells that Bibi can jump into.

## Example

**Input:**

2

2 3

2 1 1

2 1 1

4 4

6 2 5 2

4 5 3 8

9 7 8 9

9 9 9 5

**Output:**

3

5

**Output details**

A possible set of cells that Bibi could jump into is marked in bold:

2 **1** 1

2 1 **1**

6 **2** 5 2

4 5 **3** 8

9 7 8 9

9 9 9 5

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » [Se7en](#)

## Se7en Problem Code: F4

Tomek and his numerous friends are standing in a circle. They are all numbered with consecutive identifiers, from 1 to 1337 in the clockwise direction. Starting from person 1, who says "1", successive people read out successive positive integers. The starting direction is clockwise, and there is rule, that whenever integer is divisible by 7 or contains digit 7, the direction is reversed.

So, the identifiers of the persons who read-out successive numbers, are: 1,2,3,4,5,6,7 (person 7 has just read "7" and reversed the direction),6,5,4,3,2,1,1337 (person 1337 has just read "14" and reversed the direction),1,2,3 (person 3 has just read "17" and reversed the direction),2,1,1337,1336, and so on. Tomek has his favorite number, and he wants to calculate where in the circle he should stand to read that number out loud. Even though he is skilled programmer, he is a bit little lazy and would like you to help him out.

### Input

First,  $1 \leq t \leq 1000$ , the number of test cases. Each test case is in a separate line, and contains a positive integer smaller than  $10^{100}$ , representing Tomek's favorite number.

### Output

For each testcase, output one integer, the identifier which Tomek should choose in the circle to read his favorite integer loud.

### Example

Input:

3

10

100

1000

Output:

4

2

1311

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » Chemical reactions

## Chemical reactions Problem Code: G5

Johnny is preparing a chemical experiment for school. The experiment involves a certain number of different liquid chemical substances which have been mixed together and are currently reacting with each other. There are  $n$  different types of liquid substances. However, each substance appears in  $k$  different variants called isomers, and Johnny cannot distinguish between the different isomers...

We can assume that time is discrete and measured in minutes. During each minute, each possible reaction is described as follows: a b ratio type, meaning that ratio of liquid a (i.e., part of total amount, no matter which isomer) transforms into liquid b. If the reaction type is normal, then isomer  $i$  of liquid a transforms into isomer  $i$  of liquid b. If reaction type is special, then isomer  $i$  transforms into isomer  $i+1$ .

At time 0, each substance is in its initial isomer form, having number 1. For the special threshold value  $k$ , once isomer  $k$  of any substance passes through a special reaction, it transforms into solid state matter and precipitates from the mixture, no longer taking part in reactions (so, for all practical purposes, we can say it disappears). Moreover, during each minute, a small part of each liquid disappears (transforms into gaseous state and evaporates). From Johnny's point of view, the evaporation happens just before the reactions in any given minute.

Starting from time 0, exactly once a minute, Johnny takes note of the amount of each liquid in the mixture. He does this for a very, very long time, and then he sums the obtained amounts for each liquid (i.e., for each liquid, he adds the amount there was at time 0, at time 1, at time 2, etc.). The resulting  $n$  numbers are the result of the experiment. However, Johnny would like to know the results of experiment beforehand, so he asks you for a little

help. Compute the results of his experiment, knowing that initially he just mixes the same amount (1 unit) of isomer 1 of each substance.

## Input

First,  $1 \leq n \leq 100$ , then  $1 \leq k \leq 1000$ . Then,  $n$  floating point numbers follow, describing the proportion of liquid of each type that evaporates in each minute. Then, one integer  $m < 10000$ , the number of reactions, followed by  $m$  reaction definitions. Each definition consists of integers  $0 \leq a < n$ ,  $0 \leq b < n$ , floating point  $0 < r \leq 1$ , and integer  $s=0$  or  $1$ , meaning that part  $r$  of the total amount of liquid  $a$  transforms into liquid  $b$  every minute (in the normal way if  $s=0$ , and in a special reaction, otherwise). No two pairs  $(a,b)$  are the same between definitions, i.e., always  $a \neq b$ .

## Output

Exactly  $n$  numbers, representing the measured results of the experiment for each substance. Precision up to at least 6 digits after the decimal point is required.

## Example

Input:

2

10

0.1 0.1

2

0 1 0.5 1

1 0 0.5 1

Output:

8.65569367258.6556936725

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » Mosaic

## Mosaic Problem Code: GX

Imagine a large mosaic made of glass. It has the form of a large square, of size  $n \times n$ , made up from unit squares of different types of glass. Different types of glass have different physical parameters, such as the speed of light in the material.

We have a source of light in corner  $(0,0)$  and a light detector in corner  $(n,n)$ . We would like to estimate as accurately as possible the path taken by a ray of light which is directed from the source and reaches the detector, knowing that of all the possible routes, the one chosen by light will be the one with the shortest travel time.

Each type of glass has 3 defining parameters, denoted  $o_1, o_2$ , and  $\text{ang}$ . In order to calculate the travel time of light along some straight line segment in one specific type of glass, we calculate the lengths  $l_1$  and  $l_2$  of the projections of this segment onto two perpendicular lines: onto the axis forming angle  $\text{ang}$  with the horizontal direction (X axis), and onto the axis perpendicular to it. Then, the travel time of light along the given segment is given as  $\sqrt{(l_1 \cdot o_1)^2 + (l_2 \cdot o_2)^2}$ .

### Input

First,  $1500 \leq n \leq 2000$ , the size of mosaic. Then,  $n$  rows,  $n$  numbers in each, describing the  $o_1$  values for each square, with the  $x$ -th number in the  $y$ -th line corresponding to the square with its top left-hand corner at  $(x,y)$ . Then, in the same way, come  $n$  rows of  $n$  numbers defining values of parameter  $o_2$ , and  $n$  rows of  $n$  numbers defining values of parameter  $\text{ang}$ . The angle is given in radians, and for each square we have,  $0.1 \leq o_1, o_2 \leq 100$ ,  $0 \leq \text{ang} \leq 2\pi$

### Output

You should describe the route from  $(0,0)$  to  $(n,n)$  by outputting  $k$  ( $k \leq 10^6$ ) and then the coordinates of  $k$  midpoints of the route (the points  $(0,0)$  and  $(n,n)$  should not be output). The route must follow the following rules: each segment between successive points of the route starts on a side of some unit square and ends on a side of this square, and the segment between them lies entirely within this unit square. Successive midpoints must be different from each other, route segments may not run along edges of unit squares, and are not allowed to lead outside of the mosaic.

### Scoring

Your score is equal to the travel time of light along the fictional route defined by your solution (summed over all data sets). The goal is to minimize the obtained score.

### Example

Input:

1.0 2.0

3.0 1.0

1.0 1.0

1.0 1.0

0 0.785398

0.785398 0

Output:

2

0.5 1

1 1.5

Score:

3.650282

[Home](#) » [Compete](#) » [September 2009 \(Contest VIII\)](#) » Alien language

## Alien language Problem Code: G4

Aliens from the planet of Zrxllrlv have an extremely well-developed language. Recently they have introduced a special alphabet which consists of only 2 symbols. Now, they would like to develop a way to write down all the words they have in their language using this alphabet. They want to be able to decode sequences of words without breaks between them, so they would like to retain the following property: no word is the proper prefix of any other word. Knowing how many words they have, that each word occurs equally often in every-day use, and knowing the effort required to write down each of the two symbols of the alphabet (the complexity of the first symbol and the second symbol need not to be equal!) help them to develop an encoding which minimizes the mean effort required to write down a word of the language.

### Input

First,  $1 \leq t \leq 10000$ , the number of test cases. Each test case contains:  $1 \leq a \leq 10^9$ ,  $1 \leq b \leq 10^9$ ,  $1 \leq n \leq 10^{12}$ , meaning: the effort required for writing down the first symbol, the effort required for writing down the second symbol, and the number of words of the language, respectively.

## Output

For each testcase, output the sum of lengths of encodings of all words in the optimal encoding.

## Example

Input:

2

2 1 3

1 1 16

Output:

7

64

## Explanation

The optimal encoding for the first testcase is:

'0','10','11'.

The optimal encoding for the second testcase is:

'0000','0001','0010','0011','0100','0101','0110','0111','1000','1001','1010','1011','1100','1101','1110','1111'.

OCT09	<a href="#">October 2009 (Contest IX)</a>	01 Oct 2009 15:00:00	10 days	368
-------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [October 2009 \(Contest IX\)](#) » A puzzle game

## A puzzle game Problem Code: H1

Johnny has some difficulty memorizing the small prime numbers. So, his computer science teacher has asked him to play with the following puzzle game frequently.

The puzzle is a 3x3 board consisting of numbers from 1 to 9. The objective of the puzzle is to swap the tiles until the following final state is reached:

1 2 3

4 5 6

7 8 9

At each step, Johnny may swap two adjacent tiles **if their sum is a prime number**. Two tiles are considered adjacent if they have a common edge.

Help Johnny to find the shortest number of steps needed to reach the goal state.

## Input

The first line contains  $t$ , the number of test cases (about 50). Then  $t$  test cases follow. Each test case consists of a 3x3 table describing a puzzle which Johnny would like to solve.

The input data for successive test cases is separated by a blank line.

## Output

For each test case print a single line containing the shortest number of steps needed to solve the corresponding puzzle. If there is no way to reach the final state, print the number -1.

## Example

Input:

2

7 3 2

4 1 5

6 8 9

9 8 5

2 4 1

3 7 6

Output:

6

-1

## Output details

The possible 6 steps in the first test case are described in the following figure:

<table border="1"><tr><td>7</td><td>3</td><td>2</td></tr><tr><td>4</td><td>1</td><td>5</td></tr><tr><td>6</td><td>8</td><td>9</td></tr></table>	7	3	2	4	1	5	6	8	9	<b>Step 1</b>	<table border="1"><tr><td>4</td><td>3</td><td>2</td></tr><tr><td>7</td><td>1</td><td>5</td></tr><tr><td>6</td><td>8</td><td>9</td></tr></table>	4	3	2	7	1	5	6	8	9	<b>Step 2</b>	<table border="1"><tr><td>4</td><td>3</td><td>2</td></tr><tr><td>6</td><td>1</td><td>5</td></tr><tr><td>7</td><td>8</td><td>9</td></tr></table>	4	3	2	6	1	5	7	8	9	<b>Step 3</b>	<table border="1"><tr><td>4</td><td>3</td><td>2</td></tr><tr><td>1</td><td>6</td><td>5</td></tr><tr><td>7</td><td>8</td><td>9</td></tr></table>	4	3	2	1	6	5	7	8	9
7	3	2																																								
4	1	5																																								
6	8	9																																								
4	3	2																																								
7	1	5																																								
6	8	9																																								
4	3	2																																								
6	1	5																																								
7	8	9																																								
4	3	2																																								
1	6	5																																								
7	8	9																																								
<b>Step 4</b>	<table border="1"><tr><td>1</td><td>3</td><td>2</td></tr><tr><td>4</td><td>6</td><td>5</td></tr><tr><td>7</td><td>8</td><td>9</td></tr></table>	1	3	2	4	6	5	7	8	9	<b>Step 5</b>	<table border="1"><tr><td>1</td><td>2</td><td>3</td></tr><tr><td>4</td><td>6</td><td>5</td></tr><tr><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	6	5	7	8	9	<b>Step 6</b>	<table border="1"><tr><td>1</td><td>2</td><td>3</td></tr><tr><td>4</td><td>5</td><td>6</td></tr><tr><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	5	6	7	8	9										
1	3	2																																								
4	6	5																																								
7	8	9																																								
1	2	3																																								
4	6	5																																								
7	8	9																																								
1	2	3																																								
4	5	6																																								
7	8	9																																								

[Home](#) » [Compete](#) » [October 2009 \(Contest IX\)](#) » Forces in the crystal

## Forces in the crystal Problem Code: HX

Suppose that we have a crystalline triangular grid of atoms. Atoms are arranged on a regular grid, and each atom has six neighbors (unless it lies on the boundary of crystal). Each atom has an electric charge of  $q$ , and can be polarized in only one of two directions: up or down (the laws of physics in Byteland are somewhat surprising!). If two atoms are neighbors and share the same polarization, then a destructive force against the crystal occurs, whose value is equal to  $q_1 \cdot q_2$ . We can choose the polarization for each atom, and we would like to minimize total force working against the crystal, i.e., the sum of values of all the individual forces.

## Input

First,  $1000 \leq n \leq 2000$ , the size of input. Then  $n$  lines with  $n$  numbers in each follow. In the  $x$ -th line,  $1 \leq x \leq n$ , the  $y$ -th number,  $1 \leq y \leq n$ , is the charge of the atom with coordinates in the crystal equal to  $(x, y)$  if  $x$  is odd,  $(x, y+1/2)$  otherwise. For any atom at coordinates  $(x, y)$ , the coordinates of the neighbors are assumed to be the following (as long as they appear in the crystal):  $(x-1, y-1/2), (x-1, y+1/2), (x, y-1), (x, y+1), (x+1, y-1/2), (x+1, y+1/2)$ . Each charge  $q$  is in the range  $0.1 \leq q \leq 1$ .

## Output

For each atom given at input, output 0 or 1 depending on whether the polarization of the atom should be directed up or down.

## Score

The score of your program is equal to the value of the force acting on the crystal. The program is tested multiple times for different data sets, and the results are averaged.

## Example

Input:

3

1 2 3

4 5 6

7 8 9

Output:

0 0 0

0 0 0

0 0 1

Score:

$269 = (1*2+1*4+2*3+2*5+2*4+3*6+3*5+4*5+4*7+4*8+5*6+5*8+7*8)$

[Home](#) » [Compete](#) » [October 2009 \(Contest IX\)](#) » Just a simple sum

## Just a simple sum Problem Code: H4

Given  $n$  and  $m$ , calculate  $1^1 + 2^2 + 3^3 + \dots + n^n$  modulo  $m$ .

## Input

The first line contains  $1 \leq t \leq 10$ , the number of test cases. Then the test case definitions follow. Each test case is of the form:  $1 \leq n \leq 10^{18}$ ,  $1 \leq m \leq 200000$

## Output

For each test case simply write a single line containing the answer to the problem.

## Example

Input:

6

1 100000

2 100000

3 100000

4 100000

5 100000

6 100000

Output:

1

5

32

288

3413

50069

# Paragraph Formatting Problem Code: H5

Johnny is developing some word processing software. Right now, he has to deal with the problem of formatting a paragraph. Johnny has formulated the problem as follows.

There are  $N$  words in a paragraph, in which the  $i^{\text{th}}$  word has  $a_i$  characters. Each line of the paragraph can hold at most  $M$  characters. For simplicity, we assume that every two consecutive words in a line are separated by **exactly one whitespace** and we disregard punctuation marks.

The text editor always uses a simple greedy algorithm to break the paragraph into lines. The algorithm puts as many words in a line as possible, then moving on to the next line to do the same until there are no more words left to be placed.

Johnny needs to write a program that, given the description of the words in a paragraph, is able to process the following two operations:

- Return the line number of a specified word
- Replace one word with another one

Since the number of words in a paragraph can be huge, Johnny needs to find an efficient algorithm to process the above queries. Please help him!

## Input

The first line contains  $t$ , the number of test cases (about 50). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains a number  $M$ , the maximum number of characters that can be put in a line.
- The second line contains a number  $N$ , the number of words in the given paragraph.
- The third line contains  $N$  numbers  $a_1, a_2, \dots, a_N$ , the lengths of the words in the paragraph.
- The next lines contain the description of the queries, one query in a line. Each query can be of one of the following 3 types:
  - $l\ i$  - Ask for the line number of the  $i^{\text{th}}$  word.
  - $C\ i\ l$  - Replace the  $i^{\text{th}}$  word by a new word of length  $l$  ( $1 \leq l \leq M$ ).
  - $E$  - Signal the end of the description of the queries.

Note that the line numbers and the word indexes are counted from 1.

## Output

For each test case, the first line of output should contain the string "Case #T:" where  $T$  should be replaced by the corresponding test case number.

For every ' $l$ ' query in the test case, print the correct line number of the word being queried.

Print a blank line after each test case.

## Constraints

- $1 \leq N \leq 50000$
- $1 \leq M \leq 100000$
- $1 \leq a_i \leq M$
- The number of lines in each paragraph never exceeds 200.
- The number of queries in each test case does not exceed 10000.

## Example

Input:

2

4

2

1 2

I 2

C 2 3

I 2

E

6

5

1 5 4 5 6

I 2

I 4

I 5

C 4 1

I 4

I 5

C 2 1

C 3 2

C 5 4

I 5

I 3

E

**Output:**

Case #1:

1

2

Case #2:

2

4

5

3

4

2

1

**Discussion of the example**

In the following description of the second exemplary test case, we use the digit  $i$  to denote a character of the  $i^{\text{th}}$  word. Note that each line can hold at most 6 characters. Initially, the paragraph has the following form:

1

22222

3333

44444

555555

After replacing the 4<sup>th</sup> word with a one-character word, the paragraph becomes:

1

22222

3333 4

555555

After the last 3 replacements, the paragraph becomes:

1 2 33

4 5555

[Home](#) » [Compete](#) » [October 2009 \(Contest IX\)](#) » [Kayaks](#)

## Kayaks Problem Code: H2

Suppose we have an even number of people going on an excursion, and we would like to fit them into two-seated kayaks. All kayaks are identical, weighing 20kg each. Each person is described by two parameters, their strength and weight. The speed of a kayak can be calculated as the sum of strengths of both persons sitting in it, divided by the total weight of the loaded kayak (i.e., the weight of the kayak plus the weight of both persons). We would like to choose the allocation of people to kayaks so as to maximize the speed at which the whole group can travel, assuming that the group travels at the speed of the slowest kayak in it.

### Input

First,  $2 \leq n \leq 10^5$ , the number of people (n will always be even). Then, n pairs of integers follow, each pair describing one person: first,  $50 \leq w \leq 100$ , the weight of the person (in kilograms), then  $50 \leq p \leq 100$ , the strength of the person.

## Output

Output one number: the maximum speed of the group which can be achieved by optimally choosing places for each person. The answer should be accurate up to 6 digits after the decimal point.

## Example

Input:

4

50 50

50 60

70 100

100 60

Output:

0.842105

[Home](#) » [Compete](#) » [October 2009 \(Contest IX\)](#) » Congruent triangles

## Congruent triangles Problem Code: H3

After learning about congruent triangles in his recent mathematics lesson, Johnny has become very excited. He has even invented an interesting counting problem involving congruent triangles!

Recall that two triangles are congruent if their corresponding sides are equal in length and their corresponding angles are equal in size. A lattice triangle is a triangle such that the coordinates of all its vertices are integers. Johnny's problem can now be described as follows:

You are given an integer  $M$  and a lattice triangle  $ABC$  all of whose vertices (A, B, and C) are inside rectangle  $R_M$ .  $R_M$  is the rectangle having  $(0,0)$  as its bottom-left corner and  $(M,M)$  as its top-right corner. In other words,  $0 \leq x_A, y_A, x_B, y_B, x_C, y_C \leq M$ . The problem is to count the

number of lattice triangles congruent to ABC also having all their vertices inside the rectangle  $R_M$ .

Could you help Johnny write a program to solve this problem?

## Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains two numbers  $M$  and  $K$  ( $1 \leq M \leq 1000$ ,  $1 \leq K \leq 1000$ ).  $K$  is the number of given triangles.
- Each line in the next  $K$  lines contains 6 integers  $x_A, y_A, x_B, y_B, x_C, y_C$  ( $0 \leq x_A, y_A, x_B, y_B, x_C, y_C \leq M$ ) describing a triangle ABC. It is guaranteed that ABC is always a triangle (i.e., non-degenerate).

The input for successive test cases is separated by a blank line.

## Output

For each test case, output a line containing the string "Case #T:" where  $T$  should be replaced by the corresponding test case number.

Then,  $K$  lines should follow, each line containing the number of triangles having vertices inside the rectangle  $R_M$ , congruent to the corresponding triangle given in the input.

Print a blank line after each test case.

## Example

Input:

2

2 2

0 0 0 2 2 0

0 0 1 1 2 0

3 2

0 0 0 2 2 0

0 0 1 1 2 0

**Output:**

Case #1:

4

8

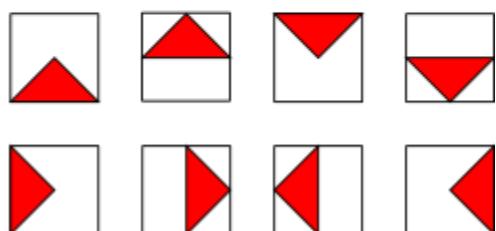
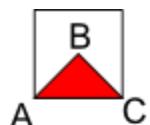
Case #2:

16

24

## Output details

The 8 triangles congruent to the second triangle (in the first test case) are presented in the following figure:



NOV09	<a href="#">November 2009 (Contest X)</a>	01 Nov 2009 15:00:00	10 days	338
-------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [November 2009 \(Contest X\)](#) » The Best Box

## The Best Box Problem Code: J7

Johnny needs to make a rectangular box for his physics class project. He has bought  $P$  cm of wire and  $S$   $\text{cm}^2$  of special paper. He would like to use all the wire (for the 12 edges) and paper (for the 6 sides) to make the box.

What is the largest volume of the box that Johnny can make?

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case contains two integers  $P$  and  $S$  in a line ( $1 \leq P \leq 40000$ ,  $1 \leq S \leq 20000$ ). You may assume that there always exists an optimal solution for the given input cases.

### Output

For each test case, print a real number that is the largest volume of the box that Johnny can make, rounded to two decimal places.

### Example

#### Input:

2

20 14

20 16

#### Output:

3.00

4.15

#### Output details

First case: the dimensions of the largest box may be 3, 1 and 1.

Second case: the dimensions of the largest box may be  $7/3$ ,  $4/3$  and  $4/3$ .

## Help the DJ Problem Code: JX

DJ N3xt0r could hardly sleep last night. He thought he was doing his job so well, but it seems like the dancers at the disco are looking for a whole different kind of music these days and the place has become nearly empty. Being the best DJ ever, he would not let that happen again. So he pulled some strings and gathered valuable information regarding the musical preferences of the dancers expected to attend the disco tonight. By 8.00PM, DJ N3xt0r has a file with the following info:

- The number of dancers (D) expected to attend tonight.
- The number of songs (S) on his laptop media library.
- The boredom factor (B) of the dancers: the number of songs disliked by a dancer that will play before he gets bored and leaves the disco.
- The musical preferences of the dancers represented by a D-row S-column matrix (M). If  $M_{ij} = 1$  then dancer  $i$  likes song  $j$ . If  $M_{ij} = 0$  he does not like the song.

He wants to build a playlist such that the number of dancers present when the party ends is maximized. Desperate, DJ N3xt0r seeks for your help. He will make you a VIP member of the disco if you write a program that performs the task. Notice that, as the playlist will be projected on a screen, dancers who don't see any song they like will leave immediately.

### Input

Input starts with two integers ( $1 \leq D, S \leq 500$ ) separated by a single space. The next line contains the boredom factors of the dancers ( $1 \leq B_1, B_2, \dots, B_D \leq S$ ) separated by single spaces. The next  $D$  lines describe the  $M$  matrix. Each line represents the musical preferences of a dancer, and will consist of  $S$  binary integers (0, 1) separated by single spaces.

### Output

Print the number of songs selected ( $P$ ) for the playlist on a single line. Then print  $P$  integers, each one on a single line, indicating the songs selected, in any order. Notice that selecting a song more than once is not something a self-respecting DJ would ever do, so this will lead to Wrong Answer. The song numbers start from 0, i.e., print 0 when selecting the first song and  $S-1$  when selecting the last.

### Scoring

Points scored for a single test will equal the number of dancers present when the party ends according to your playlist. The total score of the program is the sum of points scored in all tests.

### Example

#### Input:

4 5

```

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

```

**Output:**

```

3
0
4
1

```

**Scoring:**

```

2

```

[Home](#) » [Compete](#) » [November 2009 \(Contest X\)](#) » The LCS Problem Revisited

## The LCS Problem Revisited Problem Code: J2

The Longest Common Subsequence (LCS) problem is a well known problem in Computer Science. Every computer science students in Byteland knows this problem. Johnny does, too.

Recall that a subsequence of a string  $S$  is obtained by deleting some characters from  $S$ . Given two strings  $S$  and  $T$ , the LCS problem is the find the longest sequence that is a subsequence of both  $S$  and  $T$ .

Johnny has the habit of deriving harder problems from a familiar problem. This time, based on the LCS problem, he has thought up the following problem:

*Given two strings  $S$  and  $T$ , how many distinct LCS of  $S$  and  $T$  are there?*

Write a program to help Johnny solve this problem. Since the result may be very large, you only need to print the remainder of the result when dividing by 23102009.

### Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case consists of two lines, the first line is the string  $S$  and the second line is the string  $T$ . You may assume that the strings consists of only lowercase characters and the length of the each string is at most 1000 characters.

Successive test cases at input are separated by a single blank line.

### Output

For each test case, print a single line containing two numbers which are the length of the LCS and the remainder of the number of distinct LCS of  $S$  and  $T$  when dividing by 23102009.

## Example

### Input

2

acbd

acbd

vnnv

vn

### Output

4 1

2 1

### Output details

The only LCS in the first case is "acbd" and in the second case is "vn".

[Home](#) » [Compete](#) » [November 2009 \(Contest X\)](#) » SudokuX

## SudokuX Problem Code: J1

SudokuX is a variation of the popular game Sudoku. Similar to standard Sudoku, in SudokuX, we have to enter the numbers 1 - 9 once in each row, column and 3 x 3 square within the 9x9 puzzle grid. In addition to standard Sudoku, the numbers must only occur **once in each of the two diagonals**.

	1		4	6				
	2		5			3	1	
4		3						
					3	9		
6			7				5	
	7	3						
				4				2
8	6			2		5		
	7	8			1			

Johnny is very interested in playing SudokuX. He believes that this game is so challenging that even the best computer programs could not solve it efficiently.

Let's show Johnny the power of computers by writing a program to solve even the hardest SudokuX puzzles within seconds!

## Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow.

Each test case consists of 9 lines. Each line contains 9 characters '.' (blank grid), or '1'..'9' representing a SudokuX puzzle.

Each test case is separated by a blank line.

You are guaranteed that each given SudokuX puzzle has a unique solution.

## Output

For each puzzle, output the solution in the same format as the input. Print a blank line after each test case.

## Example

### Input

2

.8....2..

.1....5..

..34..7..

..9.5....

.2...46..

3.....

9...2....

.....

.....4.7

....41...

...6....5

.....7.9.

....1.3..

.5.....1

.2.....

..18...76

.7.....2

.....3

## **Output**

486715293

712938546

593462718

679251384

128394675

354876129

945627831

867143952

231589467

293541768

748692135

615387294

864715329

357269481

129438657

531824976

476953812

982176543

[Home](#) » [Compete](#) » [November 2009 \(Contest X\)](#) » Magic Strings

## Magic Strings Problem Code: J5

Magic strings, invented by the Bytelandians, are strings that contain immense magical power within themselves. Magic strings could bring luck and happiness to the Bytelandian citizens. Formally, a string  $S$  of length  $n$  is a magic string if it satisfies the following conditions:

- All letters of  $S$  are lowercase letters of the English alphabet.
- $S_i$  is lexicographically smaller than  $S_{n-i+1}$  for all odd  $i$  from 1 to  $[n/2]$ .
- $S_i$  is lexicographically greater than  $S_{n-i+1}$  for all even  $i$  from 1 to  $[n/2]$ .

( $S_i$  ( $1 \leq i \leq n$ ) denotes the  $i^{\text{th}}$  character of  $S$ ). For example, the word "difference" is a magic string, while "similar" is not.

Given a string  $S$  of lowercase English letters, write a program to find the longest magic string than can be obtained by removing some letters of  $S$ . If there are more than one solutions, choose the longest magic string which is lexicographically smallest.

## Input

The first line contains  $t$ , the number of test cases (about 10). Then  $t$  test cases follow. Each test case contains a string  $S$  written in a single line.  $S$  does not contain more than 2000 letters.

## Output

For each test case, print the longest magic string that is lexicographically smallest in a single line.

## Example

### Input

3

difference

similarq

caaat

### Output

difference

imilaq

aat

[Home](#) » [Compete](#) » [November 2009 \(Contest X\)](#) » Help the judge

## Help the judge Problem Code: J6

A scandal has been revealed in the ruling political party: there is a large case of corruption! You are the judge in the case, and you have a lot of evidence at your disposal.

Unfortunately, some of the statements may potentially contradict the others. Each piece of evidence says something about the involvement of two (not necessary different) politicians in the crime. To be precise, each statement says: "A (is/is not) involved or B (is/is not) involved." Your job is to process all the pieces of evidence, and decide one of three options:

- the evidence is inconsistent,
- you can decide exactly who is involved and who isn't,
- there is some doubt left as to who is involved.

## Input

In the first line of input there is an integer  $t$  ( $1 \leq t \leq 10$ ), representing the number of test cases. The subsequent test cases are defined as follows. In the first line of each test case there is an integer  $n$  ( $1 \leq n \leq 10^5$ ) which denotes the number of politicians. Then, an integer  $k$  follows ( $1 \leq k \leq 3 \cdot 10^5$ ), denoting the number of pieces of evidence. The following  $k$  lines contain the evidence. The description of each statement contains 2 integers,  $u_i, v_i$ . If statement  $i$  says that politician  $A$  is involved, then  $u_i = A$  or  $v_i = A$ , whereas if statement  $i$  says politician  $A$  is not involved, then  $u_i = -A$  or  $v_i = -A$ . There is a blank line after each test case.

## Output

For each test case, write in a separate line of the output one of the words: "CONFLICT", "UNIQUE", or "MULTIPLE", corresponding to the above described possibilities.

## Example

Input:

4

1 0

1 1

1 1

1 1

1 -1

1 2

1 1

-1 -1

**Output:**

MULTIPLE

UNIQUE

MULTIPLE

CONFLICT

DEC09

December 2009  
(Contest XI)01 Dec 2009  
15:00:00

10 days

384

[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » Plant Location

## Plant Location Problem Code: K1

A new highway called Highway 1 has just been built in the Kingdom of Byteland. Ingoo, the largest CPU manufacturer in Byteland, would like to build a new chipset plant along the highway.

Ingoo currently has  $N$  warehouses in the kingdom. None of these warehouses are near the new highway. Chipsets manufactured from the new plant must be delivered to the warehouses. Thus, the further the distances from the new plant to the warehouses, the more the delivery cost Ingoo needs to bear.

Therefore, the CEO of Ingoo would like to find a place on Highway 1 to build the new plant in such a way that the total distance from the plant to all the warehouses is minimized.

You are visiting Byteland on December vacation and have decided to help Ingoo to locate the new plant!

### Input

The first line contains  $T$  (about 20), the number of test cases. Then  $T$  test cases follow. Each test case has the following form:

The first line contains a number  $N$  ( $1 \leq N \leq 2000$ ), the number of Ingoo's warehouses.

The second line contains three integers  $A$ ,  $B$ ,  $C$  which describe the line equation  $(Ax+By+C=0)$  of Highway 1 ( $|A|, |B|, |C| \leq 5000$ ).

The next  $N$  lines contains the coordinates  $(x, y)$  of the warehouses ( $|x|, |y| \leq 5000$ ).

You are guaranteed that  $A$ ,  $B$ ,  $C$  form a valid line equation (i.e.  $A$  and  $B$  are not both 0) and no warehouses lie on Highway 1.

## Output

For each test case, print in a single line the minimum total distance between the new plant and all the warehouses, rounded to 6 decimal places. If there is no optimum distance, print "NO" instead.

## Example

**Input:**

1

5

3 -5 -7

1 3

-2 4

4 -7

7 6

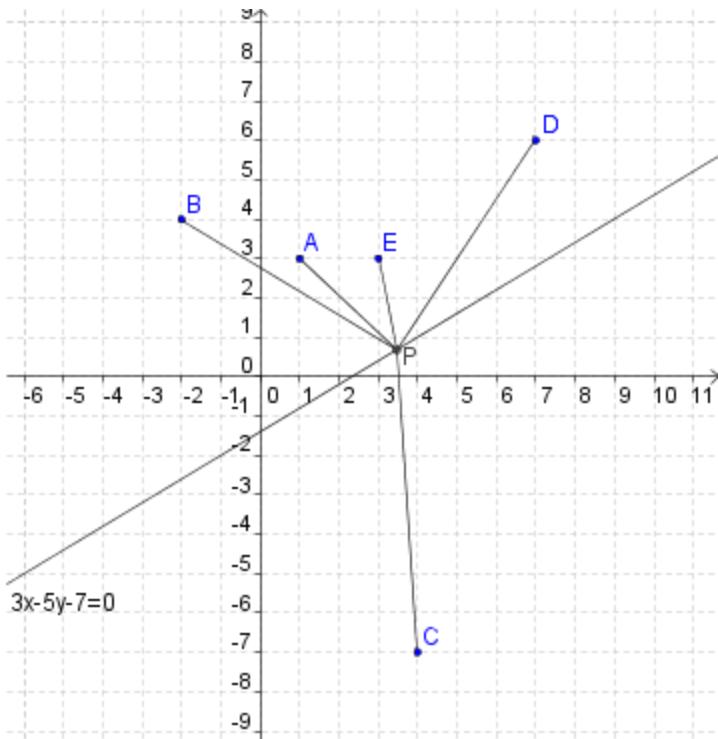
3 3

**Output:**

26.232349

## Output details

The line equation of Highway 1 is  $3x - 5x - 7 = 0$ . Ingoo have 5 warehouses: A(1,3), B(-2,4), C(4,-7), D(7,6), and E(3,3). The optimum location of the new plant is P as shown in the figure below:



[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » Palindromic Numbers

## Palindromic Numbers Problem Code: K2

Johnny has figured out that there are some numbers which have an interesting property: they are the same when read from left to right, and from right to left. For example, 5115 and 929 are such numbers, while 31 and 125 are not. Johnny calls such numbers palindromic numbers.

After a while, Johnny has realized that his definition of palindromic numbers is not really precise. Whether a number is palindromic or not depends on the base in which the number is written. For example, 21 is not palindromic in base 10 but it is palindromic in base 2 (because  $21 = 10101_2$ ).

Johnny finds it interesting that any number becomes palindromic when it is written in an appropriate base.

Given a number  $N$ , write a program to help Johnny compute the smallest base  $B$  such that  $N$  is palindromic when written in base  $B$ .

### Input

The first line contains  $t$ , the number of test cases (about 1000). Then  $t$  test cases follow.

Each test case consists of a number  $N$  written in one line ( $1 \leq N \leq 10^{10}$ ).

## Output

For each given number  $N$ , print a line containing the smallest base  $B$  such that  $N$  is palindromic in base  $B$ .

## Example

**Input:**

3

1

4

21

**Output:**

2

3

2

[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » Maze of Digits

## Maze of Digits Problem Code: K3

Johnny, a third year computer science student of Byteland University, is rushing with his AI assignment project which is due tomorrow evening. In this project, he has to build a Lego NXT robot which is able to navigate in a maze. Johnny calls his robot WallB (because he likes the movie "Walle" very much).

The maze is an  $M \times N$  rectangular grid. There are digits 0..9 and obstacles placed at some intersections. Starting at some intersection, in each step, WallB can move to one of the four intersections adjacent to its current position. However, WallB cannot step into an obstacle.



A maze intersection with a handwritten digit

Johnny needs to program WallB to be able to do the following tasks. First, WallB has to traverse through all reachable intersections of the maze and recognize the digits written at the intersections. Johnny has already finished this part of the assignment since he is very good at image recognition algorithms.



WallB needs to scan and recognize all the reachable digits.

The second part of the assignment is harder. Prof. Q will announce a number  $X$ . Based on the map of the maze obtained in the first part, Johnny needs to program WallB to find a shortest path that passes through digits which sum up to  $X$ .

To be more precise, suppose in his route, WallB passes through the digits  $d_1, d_2, \dots, d_k$  in order; then we should have  $d_1 + d_2 + \dots + d_k = X$ . WallB could pass through a digit more than once.

Johnny gets stuck in this second part of the assignment. Could you write a program to help him?

## Input

The first line contains  $t$ , the number of test cases (about 25). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains two integers M, N ( $0 \leq M, N \leq 100$ ) representing the size of the maze.
- Each line in the next  $M+1$  lines contains  $N+1$  characters describing the structure of the maze. Each character represents an intersection and could be one of:
  - ' ': empty intersection
  - '#': obstacle
  - '0'..'9': digits
  - '\*': starting position of WallB
- The last line contains the number X announced by Prof. Q ( $1 \leq X \leq 100$ ). Successive test cases are separated by blank lines. The number of digits in a maze is at most 100.

## Output

For each test case, print the minimum number of steps that WallB needs to travel to visit the digits that sum up to X. If there is no way for WallB to complete the task, print the number -1.

## Example

**Input:**

3

2 3

1 . #2

# .. #

\* . # .

3

2 3

2 . #2

# .. #

\* . # .

5

```

2 3
2 .#2
#.3#
* .#.
5

```

**Output:**

```

8
-1
6

```

[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » The N Queens Puzzle Revisited

## The N Queens Puzzle Revisited Problem Code: J3

Johnny, a computer science student from the Bytelandian Technological University (BTU), has finally managed to write a program to solve the N queens problem. However, his version of the N queens problem is not really all that similar to the classical one.

Recall that in the N queens problem, there is a square chessboard of size NxN. The goal is to put N queens on the chessboard so that no two queens would be able to attack each other.

In Johnny's version of the problem (which he states to be "much harder" than the original one), there are K "blocked" squares given on the chessboard. The goal is still the same: to put N queens on the chessboard such that no two queens would be able to attack each other. However, there are some differences due to the existence of blocked squares. First, a queen **cannot be put** on a blocked square. Secondly, two queens are considered to be able to attack each other only if there is **no blocked square** between them. Finally, there must be **exactly one** queen on every column of the board.

Johnny claims that his program solves the above problem very effectively. He generates some test cases (which his program can solve within minutes) to challenge you. You know the method Johnny used to generate the test case. For a given N, Johnny randomly picks a number K between 1 and  $N^2/2$  with equal probability. After that, Johnny picks K blocked squares on the chess board. Any combinations of K squares can be picked with equal probability.

Write a program that can solve Johnny's test cases within seconds.

## Input

The first line contains  $t$ , the number of test cases (about 5). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains two integers  $N$  and  $K$  ( $8 \leq N \leq 500$ ,  $1 \leq K \leq N^2/2$ )
- Each line in the next  $K$  lines contains two integers  $i, j$  which are the row and the column number of a blocked square ( $1 \leq i, j \leq N$ )

Each test case's input is separated by a blank line. The rows and the columns of the chessboard are numbered from 1 to  $N$ . There is always a solution for each test case.

## Output

For each test case, print a single line containing  $N$  integers  $a_1, a_2, \dots, a_N$  in which  $a_i$  is the row number of the queen in column  $i$ . Any correct solution will be accepted.

## Example

### Input

1

8 15

7 7

3 1

7 1

4 8

8 3

6 5

4 4

2 1

8 2

6 3

3 7

7 6

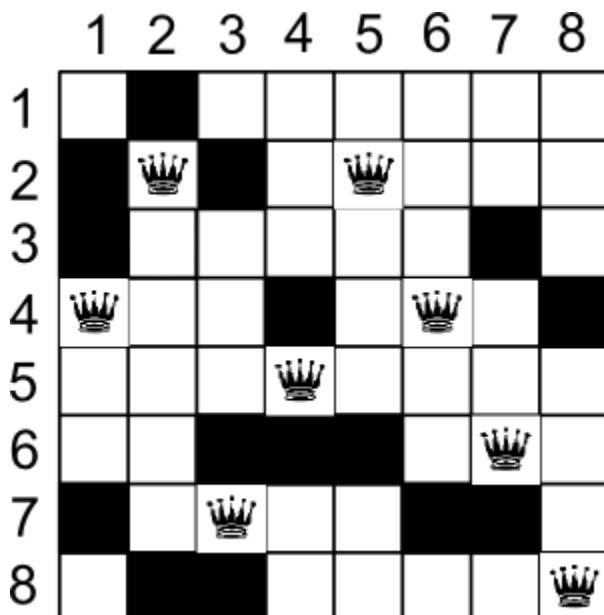
2 3

1 2

6 4

**Output**

4 2 7 5 2 4 6 8



[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » Effective Sets

## Effective Sets Problem Code: J4

Johnny has been hired to write a program to analyse strategic data for the Bytelandian Navy. His task is described below.

There are  $N$  battleships represented as  $N$  points on the plane. The **radius** of a ship is defined to be the distance from that ship to the ship closest to it. The **effective set** for a ship is defined to be the set of all ships (excluding itself) which are at a distance from that ship of **no more than twice** its radius. The number of elements in the effective set of a ship is called the **effective value** of that ship.

Johnny needs to write a program to calculate the radius and the effective value of each battleship. This task is so tough that without your help, Johnny could never finish it. Let's help him!

## Input

The first line contains  $t$ , the number of test cases (about 5). Then  $t$  test cases follow. Each test case has the following form:

- The first line contains an integer  $N$ , the number of battleships ( $1 \leq N \leq 30000$ ).
- Each line in the next  $N$  lines contains two integers  $X, Y$  ( $0 \leq X, Y \leq 10000$ ) represents the coordinates of a battleship.

Each test case's input is separated by a blank line. There may be more than one ships located at the same place.

## Output

For each test case, print  $N$  lines. Each line should contain two numbers which are the radius and the effective value of the corresponding battleship. The radius should be rounded to 2 decimal points.

Print a blank line after each test case's output.

## Example

### Input

2

3

0 0

0 0

3 4

5

5 3

7 8

0 9

3 1

4 4

#### Output

0.00 1

0.00 1

5.00 2

1.41 2

5.00 4

6.40 4

2.83 2

1.41 1

[Home](#) » [Compete](#) » [December 2009 \(Contest XI\)](#) » Park tour

## Park tour Problem Code: KX

Imagine a large  $n \times n$  square park composed of small unit squares. Some squares are occupied by trees, and the rest is free to walk through them. Your job is to create a visiting route which should be as long as possible. The route should be a closed curve with no self-intersections, and is defined by describing the sequence of squares through which it passes. From one square, the route can only proceed to a square which shares a common edge with it. The route enters and leaves each square in the central points of two of its edges. If those two points lie on opposing edges of the square (N - S or E - W), the fragment of the route within the square is a straight line segment of length  $l=1$ . Otherwise, if the two points lie on adjacent edges, the route within the square is an arc of the circle (1/4th of the full circle of radius 1/2, centered at the common corner of the edges) and its length is  $l=\pi/4$ .

#### Input

In the first line of input there is an integer  $n$  ( $100 \leq n \leq 800$ ), denoting the size of park. The following  $n$  lines of  $n$  characters each describe the park: the character '\*' represents a square with a tree, and the character '.' means that the square is empty.

## Output

First, an integer  $k$  ( $0 < k$ ), corresponding to the number of lines of the subsequent description of the output. The following  $k$  pairs of numbers should describe the squares of the route:  $i_1, j_1 \dots i_k, j_k$ , where  $i$  is the row number (counting from 0) and  $y$  is the column number (counting from 0) of a square.

## Correctness of the output

For each  $i < k$ , squares  $i$  and  $i+1$  of the route should share a common edge, and moreover the first and the  $k$ -th squares should also share a common edge. The route may not make a U-turn at any square, and may not lead through any squares with trees. The route should not self-intersect or touch itself. The route is not allowed to lead outside the park. Note that it is possible for the route to be twice in the same square (i.e. once enter from N exiting due E, later enter from S exiting due W).

## Example

Input:

3

..\*

...

\*..

Output:

8

0 0

0 1

1 1

1 2

2 2

2 1

1 1

1 0

Score:

0.897598

## Test distribution

$n$  will be chosen uniformly from the range [100,800] range. The number of trees will be chosen uniformly from the range  $[n^2 \cdot \text{alfa}, n^2 \cdot 2 \cdot \text{alfa}]$ , where  $\text{alfa} = 0.1$  for 1/3 of the testcases,  $\text{alfa} = 0.02$  for 1/3 of the testcases, and  $\text{alfa} = 0.004$  for 1/3 of the testcases. Positions of trees are chosen uniformly at random.

## Scoring

The score is given as  $I / f$ , where  $I$  is the total length of the route and  $f$  is the total number of squares without trees.

Imagine a large  $n \times n$  square park composed of small unit squares. Some squares are occupied by trees, and the rest is free to walk through them. Your job is to create a visiting route which should be as long as possible. The route should be a closed curve with no self-intersections, and is defined by describing the sequence of squares through which it passes. From one square, the route can only proceed to a square which shares a common edge with it. The route enters and leaves each square in the central points of two of its edges. If those two points lie on opposing edges of the square (N - S or E - W), the fragment of the route within the square is a straight line segment of length  $l=1$ . Otherwise, if the two points lie on adjacent edges, the route within the square is an arc of the circle (1/4th of the full circle of radius 1/2, centered at the common corner of the edges) and its length is  $l=\pi/4$ .

## Input

In the first line of input there is an integer  $n$  ( $100 \leq n \leq 800$ ), denoting the size of park. The following  $n$  lines of  $n$  characters each describe the park: the character '\*' represents a square with a tree, and the character '.' means that the square is empty.

## Output

First, an integer  $k$  ( $0 < k$ ), corresponding to the number of lines of the subsequent description of the output. The following  $k$  pairs of numbers should describe the squares of the route:

$i_1, j_1 \dots i_k, j_k$ , where  $i$  is the row number (counting from 0) and  $y$  is the column number (counting from 0) of a square.

## Correctness of the output

For each  $i < k$ , squares  $i$  and  $i+1$  of the route should share a common edge, and moreover the first and the  $k$ -th squares should also share a common edge. The route may not make a U-turn at any square, and may not lead through any squares with trees. The route should not self-intersect or touch itself. The route is not allowed to lead outside the park. Note that it is possible for the route to be twice in the same square (i.e. once enter from N exiting due E, later enter from S exiting due W).

## Example

Input:

3

..\*

...

\*..

Output:

8

0 0

0 1

1 1

1 2

2 2

2 1

1 1

1 0

Score:

0.897598

## Test distribution

$n$  will be chosen uniformly from the range [100,800] range. The number of trees will be chosen uniformly from the range  $[n^2 \cdot \text{alfa}, n^2 \cdot 2 \cdot \text{alfa}]$ , where  $\text{alfa} = 0.1$  for 1/3 of the testcases,  $\text{alfa} = 0.02$  for 1/3 of the testcases, and  $\text{alfa} = 0.004$  for 1/3 of the testcases. Positions of trees are chosen uniformly at random.

## Scoring

The score is given as  $l / f$ , where  $l$  is the total length of the route and  $f$  is the total number of squares without trees.

JAN10	<a href="#">January 2010 (Contest XII)</a>	01 Jan 2010 15:00:00	14 days	389
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Lost Primes

## Lost Primes Problem Code: L3

The BSA (Bytelandian Security Agency) has intercepted several secret keys from the Trojan Kingdom. Each secret key is a prime number. But BSA was only able to retrieve some part of the digits, not all of them!

Johnny's job is to help BSA recover these lost primes, and he has delegated the task to you!

### Input

The first line contains a number  $t$  (about 15) which is the number of test cases. Then  $t$  test cases follow. Each test case is described in a single line containing the patterns of the lost primes (the unknown digits are represented by '?'s).

Each pattern's length is at most 12.

### Output

For each test case, print the recovered prime. If there are multiple solutions, print any of them. The recovered prime should have the same number of digits as the corresponding pattern and should contain no leading zeros.

You can assume that there is at least one solution for each test case.

## Example

### Input:

3

?

?3

1??

### Output:

5

23

101

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Circle of towers

## Circle of towers Problem Code: L4

Two friends live in a small city. The city is built in the form of a circular road and skyscrapers surrounding the road. But one day the two friends quarreled with each other, and although they remained in the city, they decided to live in apartments located as far as possible from each other, or more precisely, so as to maximize the travel time between them. All the skyscrapers are arranged in a circle around the road (so you can travel in a cycle as follows: 1 <-> 2 <-> 3 <-> ... <-> n <-> 1), and the travel time between two neighbouring buildings is 1. The travel time between two adjacent floors of a skyscraper is also 1, and the friends can choose to live in any floor of the building, from the 0-th to the k-th, assuming that the skyscraper in question has k floors. Thus, the travel time from the j-th floor to ground level is j time units.

### Input

First,  $1 \leq n \leq 10^6$ , the number of skyscrapers. The following numbers,  $1 \leq h_1 \leq 10^9, \dots, 1 \leq h_n \leq 10^9$ , are the heights of the buildings, given in clockwise order.

### Output

Output exactly one number, the maximal possible travel time.

## Example

Input:

4 1 2 3 4

Output:

8

Input:

10 1 7 4 3 2 9 2 3 4 5

Output:

20

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Cell Phone Towers

## Cell Phone Towers Problem Code: L1

Byteline is currently the largest mobile phone operator in Byteland. They have  $N$  cellphone towers at different places in Byteland.

Byteland can be viewed as a two dimensional plane, with each tower or skyscraper described by a point in that plane. Of course, no two points have the same coordinates.

Two towers can only communicate with each other if there is no tower nor skyscraper lying on the straight line segment connecting them.

As the most skillful programmer in the company, Johnny's job is to write a program to help Byteland compute the number of pairs of towers which can communicate with each other.

And once again, Johnny has asked for your help!

### Input

The first line contains a number  $t$  (about 15) which is the number of test cases. Then  $t$  test cases follow. Each test case has the following form.

The first line contains two numbers  $N$  and  $M$  ( $1 \leq N, M \leq 1000$ ), the number of towers and skyscrapers in Byteland.

Each line in the next  $N$  lines contains two integers representing the coordinates of the cellphone towers.

Finally, each line in the next  $M$  lines contains two integers representing the coordinates of the skyscrapers.

All coordinates have absolute values not larger than 10000.

## Output

For each test case, print a single number, describing the number of pairs of towers which can communicate with each other.

## Example

### Input

1

5 4

-1 -1

1 -1

0 -2

-2 -2

0 0

-1 -2

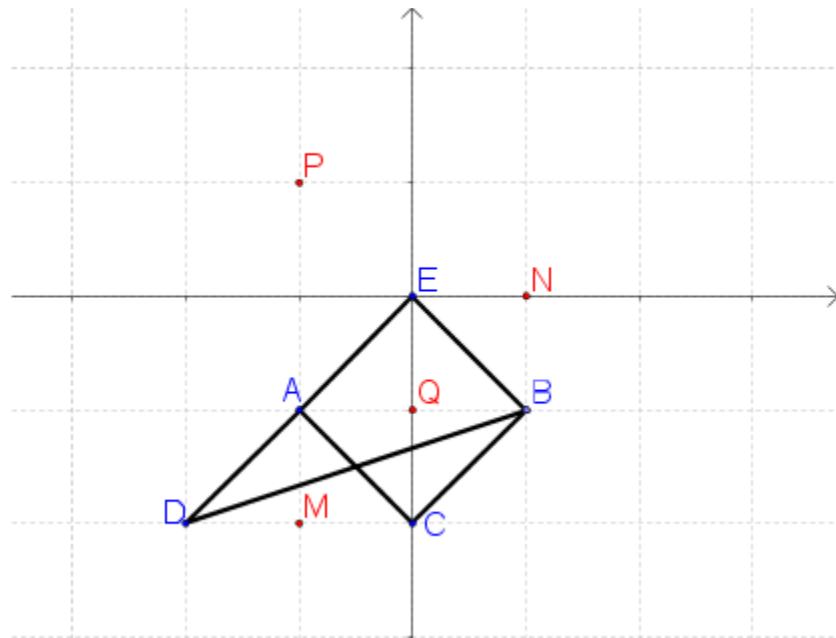
1 0

-1 1

0 -1

### Output

## Output details



In the above figure, the blue points represent the cell towers and the red points represent the skyscrapers as in the example input. The 6 segments AE, BE, BC, AC, AD and BD correspond to the 6 pairs of towers that can communicate with each other.

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Jawbreaker

## Jawbreaker Problem Code: LX

Jawbreaker is a one-person game. You start with a square of dimension  $n \times n$ , composed of small colored stones. One move consists of indicating one stone. Then, the maximal possible connected group of stones the of the same color, containing the indicated stone, is destroyed. The destroyed rocks disappear, and all rocks which have empty space below them drop to a lower row. If any column is left empty, rocks that are to the right of it are shifted accordingly to the left. When a group of  $k$  stones is destroyed, you score  $k^2$  points. Your task is to maximize the total score.

## Input

First,  $50 \leq n \leq 100$ , the size of the starting board. Then,  $n$  lines follow,  $n$  characters long each, containing a description of the board, with up, down, left, right oriented just as we see it on a computer screen (upper rows first, left columns first). The color of each stone is described by one character, from '0' to '9'.

## Output

First, output  $t$ , the length of answer. The following  $t$  pairs of integers are a sequence of moves. Each pair  $i,j$  defines one move,  $i$  being the 0-based number of a column (column 0 is the one most to the left), and  $j$  being the 0-based number of a row (row 0 is the lowest one).

## Example

Input:

3

202

202

110

Output:

3

0 0

2 0

1 1

Moves:

202

202

110

..2

202

200

...

22.

22.

Score:

$2^2 + 3^2 + 4^2$

## Tests

The size of the tests will be chosen uniformly between 50 and 100, and then the number of colors will be chosen uniformly between 4 and 10. The colors of the rocks are chosen independently at random.

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Wildcard Matching

## Wildcard Matching Problem Code: L2

Johnny is the author of the famous open source software ByteDict (Bytelandian electronic dictionary).

Being a fan of the word game Scrabble, Johnny would like to make ByteDict useful for Scrabble training. Therefore, he has decided to implement the "Wildcard Matching" feature.

Wildcard matching allows ByteDict's users to search for words matching a given wildcard pattern. A wildcard pattern may include the characters '\*' or '?' in addition to the characters of the alphabet. A '\*' matches any sequence of characters (including the empty sequence) and a '?' matches any single character.

To be more precise, a wildcard pattern is matched with a word if we can replace each '?' with an alphabet character and each '\*' with a sequence of alphabet characters so that after replacement, the pattern becomes identical to the word.

Now, as usual, please help Johnny implement the feature!

### Input

The first line contains a number  $t$  (about 2) which is the number of test cases. Then  $t$  test cases follow. Each test case has the following form.

The first line contains two integers  $N$  and  $Q$  ( $1 \leq N \leq 55000$ ,  $1 \leq Q \leq 300$ ) which are the number of words in the dictionary and the number of wildcard queries.

Each line in the next  $N$  lines contains a word from the dictionary. The alphabet consists of lower and uppercase characters as well as digits.

Each line in the next  $Q$  lines contains a wildcard pattern.

The length of each word does not exceed 25 and the length of each wildcard pattern does not exceed 5.

Each test case is separated by a blank line.

## Output

For each wildcard query print a line containing the query itself and the number of words in the dictionary that match the query.

Print a blank line after each test case's output.

## Example

**Input:**

2

1 7

abc

abc

cba

a??

??a

\*?a

\*\*\*

\*\*a

7 4

spoj

codechef

uva

tju

onlinejudge

Acmicpc

Worldcup2010

\*a\*

sp?\*

\*c\*

uvc

**Output:**

abc 1

cba 0

a?? 1

??a 0

\*?a 0

\*\*\* 1

\*\*a 0

\*a\* 1

```
sp?* 1
```

```
*c* 3
```

```
uvc 0
```

[Home](#) » [Compete](#) » [January 2010 \(Contest XII\)](#) » Place the balloon

## Place the balloon Problem Code: L5

Given a set of points on a plane, find the largest possible disc which does not "cover" any of these points (i.e., they can lie on the boundary of the disc, but not inside it) and which is constrained by this set of points (i.e., the disc cannot be moved to infinity along a continuous trajectory).

### Input

Input will be consist of several test cases. First will be given  $1 \leq t \leq 20$ , then number of testcases. Then  $t$  testcases follow. Each testcase is of following form. First,  $1 \leq n \leq 50000$ , the number of points. The following  $n$  lines define the points; the  $i$ -th line contains two integers:  $-10000 \leq x_i, y_i \leq 10000$

### Output

Output exactly one value: the radius of largest possible disc satisfying the given property. If there is no such disc, output the value  $-1$ . Answers should be exact up to  $10^{-6}$

### Example

Input:

```
2
```

```
3
```

```
0 1
```

```
-1 -1
```

```
1 -1
```

```
4
```

```
0 1
```

-1-1

1 -1

0 0

Output:

1.25

-1

FEB10	<a href="#">February 2010 Contest</a>	01 Feb 2010 15:00:00	10 days	470
-------	---------------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [February 2010 Contest](#) » Soccer League

## Soccer League Problem Code: M3

The new season of the Bytelandian Premier League (BPL) has started!

In the BPL, any two soccer teams play with each other exactly once. In each match, the winner earns 3 points and the loser earns no point. There is no draw (if the match is level after the two halves, two teams will take part in a penalty shootout to decide the winner).

At the end of the league, the winner is the team having the largest number of points. In case there are more than one team which has the largest number of points, these teams will be co-champions of the league.

The league has been running for some time. Now, the following problem has arisen: we would like to know if a specific team still has a chance of winning the league.

### Input

The first line contains  $T$  (about 20), the number of test cases. Then  $T$  test cases follow. Each test case has the following form.

The first line of the test case contains a number  $N$  ( $1 \leq N \leq 140$ ), the number of teams in the league.

The  $i$ -th line in the next  $N$  lines contains  $N$  numbers  $a_{i1}, a_{i2}, \dots, a_{in}$ . The number  $a_{ij}$  gives the status of the match between the  $i$ -th team and the  $j$ -th team:

- $a_{ij} = 1$  if the  $i$ -th team wins,
- $a_{ij} = 0$  if the  $i$ -th team loses,

- $a_{ij} = 2$  if the match has not taken place yet.

The input data is such that if  $i \neq j$ , then  $a_{ij} + a_{ji} = 1$  or  $a_{ij} = a_{ji} = 2$ . Moreover,  $a_{ii} = 0$  for all  $i$ .

## Output

For each test case, print a binary string of length  $N$ , in which the  $i$ -th character is 1 if the  $i$ -th team still has a chance to be a champion of the league, and 0 otherwise.

## Example

### Input:

3

3

0 0 0

1 0 1

1 0 0

4

0 1 1 0

0 0 2 0

0 2 0 0

1 1 1 0

5

0 2 2 1 0

2 0 1 1 0

2 0 0 1 0

0 0 0 0 1

1 1 1 0 0

### Output:

010

0001

11001

[Home](#) » [Compete](#) » [February 2010 Contest](#) » Sorting device

## Sorting device Problem Code: MX

You work for a large company, and your job is to design sorting devices. Devices are built from:

- $n$  inputs,
- $n$  outputs,
- gates which have two inputs and send to output the minimum ("min gates") or maximum ("max gates") of the two numbers given at their input,
- connecting wires.

You are competing in a bid for a Bytelandian Ministry of Information contract to design the smallest possible device (in terms of the number of gates) which sorts any input. Each device will go through a rigorous test process on a number of data sets. However, through some Good Friends in High Places you have managed to acquire some additional information concerning the exact data sets (permutations) your device will be tested on. Make use of this, and of course your superior programming skills, to win the bid!

### Input

First, two numbers,  $2 \leq n \leq 20$ ,  $1 \leq k \leq 1000$ , the number of inputs and the number of different permutations for which your device will be tested. The next  $k$  lines contain permutations of the numbers  $1 \dots n$ .

### Output

First, output  $p$ , the number of gates in your device ( $0 \leq p \leq 10^6$ ). The next  $p$  lines should contain definitions of the gates used, in the form of a pair of integers,  $x_i, y_i$ , and exactly one of the strings "min" or "max". To be able to use the output of a gate as the input of a subsequent gate, we use the following convention. First of all, the range for inputs  $x_i$  and  $y_i$  is as follows:  $1 \leq x_i, y_i < n+p$ . The output of the  $i$ -th gate is assumed to be input  $n+i$ .

Finally, a sequence of exactly  $n$  integers in the range  $1..n+p$  should follow, describing which of the "inputs" should be hard-wired to successive outputs of the device.

### Scoring

If your program does not produce a sorting machine which works for every input (sorting it in ascending order), it will be deemed incorrect. Otherwise, you will score  $p$  penalty points for each test case solved.

## Example

### Input:

```
3 2
1 2 3
1 3 2
```

### Output:

```
2
2 3 min
2 3 max
1 4 5
```

### Score:

0.5

[Home](#) » [Compete](#) » [February 2010 Contest](#) » The Mine Field

## The Mine Field Problem Code: M1

It's 2012. The times of peace are over. The Trojan Kingdom has just declared war on Byteland! As part of a massive defense plan against the Trojans, the Bytelandian Ministry of Defense (BMoD) wants to lay mines on a field near the borderline of the country.

The field has the form of a  $M \times N$  rectangular grid divided into  $M \times N$  unit-square cells. It is only possible to plant mines on some cells, and the number of mines on a cell cannot exceed one.

The objective is obviously to lay as many mines as possible. However, the BMoD does not want any two adjacent cells to both have mines, because such a situation would be dangerous: one mine being triggered could lead to a chain of explosions! Two cells are adjacent if they share a common edge.

Johnny, the most renowned computer scientist in Byteland, needs to compute the maximum number of mines that could be planted on the field, and the number of different ways in which these mines could be planted.

Hurry up and help Johnny save his beloved country!

### Input

The first line contains  $T$  (about 15), the number of test cases. Then  $T$  test cases follow. Each test case has the following form.

The first line contains two numbers M and N representing the size of the field ( $1 \leq M, N \leq 20$ ).

Each line in the next M lines contains N characters. Each character is either '.' or '\*' representing the status of a cell in the field. A character '.' represents an empty cell (i.e. it is possible to lay a mine on that cell), while a character '\*' represents a blocked cell (i.e. there are obstacles on that cell which prevent mines from being laid on it).

Input for successive test case is separated by a blank line.

## Output

For each test case, print two numbers on a line. The first number is the maximum number of mines that could be planted on the field. The second number is the number of ways to plant the maximum number of mines on the field, modulo 151109.

## Example

### Input:

2

2 3

...

.\*.

3 3

...

.\*.

...

### Output:

3 1

4 2

## Toll Trolls Problem Code: M4

Johnny has gone on a short excursion to the island of Bitland, and found himself with plenty of time to spare for sight-seeing. As a matter of fact, he does really have any itinerary at hand. All he would like to do is to walk around a bit, without spending too much money in the process.

The cities of Bitland are connected by a network of one way streets. If there exists a road leading directly from city A to city B, then there also exists a road directly from city B to city A. There maybe any number (possibly zero) of parallel roads between any two cities. However, it is possible to reach any city starting from any other city (possibly indirectly, going through other cities on the way).

Now, the trouble is that all of the roads in Bitland are privately owned, and toll trolls have been hired to guard them and collect the dues. What's worse, the trolls are moody and will in some circumstances change the sum of the toll charged for using a road...

Johnny starts his sight-seeing tour from the capital of Bitland. He always leaves each city by the cheapest outgoing road, i.e. the one which currently has the lowest toll. However, as soon as Johnny has used the road from A to B, the toll troll in charge of it increases the toll for the road, setting it at exactly 1 Bitlandian dinar more than the toll currently being charged on the *most expensive* of the roads leaving city A.

Johnny has been walking around the towns of Bitland for a while, and he is getting a little bored. Help him answer the following question: what is the toll he is going to pay when traversing the *k*-th road of his tour?

### Input

The first line of input contains two integers: *n*, the number of towns in Bitland and *m*, the number of pairs of roads connecting them ( $1 \leq n \leq 10^5$ ,  $0 \leq m \leq 10^5$ ). Each of the *m* following lines contains four integers *A* *B* *c<sub>AB</sub>* *c<sub>BA</sub>*, denoting the identifier of city A, the identifier of city B, the initial toll on the road from A to B, and the initial toll on the road from B to A ( $1 \leq A < B \leq n$ ,  $1 \leq c_{AB}, c_{BA} \leq 10^9$ ). All tolls are expressed in dinars. You may assume that the initial toll values for all of the roads are different, and that the city labeled 1 is the capital of Bitland.

The next line of input contains the number *t* of queries asked by Johnny,  $t \leq 10^4$ . Each of the following *t* lines contains a single integer *k* ( $1 \leq k \leq 10^{12}$ ).

### Output

For each of the values of *k* given at input, print a line containing the toll paid by Johnny when traversing the *k*-th road along his route.

## Example

### Input:

3 2

1 2 5 10

1 3 11 7

4

1

2

3

5

### Output:

5

10

11

12

[Home](#) » [Compete](#) » [February 2010 Contest](#) » Overlapping discs

## Overlapping discs Problem Code: M5

You are given  $n$  discs on plane, of the same radius ( $r=1$ ). Output the area of the part of the plane which is covered by **each** disc.

### Input

First,  $1 \leq t \leq 10$ , the number of tests. Each testcase is of the following form:  $1 \leq n \leq 100000$ , then  $n$  pairs of floating point coordinates follow:  $-1000 \leq x_i, y_i \leq 1000$ , representing the location of the  $i$ -th disc.

### Output

For each testcase, output the total area covered by the intersection of all discs. Round the answer to 6 decimal digits after the point.

## Example

Input:

1

2

0 0

0 1

Output:

1.228370

[Home](#) » [Compete](#) » [February 2010 Contest](#) » Motorbike Racing

## Motorbike Racing Problem Code: M2

It's time for the annual exciting Motorbike Race in Byteland.

There are  $N$  motorcyclists taking part in the competition. Johnny is watching the race. At the present moment (time 0), Johnny has taken note of the current velocity and position of each motorcyclist.

Johnny wants to know at a given point of time, which motorcyclist is in a specific place in the rank list. Please help him!

If at any given time two motorcyclists are in same position, the motorcyclist with the smaller index will be placed before the one with the larger index.

To make the problem simple, Johnny assumes that each motorcyclist is moving at a constant velocity.

## Input

The first line contains a number  $t$  (about 10) which is the number of test cases. Then  $t$  test cases follow. Each test case has the following form.

The first line of the test case contains a number  $N$  ( $1 \leq N \leq 2000$ ), the number of motorcyclists.

The  $i$ -th line in the next  $N$  lines contains two numbers,  $v$  and  $x$ , which are the velocity and the current position of the  $i$ -th motorcyclist ( $1 \leq v, x \leq 100,000$ ).

The next line contains a number  $Q$  ( $1 \leq Q \leq 2000$ ), the number of time queries.

Each line in the next  $Q$  lines contains two numbers,  $t$  ( $1 \leq t \leq 1,000,000,000$ ) and  $k$  ( $1 \leq k \leq n$ ), representing the query: "at time  $t$ , which motorcyclist is positioned  $k$ -th in the rank list?"

## Output

For each test case, print  $Q$  lines, with each line containing the index of the motorcyclist for the corresponding query.

Remember to print a new line after each test case.

## Example

**Input:**

1

4

2 100

3 50

4 60

5 1

4

1 1

50 2

60 4

100 1

**Output:**

1

4

1

4

MARCH10

[March 2010 Contest](#)01 Mar 2010  
15:00:00

12 days

262

[Home](#) » [Compete](#) » [March 2010 Contest](#) » Tri-Hexagonal Puzzle

## Tri-Hexagonal Puzzle Problem Code: N4

Little Johnny wants to play with his friends today. But his babysitter won't let him go! After a lot of begging, the heartless nanny gives him her brand new electronic puzzle and says: "If you solve the puzzle then you are free to go". Not being aware of Little Johnny's IT skills, the nanny leaves the kid alone.

Rapidly, Little Johnny sends you an e-mail asking for your help.

The puzzle consists of three hexagons as shown in the figure. Each vertex is painted black or white. Some of them belong to just one hexagon and some of them belong to more than one. Exactly four of them are painted black, and the other nine are white. The goal is to make the shared vertexes black by means of allowed moves: rotating a hexagon 60 degrees clockwise or counter-clockwise.

Can you help Little Johnny?

### Input

Input starts with an integer  $T$ , the number of puzzles to solve ( $1 \leq T \leq 100$ ).  $T$  lines follow, each one having 13 binary digits, corresponding to the top-down, left to right labeling of the vertexes in the puzzle. A '0' means the  $i$ -th vertex is painted white, while a '1' means it is painted black.

### Output

For each test case output  $M$  on a single line, the minimum number of moves required to solve the puzzle. Then print  $M$  lines, each one describing a move in the following manner: two space separated integers  $H$  and  $C$ , the rotated hexagon (upper left is 0, upper right is 1, lower one is 2) and the direction (0 for counter-clockwise, 1 for clockwise).

If there is more than one solution, any will suffice.

## Example 1

Input:

1

0000000101011

Output:

3

2 0

2 0

1 1

[Home](#) » [Compete](#) » [March 2010 Contest](#) » Treasure Hunting

# Treasure Hunting Problem Code: N1

Treasure Hunting is a great computer game that has attracted generations of Bytelandian children.

In the game, there is a maze divided into  $N \times N$  squares. Dave starts in the top-left corner, which is square  $(1,1)$ , and needs to go to the bottom-right corner, which is square  $(n,n)$ .

Some of the squares are blocked and some of the squares contain treasures.

Dave needs to capture all the treasures in the maze before going to square  $(n,n)$ .

In each second, Dave can go to one of its four adjacent squares (if the destination is not blocked).

Find the earliest time that Dave can reach the destination  $(n,n)$  after collecting all the treasures in the maze.

## Input

The first line contains  $t$ , the number of test cases (about 15). Then  $t$  test cases follow. Each test case has the following form.

The first line contains  $N$  ( $1 \leq N \leq 13$ ), the size of the maze

The  $N$  following lines describe the maze. The meaning of the symbols is as follows:

'.' : an empty square

'\*' : a treasure

'#' : a blocked square

The number of treasures in the maze does not exceed 13. Squares  $(1,1)$  and  $(n,n)$  are always empty.

Each test case's input is separated by a blank line.

## Output

For each test case, print in a single line the earliest time that Dave can reach the destination after collecting all the treasures. If Dave cannot reach the destination, print -1.

## Example

**Input:**

4

3

...

.##

\* #.

3

.. \*

...

...

3

.. \*

\* ..

...

4

...

. #. \*

. #\*. \*

\* \* #.

**Output:**

-1

4

6

16

[Home](#) » [Compete](#) » [March 2010 Contest](#) » K-important Strings

## K-important Strings Problem Code: N3

You are given a set of  $N$  strings  $S_0, S_1, \dots, S_{N-1}$ . These strings consist of only lower case characters  $a..z$  and have the same length  $L$ .

A string  $H$  is said to be  $K$ -important if there are at least  $K$  strings in the given set of  $N$  strings appearing at  $K$  different positions in  $H$ . These  $K$  strings need not to be distinct.

Your task is to find the shortest  $K$ -important string. If there are more than one possible solution, your program can output any of them.

### Input

The first line contains a number  $t$  (about 10) which is the number of test cases.

Each test case has the following form.

The first line contains three integers  $N$ ,  $L$  and  $K$ . The next  $N$  lines contain the strings in the given set.

Each test case's input is separated by a blank line.

### Constraints

- $1 \leq N \leq 150$
- $1 \leq L \leq 300$
- $1 \leq K \leq 500$

### Output

For each test case, output the following information.

The first line contains the length of the shortest  $K$ -important strings.

The second line contains H, one of the K-important strings.

Each line in the next K lines contains the index of one string in the given set that appears in H and the corresponding position (0-based) in H.

Print a blank line after each test case's output.

## Example

### Input

3

3 3 1

abc

cde

bcf

3 3 2

abc

cde

bcf

3 3 3

abc

cde

bcf

### Output

3

abc

0 0

4

abcf

0 0

2 1

7

abcfabc

0 0

2 1

0 4

[Home](#) » [Compete](#) » [March 2010 Contest](#) » The Best Tower

## The Best Tower Problem Code: NX

ByteTel, the largest mobile operator in Byteland, would like to find a place to construct a cellphone tower. There are N important places in Byteland that need to be considered.

These N buildings can be represented by N points in a 2D plane. The cellphone tower can be represented by a circle, and in order to construct the tower, ByteTel needs to specify its center and radius.

Moreover, the center of the tower cannot be outside the secure area, which is also a circle.

ByteTel wants to find a center and radius for the tower such that **the total squared distance** between each of the building and the tower is as small as possible.

The distance between a point and a circle is defined to be the difference between the distance from the circle's center to the point and the circle's radius.

Your task is to help ByteTel find a center and radius for their cellphone tower.

## Input

The first line contains a number  $t$  (about 10), which is the number of test case. Each test case has the following form.

The first line contains three numbers ( $X_s$ ,  $Y_s$ ) and  $R_s$ , specifying the secure circle.

The second line contains  $N$ , the number of buildings ( $1 \leq N \leq 100$ ).

Each line in the next  $N$  lines contain two numbers which are the coordinates of a building.

All coordinates are integer and are between 0 and 10000 (inclusive).

## Output

For each test case, output 3 numbers ( $X, Y$ ) and  $R$ , rounded to 3 decimal digits, which are the center and radius of the cellphone tower.

## Scoring

For each test case, your score is  $d/D$  in which:

$d$  is the total square distance between each of the building and the tower your program has found

$D$  is the total square distance between each of the building and the secure circle's center.

Your program's score is the sum of each test case's score, the lower the better.

## Example

**Input:**

1

1 1 4

5

3 -2

2 1

5 3

4 3

4 -1

**Output:**

4.000 1.000 2.000

The sample output's score is 0.023444. The secure circle is red and the tower is yellow.

[Home](#) » [Compete](#) » [March 2010 Contest](#) » Traffic jam

## Traffic jam Problem Code: N5

Little Johnny has a selection of boxes. Each box has a number on its side. The boxes are placed in a sequence, and Johnny wants to sort them (in ascending order). He has a device to manipulate the boxes, which performs the following operation. Johnny can select a subset of boxes, and the machine will lift the selected subset, shift the selected subset to the right (keeping the order in the subset), shift the not-selected subset to the left, filling up empty spaces (keeping the order in the subset), then finally move the raised boxes to be in one level again.

For example: if Johnny has the sequence: 1,2,3,4,5,6, and selects the subset in bold: 1,2,3,4,5,6, then the result is: 1,2,5,3,4,6.

Help Johnny to write a program that will calculate the minimal number of moves required to sort the given sequence of boxes in ascending order of numbers.

### Input

First,  $1 \leq t \leq 10$ , the number of test cases. Then,  $t$  testcases follow. Each starts with  $1 \leq n \leq 10^5$ , the number of boxes. Then,  $n$  integer values describing the sizes of boxes in the sequence.

### Output

For each testcase, in a separate line, print the answer to that testcase.

### Example

**Input:**

1

6 1 3 5 2 4 6

Output:

2

[Home](#) » [Compete](#) » [March 2010 Contest](#) » Heroes of Magical Might

## Heroes of Magical Might Problem Code: N2

You are playing the game Heroes of Magical Might. Your hero is in a grid map divided into  $M \times N$  squares.

There are three kinds of possible objects in each square:

- Obstacles: these can be trees, mountains, houses, and so on. Your hero cannot walk into these squares. These squares are represented by the characters '#'.
- Treasures: your hero can find some gold or earn experience points when collecting these treasures. These squares are represented by the characters '\*'.
- Gates: these gates form a complex interconnected network in the map. Your hero can travel almost instantly between the gates, represented by the characters '@'.

You control the hero turn by turn. In each turn, you can:

Choose one of the four possible directions, and move to the adjacent square in that direction. However, due to fog, your hero will only succeed with probability  $P$ . He may go in the remaining 3 directions with equal probability  $(1-P)/3$ . If the adjacent square is an obstacle or is outside the maze, the hero remains in the current square.

In addition, if your hero's current square is a gate, your hero could choose to enter the gate. He will appear in one of the remaining gates with equal probabilities, i.e. with probability  $1/(\text{Number of Gates} - 1)$ .

..@.

@ # @

For example, in the above map, suppose your hero's starting position is at (1,1) and  $P = 0.8$ . If the hero moves down in the first turn, he may succeed and end up in the gate at (2,1) with probability 0.8. With probability  $0.2 / 3$ , he may go right and end up in the gate at (1,2). With probability  $2 * 0.2 / 3$ , he just remains at (1,1) since the top and left direction would lead him out of the maze.

If the hero chooses to enter the gate at (2,1), he may end up at (1,2) or (2,3) with equal probability 1/2.

Find the optimum strategy to control your hero such that *the expected number of turns to collect the first treasure* is minimized.

## Input

The first line contains  $t$ , the number of test cases (about 15). Then  $t$  test cases follow, preceded by empty lines. Each test case has the following form.

The first line contains two numbers  $M$  and  $N$  ( $1 \leq M, N \leq 20$ ), describing the dimensions of the map.

Each line in the next  $M$  lines contains  $N$  characters describing the maze. The meanings of the characters are:

- '\*' : treasure
- '#' : obstacle
- '.' : empty square
- '@' : gate
- 'S' : starting point of the hero

You are guaranteed that the hero can always collect at least one treasure with probability greater than 0. The number of gates will never be 1. There is exactly one 'S' character in the maze.

## Output

For each test case, output the minimum expected number of turns until the hero collects the first treasure, rounded to two decimal places.

## Example

**Input:**

4

1 2

S\*

0 . 7

1 2

S \*

0

2 4

S # # \*

@ # # @

0 . 8

2 18

S . . . \* . # @ \* . . . . . \* @ .. @ . . . . # . . . . . . . . .

0 . 6

**Output:**

1 . 43

3 . 00

3 . 50

6 . 41

APRIL10

[April 2010 Contest](#)01 Apr 2010  
15:00:00

12 days

387

[Home](#) » [Compete](#) » [April 2010 Contest](#) » Adding Fractions

## Adding Fractions Problem Code: ADDFRAC

You have discovered a new way to add fractions ! Now, the result of adding fractions  $a/b$  and  $c/d$  is  $(a + c)/(b + d)$ . Similarly the result of adding 3 fractions  $a/b$ ,  $c/d$ ,  $e/f$  is  $(a + c + e)/(b + d + f)$ , and so on. You are given a list of  $N$  fractions  $a_1/b_1$ ,  $a_2/b_2$ , ...,  $a_N/b_N$ . You

wonder what for each fraction  $i$  is the maximum fraction that you can obtain by adding together some continuous fractions in the list (possibly 1) starting at  $i$  ?

For example, let  $N = 4$  and the fractions be  $1/1$ ,  $4/3$ ,  $10/1$  and  $5/4$ . The maximum fraction you can obtain by starting at index 1 is  $3/1$  ( $1/1 + 4/3 + 10/1$ ). Similarly, the maximum fraction you can obtain by starting at index 2 is  $7/2$  ( $4/3 + 10/1$ ). By starting at index 3, the maximum sum you can obtain is  $10/1$  itself, and by starting at index 4, you can obtain sum  $5/4$ .

### **Input :**

The first line contains  $T$  the number of test cases.  $T$  test cases follow. The first line of each test case contains  $N$ . Each of the next  $N$  lines contains a fraction given in the form " $a_i/b_i$ ". A blank line separates two test cases.

### **Output :**

For each test case, output  $N$  lines. The  $i$ th line contains the maximum fraction you can obtain by adding continuous fractions in the list starting at index  $i$ . The numerator and denominator of any fraction you output should have gcd 1. Output a blank line after each test case.

### **Sample Input :**

2

4

$1/1$

$4/3$

$10/1$

$5/4$

5

$1/3$

$3/1$

$2/5$

5/6

6/5

**Sample Output :**

3/1

7/2

10/1

5/4

1/1

3/1

13/16

1/1

6/5

**Constraints :**

1 &lt;= T &lt;= 5

1 &lt;= n &lt;= 100000

1 &lt;= a\_k, b\_k &lt;= 100000

[Home](#) » [Compete](#) » [April 2010 Contest](#) » [Flood](#)**Flood** Problem Code: FLOOD

Each year in the rainy season, the LCD province in Byteland is flooded with water. The province has the form of a rectangular land divided into unit cells of M lines and N columns. Some cells are flooded and some cells are not. Each flooded cell has a water depth level which is a positive integer.

Because of flood, the land is divided into several islands, each island consists of the cells that can move around one another but cannot move to cells in different islands. The people can move between two non flooded cells if these cells have at least one common point.

To improve the transportation quality during the flooding season, the LCD's government decides to raise some flooded cells to become non flooded cells so that after the process, the people are able to travel between any two non flooded cells.

To make a flooded cell whose water depth level is  $D$  become non flooded, the workers need to put in  $D$  units of clay.

The government wants to achieve the goal using the fewest units of clay possible.

## **Input**

The first line contains  $t$ , the number of test cases (about 20), each test case has the following form.

The first line contains two numbers  $M$  and  $N$  ( $1 \leq M, N \leq 200$ ).

Each line in the next  $M$  lines contain  $N$  numbers describing the status of each cell in the land. If a cell is not flooded, the corresponding number is  $-1$  while if it is flooded the corresponding number is the cell's water depth level.

Each test case's input is separated by a blank line.

## **Output**

For each test case, output the result in the following format.

The first line contains  $T$ , the number of units of clay used.

The second line contains a number  $S$  which is the number of flooded cells needed to raise.

Each line in the next  $S$  lines contains two numbers representing the row and column index (1-based) of a raised cell.

Print a blank line after each test case's output.

## **Scoring**

For each test case, your program's score is equal to  $T/H$  in which  $H$  is the number of units of clays to raise all the flooded cell.

## **Example**

### **Input**

1

3 3

-1 10 -1

10 2 10

-1 10 -1

#### Output

2

1

2 2

This output will give score to be  $2 / (10 + 10 + 2 + 10 + 10) = 1/21$

[Home](#) » [Compete](#) » [April 2010 Contest](#) » [Trip](#)

## Trip Problem Code: TRIP

You are starting out on a long (really long) trip. On the way, there are N gas stations, the locations of which are given as a<sub>1</sub>,a<sub>2</sub>,...,a<sub>N</sub>. Initially you are located at the gas station at a<sub>1</sub>, and your destination is at location a<sub>N</sub>. Your car can only store enough fuel to travel atmost M units without refilling. You can stop at any station and refill the car by any amount. Now you wish to plan your trip such that the number of intermediate stops needed to reach the destination is minimum, and also how many ways are there to plan your trip accordingly.

#### Input :

The first line two space seperated integers N and M. N lines follow, and the ith line has the value a<sub>i</sub> ( $0 \leq a_i \leq 1000000000$ ). The input will be such that a solution will always exist.

#### Output :

Output two space seperated integers : The least number of stops, and the number of ways to plan the trip which uses the least number of stops. Output this value modulo 1000000007.

#### Sample Input :

6 3

0

1

3

4

7

10

### Sample Output :

3 2

### Constraints :

$2 \leq N \leq 1000000$

$1 \leq M \leq 1000$

$a_1 < a_2 < \dots < a_N$

[Home](#) » [Compete](#) » [April 2010 Contest](#) » Travelling Salesman Again !

## Travelling Salesman Again ! Problem Code: TSPAGAIN

There are  $N$  cities numbered from  $0..N-1$ . A salesman is located at city 0. He wishes to visit all cities exactly once and return back to city 0. There are  $K$  toll booths. Each toll booth has a certain range of functioning. The parameters for toll  $k$  are given as  $x_k$  and  $y_k$ . If the salesman travels from city  $i$  to  $j$ , he has to pay 1 dollar toll fee to each toll  $p$  having  $x_p \geq i$  and  $y_p \leq j$ . Calculate the cheapest way for the salesman to complete his tour.

### Input :

The first line contains  $T$  the number of test cases.  $T$  test cases follow. The first line of each test case contains two space separated integers  $N$  and  $K$ . Each of the next  $K$  lines contains 2 integers, the  $i$ th line containing  $x_i$  and  $y_i$  ( $0 \leq x_i, y_i < N$ ). A blank line separates two test cases.

### Output :

Output  $T$  lines, one for each test case, containing the required answer.

**Sample Input :**

2

3 2

2 0

0 2

3 4

1 0

2 1

0 1

1 2

**Sample Output :**

3

6

**Constraints :**

1 &lt;= T &lt;= 50

2 &lt;= n &lt;= 1000

1 &lt;= K &lt;= 10000

[Home](#) » [Compete](#) » [April 2010 Contest](#) » The Reversed World

# The Reversed World

**Problem Code: REVERSED**

Given three numbers A, B and C all having N digits, little Johnny needs to write a program to permute the digits of the number C so that the new number is greater than both A and B.

This problem would be so easy if Johnny's teacher didn't require that Johnny's program must be correct in both the real world and *the reversed world*. What happens once the world is reversed? One thing Johnny knows for sure is that all the numbers are also reversed. The second thing, the "greater than" relation becomes the "smaller than" relation.

For the sake of clarity, we will use  $a^R$  to denote the number  $a$  in the reversed world (for instance,  $113^R = 311$ ).

Thus Johnny task is now permute the digits of  $C$  so that the new number is greater than both  $A$  and  $B$ . In additional, the reverse of the new number should be smaller than both  $A^R$  and  $B^R$ . If there are multiple possible solutions, Johnny needs to find the smallest one.

As usual, please help Little Johnny with his task!

## Input

The first line contains a number  $T$  (about 10) which is the number of the test cases. Each test case has the following form.

The first line contains  $N$  ( $1 \leq N \leq 1000000$ ).

The second line contains the number  $A$ .

The third line contains the number  $B$ .

The fourth line contains the number  $C$ .

Each test case's input is separated by a blank line.

## Output

For each test case, print the solution in a single line. If there is no solution, print the number  $-1$ .

## Example

**Input:**

2

2

12

23

32

3

124

324

335

**Output:**

-1

353

[Home](#) » [Compete](#) » [April 2010 Contest](#) » Lighting the shop

## Lighting the shop Problem Code: LIGHTING

The exciting World Cup 2010's summer is coming! And Johnny's family shop is busy preparing a colorful lighting board for the summer.

The board has a rectangular shape of size  $M \times N$ . There are certain positions on the board which Johnny wants to put in light bulbs.

Johnny wants to decorate the board by colorful light bulbs. To make the board look attractive, he wants the bulbs in the same column or row to have different colors! On the other hand, to preserve the harmony look, Johnny doesn't want the bulbs to have many different colors.

Write a program to help Johnny decorate the boards in such a way that the number of color used is minimum.

### Input

The first line contains a number  $t$  (about 15), the number of test cases. Each test case has the following form.

The first line contains two numbers  $m, n$  ( $1 \leq m, n \leq 700$ ). Each line in the next  $m$  lines contains  $n$  characters '0' or '1' representing the board, in which '1' denotes a place which Johnny wants to put in a light bulb.

Each test case's input is separated by a blank line.

## Output

For each test case, print the output in the following format.

The first line contains a number  $p$  which is the minimum number of used colors.

The  $i$ th line in the next  $m$  lines contain  $n$  integers, in which the  $j$ th number is the color index of the light bulb in the corresponding place, or 0 if the place does not need a light bulb. The colors are numbered from 1 to  $p$ . Your program can print any correct solution.

Print a blank line after each test case's output.

## Example

### Input

1

4 4

1001

1101

1011

1001

### Output

4

1 0 0 2

2 1 0 3

3 0 1 4

4 0 0 1

MAY10	<a href="#">May 2010 Contest</a>	01 May 2010 15:00:00	11 days	270
-------	----------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Rendezvous

## Rendezvous Problem Code: TR1

There are  $N$  cities in Byteland numbered 1 to  $N$  with city 1 as the capital. The road network of Byteland has a tree structure. All the roads are bi-directional. Alice and Bob are secret agents working for the Bytelandian Security Agency (BSA). Each secret agent is assigned a random city each month, where he/she must be stationed for the complete month. At the end of each month, they come to the capital to submit their reports to the head of BSA.

They always take the shortest path from their city to the capital. If Alice is assigned city A and Bob is assigned city B then they meet at a city C which is common to both their routes to the capital and then travel together from C to the capital.

Alice and Bob wish to spend maximum time together in any trip. So for any pair of assigned cities (A,B) they meet at a city C such that C is the farthest city from the capital and appears in the shortest path from capital to A and capital to B. Since this happens each month they compute this for each pair of assigned cities (A,B) and store it in a matrix  $M$ , where  $M[A][B] = C$ , the city where they meet.

The importance of a city C(according to Alice and Bob),  $Im(C)$  is defined as the number of times C appears in their matrix  $M$ . Alice and Bob are interested in finding the importance of each city. Since this output can be large, output the sum  $S$  defined as  $S = (\text{sum } i=1 \text{ to } N) i * Im(i)$  modulo 1000000007.

### Input

First line of input contains an integer  $t$  ( $t \leq 25$ ), the number of test cases. Each test case starts with an integer  $N$  ( $1 \leq N \leq 10000$ ), the number of cities

The next  $N-1$  lines contain two space separated integers  $u$   $v$  ( $1 \leq u, v \leq N$ ) denoting a road from  $u$  to  $v$ .

### Output

For each test case output the required sum  $S$

### Example

**Input:**

5

1 2

1 3

2 4

2 5

3

1 2

1 3

1

**Output:**

41

12

1

**Explanation**

For the first test case, the matrix M is:

1 1 1 1 1

1 2 1 2 2

1 1 3 1 1

1 2 1 4 2

1 2 1 2 5

and the corresponding importance array is: 15 7 1 1 1

so the required sum is  $1*15 + 2*7 + 3*1 + 4*1 + 5*1 = 41$

For the second test case, the matrix M is:

1 1 1

1 2 1

1 1 3

and so the Importance array is: 7 1 1

So the required sum is  $1*7 + 2*1 + 3*1 = 12$

For the third test case, there is only one city, so the Matrix M just has one entry 1, so  $S = 1$

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Nice Quadrangles

## Nice Quadrangles Problem Code: NICEQUAD

There are  $n$  points with integer coordinates. We can form different quadrangles out of them by taking four different points and connecting them with lines. Let's call a quadrangle ABCD nice if and only if:

- $Ax > 0$  and  $Ay > 0$ ;
- $Bx > 0$  and  $By < 0$ ;
- $Cx < 0$  and  $Cy < 0$ ;
- $Dx < 0$  and  $Dy > 0$ ;
- ABCD has an integer area.

Your task is to count all different nice quadrangles that can be formed on the given points.

### Input

The first line of input file contains number  $t$  – the number of test cases. Then the description of each test case follows. The first line of each test case contains number  $n$  – the number of points. Then  $n$  lines follow each consisting of two integers  $x, y$  – the coordinates of a point. No two points in the same test case coincide.

### Constraints

$1 \leq t \leq 10$   
 $1 \leq n \leq 30000$   
 $-30000 \leq x, y \leq 30000$

## Output

For each test case print the number of nice quadrangles that can be formed using given points.

## Example

**Input:**

```

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

```

**Output:**

```
2
```

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Summing Slopes

## Summing Slopes Problem Code: SUMSLOPE

A digit in a number  $N$  is a minima if it is lesser than both the digits adjacent to it. Similarly, a digit is a maxima if it is greater than both the digits adjacent to it. The slope of  $N$  is the number of digits in  $N$  (leaving out the first and the last digit) which are either a minima or a maxima. Given  $A$  and  $B$ , count the sum of the slopes of all numbers between  $A$  and  $B$ .

Input :

The first line contains the number of test cases T. Each of the next T lines contains two integers A and B.

Output :

Output T lines one for each test case, containing the required sum for the corresponding test case.

Sample Input :

3

101 101

1 100

100 150

Sample Output :

1

0

19

Constraints :

1 <= T <= 50000

1 <= A <= B <= 1000000000000000 (10^15)

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Lights Out

## Lights Out Problem Code: DORADM01

Johnny is playing Lights Out, an electronic game consisting of an NxN grid of lights. Initially, each light may be on or off. Pressing a light will toggle it and the lights non-diagonally adjacent to it.

The object of the game is to try and get all the lights turned off. It is not always possible, and what's more, Johnny's game is faulty. Pressing some lights has no effect. (They can still be turned on and off if they are next to working lights.)

Help Johnny write a program that checks whether a puzzle can be solved or not, given the initial description of the grid and the positions of the faulty lights.

## Input

The first line contains a number  $t$  (about 10), the number of test cases. Each test case has the following form:

The first line contains two numbers  $N$  and  $K$  ( $2 \leq N \leq 200$ ,  $0 \leq K \leq N$ ). Each line in the next  $N$  lines contains  $N$  characters '0' or '1' representing the initial state of the grid. '1' denotes a light that is on and '0' a light that is off.

Each line in the next  $K$  lines contains 2 integers,  $i$  and  $j$  ( $1 \leq i, j \leq N$ ), representing the row and column of a faulty light. Rows are numbered from the top and columns from the left.

## Output

For each test case, print YES if there is a solution, otherwise print NO.

## Example

Input:

2

3 1

010

111

010

2 2

4 2

1110

1110

1001

1101

2 2

4 3

**Output:**

NO

YES

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Generalized Spanning Trees

## Generalized Spanning Trees Problem Code: GST

A spanning tree of an undirected, connected graph  $(V, E)$  with vertex set  $V$  and edge set  $E$  is a subset of edges  $F$  such that the graph  $(V, F)$  is connected and has no cycles. After a few moments thought, one realizes that an equivalent way to say this is to say:

- 1)  $|F| = |V| - 1$
- 2) For any non-empty subset of nodes  $S$ , the number of edges with **both** endpoints in  $S$  is does not exceed  $|S|-1$ .

There is a fairly simple algorithm that can determine if a set of edges  $F$  is indeed a spanning tree of a graph. Simply check that  $|F| = |V| - 1$  and that the graph  $(V, F)$  is connected using your favorite connectivity algorithm.

As in all topics in mathematics, the next step is to generalize what is known. So, consider the following generalization of spanning trees. Say a generalized spanning tree (GST) of a graph  $(V, E)$  is a function  $f$  from the set of edges  $E = \{e_1, e_2, \dots, e_M\}$  to the real interval  $[0, 1]$  such that the following conditions hold:

- 1)  $f(e_1) + f(e_2) + \dots + f(e_M) = |V| - 1$
- 2) For any non-empty subset of nodes  $S$ , the sum of the  $f(e)$  values for edges  $e$  with **both** endpoints in  $S$  does not exceed  $|S|-1$ .

One can see that this generalizes the notion of a spanning tree since spanning trees (in the classical sense) correspond to GSTs by setting  $f(e) = 1$  if  $e$  is in the spanning tree and  $f(e) = 0$  if  $e$  is not in the spanning tree.

Your task is to determine if a given function is a GST of a given graph.

**Input:**

The first line contains a single integer  $T$  (about 20) indicating the number of test cases to follow.

Each test case begins with two integers  $N$  and  $M$  where  $N$  is the number of nodes and  $M$  is the number of edges. The following  $M$  lines describe the edges, one per line. Each line contains three values  $u, v, f$  where  $1 \leq u, v \leq n$  are the endpoints of the edge and  $f$  is a rational number between 0 and 1 (inclusive). This rational number is the value of the edge in the proposed GST.

The rational numbers are always presented as  $n/d$  (even if  $d=1$ ) where both  $n$  and  $d$  are non-negative and have no common factors.

The input graphs have no parallel edges or loops and are, of course, undirected. Consecutive cases will be separated by a blank line (including a blank line preceding the first case).

Bounds:  $2 \leq N \leq 50$ ,  $1 \leq M \leq 100$ , and no denominator of any fraction in the input exceeds 30.

## **Output:**

If the function is a GST, you are to simply print a single line saying "GST".

If the first constraint (the sum of the values of all edges is  $N-1$ ) is violated, then you should print a single line displaying just the number 1.

Otherwise, if constraint 1) holds but constraint 2) is violated for some non-empty set  $S$ , you should print three lines. The first line simply contains the integer 2. The second contains the size of a non-empty subset of the nodes whose corresponding constraint is violated. The third line lists the nodes (in any order) that are part of such a set with a violated constraint (the number of nodes is as reported in the second line).

If there are multiple non-empty sets whose constraints are violated, then any will do. Print a blank line after the output for each case.

## **Example:**

### **Input:**

3

3 3

1 2 2/3

2 3 2/3

3 1 2/3

3 3

1 2 1/1

2 3 1/1

3 1 1/1

4 6

1 2 1/5

1 3 1/5

1 4 1/5

2 3 4/5

2 4 4/5

3 4 4/5

## **Output:**

GST

1

2

3

2 3 4

[Home](#) » [Compete](#) » [May 2010 Contest](#) » Rotation Puzzle

# Rotation Puzzle Problem Code: STORO

Recently, with the release of the ultimate computing device bytePad, a platform game with the simple name "Rotation Puzzle" immediately became a phenomenon among Bytelandians. The game is extremely simple, yet quite additive. Here's the rule:

Given a  $M \times N$  rectangular grid in which each cell contains a unique number from 1 to  $M \times N$ . In each step, the player can pick any  $2 \times 2$  subgrid and perform a rotation (whether clockwise or counterclockwise).

The task is to transform from the initial grid to the final configuration, using as few steps as possible.

The final configuration is the configuration

1	2	...	n
$n+1$	$n+2$	...	$2n$
...	...	...	...
$(m-1)n+1$	$(m-1)n+2$	...	$mn$

You may have guessed why this game is addictive: it requires a tremendous visualization skill!!

## Input

The first line contains a number  $T$  (about 5000), which is the number of test cases. Each test case has the following form.

The first line contains two numbers  $M$  and  $N$  ( $2 \leq M, N \leq 33$ ).

The next  $M$  lines contains the description of the grid.

Each test case's input is separated by a blank line.

It is guaranteed that each input data has a solution.

## Output

For each test case, output the result in the following format.

The first line contains a number  $K$ , the number of steps you need to solve the puzzle.  $K$  must not exceed 10000.

Each line of the next K lines contain three numbers c, i, j (c=0 or c=1, 1<=i < M, 1 <= J < N). (i,j) is the top-left coordinate of the 2x2 square that is need to be rotated. c=1 if the rotation if clockwise and c=0 if the rotation is counter-clockwise.

Prints a blank line after each test case's output.

## Scoring

Given m and n, we pick a random positive integer  $K^*$  and starting from the final configuration, we perform a random operation  $K^*$  times to generate the test case. For each test case, your score is  $K^*/K$ . Higher score is better.

## Example

**Input:**

2

2 3

1 5 2

4 6 3

2 3

5 6 2

1 4 3

**Output:**

1

0 1 2

2

1 1 1

0 1 2

JUNE10	<a href="#">June 2010 Contest</a>	01 Jun 2010 15:00:00	10 days	337
--------	-----------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [June 2010 Contest](#) » Squirrel and chestnut

## Squirrel and chestnut Problem Code: SQUIRREL

There are  $n$  squirrel(s) waiting below the feet of  $m$  chestnut tree(s). The first chestnut of the  $i$ -th tree will fall right after  $T_i$  second(s), and one more every  $P_i$  second(s) after that. The “big mama” of squirrels wants them to bring their nest no less than  $k$  chestnuts to avoid the big storm coming, as fast as possible! So they are discussing to wait below which trees to take enough chestnuts in the shortest time. Time to move to the positions is zero, and the squirrels move nowhere after that.

### Request

Calculate the shortest time (how many seconds more) the squirrels can take enough chestnuts.

### Input

- The first line contains an integer  $t$ , the number of test cases, for each:
    - The first line contains the integers  $m, n, k$ , respectively.
    - The second line contains the integers  $T_i$  ( $i=1..m$ ), respectively.
    - The third line contains the integers  $P_i$  ( $i=1..m$ ), respectively.
- (Each integer on a same line is separated by at least one space character, there is no added line between test cases)***

### Output

For each test cases, output in a single line an integer which is the shortest time calculated.

### Example

#### Input:

2

3 2 5

5 1 2

1 2 1

3 2 5

5 1 2

1 1 1

**Output:**

4

3

\* *Explain (case #1): 2 squirrels waiting below the trees 2 and 3.*

## Limitations

- $0 < t \leq 20$
- $0 < m, n \leq 10,000$
- $0 < k \leq 10^7$
- $0 < T_i, P \leq 100$

[Home](#) » [Compete](#) » [June 2010 Contest](#) » Robot Game

## Robot Game Problem Code: TR2

Byteland is a kingdom of islands. Alice has  $N$  maps of the different islands constituting Byteland. Map of each island consists of cities, connected by roads. Being Byteland, the road layout of each island has a binary tree structure. Each island has a capital, which is the root of the binary tree of its map.

Alice is bored and plays the following game:

She chooses  $k$  out of these  $N$  maps and places a robot on the capital city of each map. Every robot is controlled by a common control string. If the present character of the control string is L(R), then all the robots move to the left(right) subtree on their map. All the robots follow the same control string and move on their respective maps.

A control string is valid with respect to a given map if it corresponds to a valid path on that map. A control string is considered valid for a given set of  $k$  maps if it is valid for each of the  $k$  chosen maps.

For example in a complete binary tree of 3 nodes, control strings "L" and "R" are valid and "LL" is an example of an invalid control string.

Alice is interested in knowing the longest control string which is valid for at least  $k$  maps, for each value of  $k$ . Can you find this for her?

## Input

First line of input contains an integer  $t$  ( $t \leq 5$ ), the number of test cases. Each test case starts with an integer  $M$  ( $1 \leq M \leq 100$ ), the number of maps. Then  $M$  maps description follow.

The  $i$ th map description starts with an integer  $N$  ( $2 \leq N \leq 1000$ ), the number of cities on the island. Each of the following  $N-1$  lines contain two space separated integers  $u$   $c$   $v$  ( $1 \leq u, v \leq N$ ,  $c=L$  or  $R$ ) which means a road between city  $u$  and city  $v$  on the  $i$ th island and if  $c$  is  $L$ ( $R$ ) then  $v$  is in the left(right) subtree of  $u$ . The capital city of each island is the city with number 1.

## Output

For each test case output  $M$  space separated integers, where the  $i$ th integer denotes the length of the longest valid control string she can obtain assuming she can choose  $i$  maps.

## Example

Input:

2

3

5

1 L 2

1 R 3

3 L 4

3 R 5

4

1 R 2

1 L 3

2 L 4

3

1 L 2

1 R 3

2

5

1 L 2

2 L 3

3 L 4

4 L 5

6

1 L 2

2 R 3

3 L 4

4 R 5

5 L 6

**Output:**

2 2 1

5 1

[Home](#) » [Compete](#) » [June 2010 Contest](#) » Pricing Tollbooths

## Pricing Tollbooths Problem Code: TOLLS

Your company has recently built its own highway from which they hope to generate some revenue. The highway has no branches or intersections, it is a simple line segment. Your company also has access to simple data from some potential customers that describe their start and endpoint locations and their budget.

Tollbooths are also located on various locations on the highway. The total toll a customer pays is the sum of all tolls on the tollbooths that lie between their start and end locations. If a customer cannot afford the total toll they must pay, then they simply don't make the trip and end up paying nothing.

No customer wants to start or end their destination at the precise location of a tollbooth so we can represent the problem as follows. We have a graph that is a simple path with  $N$  nodes. Each node represents a potential start or end location of a customer and there is a single tollbooth located on the middle of each edge. So the total toll a customer pays is the sum of the tolls on the edges they cross to reach their endpoint.

Your task is to set the tolls on each of the tollbooths to generate some revenue for your company.

### **Input:**

The first line contains an integer  $T$  (about 40) indicating the number of test cases.

Each test case begins with two positive integers  $N$  and  $M$  (both at most 100) where  $N$  is the number of nodes in the graph and  $M$  is the number of potential customers.  $M$  lines follow where the  $i$ 'th line contains the information of the  $i$ 'th client.

Each such line consists of three integers  $S$ ,  $E$ ,  $B$  where  $1 \leq S, E \leq N$  are, respectively, the start and end locations of the customer and  $1 \leq B \leq 1,000,000$  is the customer's budget.

A blank line separates test cases (as well as the first line containing  $T$  and the first test case).

### **Output:**

For each test case, you are to output a single line consisting of  $N-1$  integers, each of which is between 0 and 1,000,000. The  $i$ 'th integer ( $1 \leq i \leq N-1$ ) denotes the toll placed on the edge connecting nodes  $i$  and  $i+1$ .

### **Scoring:**

From the output, we will calculate the total revenue obtained from all customers who can afford the total toll on their route according to the prices you set. The score for each test case is then calculated as  $R/T$  where  $R$  is the revenue obtained and  $T$  is the total budget of all customers. The score for the entire set of test cases is just the sum of the scores of each individual test case.

### **Example: Input:**

2

3 3

1 2 10

2 3 10

1 3 10

4 3

1 2 10

2 3 10

1 4 15

### Output:

5 5

10 10 0

In the first test case, the first two customers only pay a total toll of 5 and the last customer pays 10 so the total revenue is 20. The score for this case is 20/30.

In the second case, the first two clients pay their full budget but the last client cannot afford the desired route. The total revenue is then 20 and the score for this case is 20/35.

[Home](#) » [Compete](#) » [June 2010 Contest](#) » Prime Pattern

## Prime Pattern

Problem Code: PRIMPATT

There is a very large field, colored white and divided into squares. There is a coordinate system attached to the field: y-axis goes from south to north and x-axis goes from west to east. Sides of the squares are parallel to the axes. There is a robot in the (0, 0) square. Robot starts to move in spiral as depicted. First it moves one step east and one step north. Then two steps west and two steps south. Then three steps east and three steps north, four steps west and four steps south and so on. It moves to a new square with each step. As it moves it counts squares it passes and, if the number of the square is the prime number, then the robot fills this square with black color. The (0, 0) square has the number 0. Given the coordinates of a square you are to calculate the distance from this square to the nearest to it black square. For two squares with coordinates (x1, y1) and (x2, y2) the distance between those squares is  $|x1-x2|+|y1-y2|$ .

### Input

Input file consists of a set of tests. The first line of the file is number T – the number of tests ( $T \leq 500$ ). Following T lines contains two integers each separated with a space: x and y – the coordinates of the square ( $-2000001 < x < 2000001, -2000001 < y < 2000001$ ).

## Output

For each coordinate pair in the input file you are to output the distance between this square and the nearest to it black square.

## Example

### Input:

8

0 0

1 0

1 1

0 1

3 3

-3 -3

-1 2

0 -3

### Output:

1

1

0

0

1

1

2

2

## Graph Counting Problem Code: GRAPHCT

First for some definitions :

Let  $G = (V, E)$  be an undirected graph containing an edge  $e = (u, v)$  with  $u \neq v$ . Let  $f$  be a function which maps every vertex in  $V \setminus \{u, v\}$  to itself, and otherwise, maps it to a new vertex  $w$ . The contraction of  $e$  results in a new graph  $G' = (V', E')$ , where  $V' = (V \setminus \{u, v\}) \cup \{w\}$ ,  $E' = E \setminus \{e\}$ , and for every  $x \in V$ ,  $x' = f(x) \in V'$  is incident to an edge  $e' \in E'$  if and only if, the corresponding edge,  $e \in E$  is incident to  $x$  in  $G$ .

An undirected graph  $H$  is called a minor of the graph  $G$  if  $H$  is isomorphic to a graph that can be obtained by zero or more edge contractions on a subgraph of  $G$ .

A graph is connected if there exists a path between any two vertices. A biconnected graph is one which remains connected even after the removal of any one vertex and all edges incident to it.

A simple graph is one which does not have more than one edge between any pair of vertex, nor does it have an edge from a vertex to itself.

You need to count how many simple biconnected graphs having  $n$  vertices and  $m$  edges exist such that they do not have a cycle of length 5 as a minor. Two graphs are considered distinct if there exist vertices having labels  $i$  and  $j$  which are adjacent in the first graph, but not in the second graph.

Input :

The first line contains the number of test cases  $T$ . Each of the next lines contains two integers  $n$  and  $m$ .

Output :

Output  $T$  lines, one corresponding to each test case. For a test case, output the number of graphs as described in the question. Output the answer modulo 1000000007.

Sample Input :

5

1 0

3 2

3 3

4 4

5 10

Sample Output :

1

0

1

3

0

Constraints :

1 <= T <= 2000

1 <= n <= 100

0 <= m <= 10000

[Home](#) » [Compete](#) » [June 2010 Contest](#) » Tidying Posters

## Tidying Posters Problem Code: POSTERS

A public wall at your university is littered with posters advertising various events. A new policy has just been enacted that states no two posters on the wall may overlap. You have been asked to remove some posters from the wall so the remaining posters do not obscure each other. To keep the students as happy as possible, you should remove the minimum number of posters to achieve this goal. You may not remove a poster and place it in another position; all posters you leave on the wall must be in their original position.

When you examined the wall, you noticed something very nice. Every poster was placed on the wall by pinning the four corners. This was done in a very courteous manner since each pin goes only through the poster it is holding.

Time to get to work! Oh, the reason you were hired is that you are very good at taking down a single poster without disturbing the rest, even if that poster is obscured by many others.

### Input

The first line consists of a single integer  $T$  indicating the number of test cases (about 25).

Each test case consists begins with a single integer  $n$  indicating the number of posters. The next  $n$  lines consist of 4 integers  $x_0, x_1, y_0$ , and  $y_1$  satisfying  $x_0 < x_1$  and  $y_0 < y_1$ . This means the poster covers all points  $(x,y)$  satisfying  $x_0 \leq x \leq x_1$  and  $y_0 \leq y \leq y_1$ .

As stated before hand, no corner of any poster will intersect any other poster anywhere. That is, if  $(x,y)$  is a corner point of one poster and another poster is described by  $x_0, x_1, y_0$ , and  $y_1$ , then we do not have  $x_0 \leq x \leq x_1$  and  $y_0 \leq y \leq y_1$ .

Bounds:  $1 \leq n \leq 100$  and each integer in a poster description fits in a signed 32 bit integer.

## Output

The output for each test case is a single line containing a single integer that is the maximum number of posters that can be left on the wall such that no two posters share a common point on the wall.

## Example

Input:

2

3

0 5 1 2

1 2 0 3

3 4 0 3

3

-3 3 -1 1

-2 2 -2 2

-1 1 -3 3

**Output:**

2

1

JULY10	<a href="#">July 2010</a>	01 Jul 2010 15:00:00	10 days	464
--------	---------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [July 2010](#) » Maximal crosses

## Maximal crosses Problem Code: MAXCROSS

On the matrix A sized  $n \times n$ , some cells were marked by crosses (X). For each cell  $(i, j)$  (with  $i$  – the row index,  $j$  – the column index), we define  $B(i, j)$  as the maximal number of continuous crosses “going across” the cell  $(i, j)$  in the same horizontal, vertical or diagonal;  $B(i, j)=0$  if  $A(i, j)$  is empty ('.'). [ Empty cells are marked by '.' ].

**Clarification:**

Continuous crosses going across cell  $(i, j)$  in horizontal direction =  $x_1 + x_2 - 1$   
where  $x_1$  = highest possible  $x$  such that

$j + x - 1 \leq n$  and  $A(i, j \dots j + x - 1)$  are all 'X'

and  $x_2$  = highest possible  $x$  such that

$j - x + 1 > 0$  and  $A(i, j - x + 1 \dots j)$  are all 'X'

Similarly we can extend the definition for vertical and diagonal directions.

### Task

Given matrix A as input, calculate matrix B.

### Input

- The first line contains the integer  $n$ .
- Then follow  $n$  lines, each containing  $n$  characters. The  $j$ -th character of the  $i$ -th line represents the cell  $(i, j)$  of the matrix A, with  $A(i, j)='X'$  if it contains a cross, or  $A(i, j)='.'$  if it is empty.

### Output

Output  $n$  lines, each contains  $n$  integers, the  $j$ -th integer of the  $i$ -th line shows the value of  $B(i, j)$ .

**(Each integer on a same line must be separated by exactly one space character)**

### Example

**Input:**

10

..X....XX.

XX.X..XX.X

....XX..X

.XXX..X.X.

....X..XX

....X....X

X.X....XX.

.X...X.X.X

X.X..X....

..XXXXX.XX

**Output:**

0 0 2 0 0 0 0 3 3 0

2 2 0 2 0 0 3 3 0 2

0 0 0 0 0 3 3 0 0 2

0 3 3 3 0 0 3 0 2 0

0 0 0 0 0 3 0 0 2 2

0 0 0 0 3 0 0 0 0 3

4 0 3 0 0 0 0 2 3 0

0 4 0 0 0 3 0 3 0 2

3 0 4 0 0 3 0 0 0 0

0 0 5 5 5 5 0 2 2

[Home](#) » [Compete](#) » [July 2010](#) » Closest Ranking

## Closest Ranking

Problem Code: VOTING

You have just begun his job with Educated Guessing Pollsters. An election is only weeks away and the company would like to get a general idea of how the voters feel about the candidates. They asked a (probably not) random subset of N voters to submit a ranking of

the M candidates. Say the candidates are numbered from 1 to M. A ranking is then a permutation of the set  $\{1, 2, \dots, M\}$  listed as:

$c_1 \ c_2 \ \dots \ c_M$

The list is such that the voter prefers candidate  $c_i$  to candidate  $c_j$  if  $i > j$ . Your task is to determine a ranking  $R = r_1, r_2, \dots, r_M$  such that the total number of disagreements with all voters who were polled is minimized.

More precisely, for a voter  $V$  we say the distance  $d(V, R)$  between voter  $V$  and ranking  $R$  is precisely the number of pairs  $(i, j)$  with  $1 \leq i < j \leq M$  such that  $V$  actually prefers candidate  $r_i$  over candidate  $r_j$ . Say there are  $N$  voters labeled  $V_1, V_2, \dots, V_n$ . The task is then to find a ranking  $R$  such that the sum

$$d(V_1, R) + d(V_2, R) + \dots + d(V_n, R)$$

is minimized.

### **Input:**

The first line of input consists of a single integer  $T \leq 25$  indicating the number of test cases to follow. Each test case begins with two integers  $N$  and  $M$  on a single line where  $N$  represents the number of voters and  $M$  represents the number of candidates. The following  $N$  lines contain the rankings of the voters, one per line.

Each ranking is given as a permutation  $c_1, c_2, \dots, c_M$  of the numbers  $1, 2, \dots, M$  where consecutive numbers are separated by a space. In this ranking,  $c_i$  is preferred over  $c_j$  by precisely when  $i > j$ .

A blank line appears between test cases (including a blank line just before the first test case).

The bounds are  $1 \leq N \leq 1000$  and  $1 \leq M \leq 15$ .

### **Output:**

Each test case has a single line of output of the following form:

$D: r_1 \ r_2 \ \dots \ r_M$

Where  $D$  is the sum of the distances between the optimum ranking and the input rankings and  $r_1, r_2, \dots, r_M$  is a ranking of the same form as the input rankings that achieves this minimum distance. If there are multiple such orderings, then output the one that minimizes  $r_M$ . Among those that minimize  $r_M$ , you should output the one that minimizes  $r_{\{M-1\}}$  and so on.

### **Example:**

**Input:**

2

2 3

1 2 3

1 2 3

4 3

1 2 3

2 1 3

1 2 3

3 2 1

**Output:**

0: 1 2 3

4: 2 1 3

(note that in the second case that the ranking 1 2 3 also has total distance 4, but 2 1 3 is output based on the rules outlined in the output specification)

[Home](#) » [Compete](#) » [July 2010](#) » Block Tower

## Block Tower Problem Code: BLOCKS

Dave has several rectangular blocks with which he wishes to build a tower. He may only place a block on another if its base fits within the other's base with the edges parallel. So he could place a block with a base of 1x9 on top of a block with a base of 2x9, but not on top of a block with a base of 8x8. Help Dave make the tower as tall as possible. Blocks may be rotated by any multiple of 90 degrees about any axis. Each block may only be used once, and you must use at least 3 blocks.

**Input:**

Input begins with an integer  $T$  (about 500), the number of test cases. Each test case begins with an integer  $N$  (chosen uniformly between 10 and 200, inclusive), the number of blocks.  $N$  lines follow, each containing 3 integers that give the dimensions of a block. All dimensions are chosen uniformly between 1 and 100, inclusive. A blank line separates each case.

### **Output:**

For each case, first output an integer  $P \geq 3$ , indicating the number of blocks you wish to use. Follow with  $P$  lines, each containing the dimensions of a block, with the height of the block listed first (the order of the other 2 dimensions does not matter). Output the blocks in order from the top of the tower to bottom of the tower.

### **Scoring**

Your score for each test case is the height of your tower divided by  $N$ . Your total score is the average of your scores on the individual test cases.

### **Sample input:**

1

10

7 2 10

8 8 8

7 1 1

2 7 9

6 8 1

6 6 5

3 2 5

10 3 9

10 10 8

4 4 1

## Sample output:

5

4 1 4

1 1 7

10 2 7

2 7 9

8 10 10

Score:  $(4+1+10+2+8)/10 = 2.5$

[Home](#) » [Compete](#) » [July 2010](#) » Happy Days

## Happy Days Problem Code: HAPPY

Johnny has a pool in his garden. There are several islands in the pool. Some islands are connected by bridges. Any bridge can be removed. Every day Johnny removes some bridges so that there is only one way from any island to any other. In the evening he returns removed bridges to their places. Also he has some favorite bridges which he never removes. Johnny will be happy if he is able to make a configuration of bridges on the given day which he has never made before. You have to count the amount of days he will be happy. Of course, if the favorite bridges themselves don't satisfy the happiness condition Johnny will not be happy for even single day.

### Input

The first line of input file contains number  $t$  – the number of test cases. Then the description of each test case follows. The first line of each test case contains number  $n$  – the number of islands. Islands are numbered with integers from 1 to  $n$ . Then  $n$  lines follow each containing  $n$  characters defining the connectivity matrix of those islands. Character in column  $x$  of line  $y$  will be '1' if the islands with numbers  $x$  and  $y$  are connected and '0' otherwise. The next line is number  $p$  – the number of favorite bridges. The next  $p$  lines contain the pairs of islands that are connected by favorite bridges.

### Output

For each test case print the number of days Johnny will be happy in this situation.

### Constraints

$1 \leq t \leq 5$   
 $2 \leq n \leq 30$   
 $1 \leq p \leq \min(6, n-1)$

## Example

Input:

1  
 4  
 0111  
 1011  
 1101  
 1110  
 2  
 1 2  
 3 4

Output:

4

[Home](#) » [Compete](#) » [July 2010](#) » Rearranging Digits

## Rearranging Digits Problem Code: ARRANGE2

A number X is called Permut number if X and  $2X$  ( both without leading zeroes ) have same number of digits and are permutations of each other. Given A and B, find the number of Permut numbers  $\geq A$  and  $\leq B$ .

Note : X is called permutation of Y if every digit ( 0 - 9 ) occurs same number of times in both the numbers ( maybe at different positions ).

Input :

The first line contains the number of test cases T. T lines follow, containing two integers A and B.

Output :

Output T lines, one for each test case containing the desired answer for the corresponding test case.

Sample Input :

2

1 100

499875921 499875921

Sample Output :

0

1

Constraints :

$1 \leq T \leq 10000$

$1 \leq A \leq B \leq 10000000000 \ (10^{10})$

[Home](#) » [Compete](#) » [July 2010](#) » Adding Least Common Multiples

## Adding Least Common Multiples Problem Code: LCM

Given A and B, compute the sum of  $\text{lcm}(a, b)$  over all pairs of positive integers a and b such that:

- (1)  $a \leq A$  and  $b \leq B$ .
- (2) There is no integer  $n > 1$  such that  $n^2$  divides both a and b.

Give your answer modulo  $2^{30}$ .

### Input

The first line contains the number of test cases,  $t$  (about 200). Each of the next  $t$  lines contains two space-separated integers  $A$  and  $B$  ( $1 \leq A, B \leq 4000000$ ).

## Output

Print the answer to each test case on a separate line.

## Example

Input:

4

2 4

3 3

6 5

8 3

Output:

24

28

233

178

COOK01

[July Cook Off](#)

24 Jul 2010  
21:30:00

2 hours 30 minutes

215

[Home](#) » [Compete](#) » [July Cook Off](#) » The Black and White Knights

# The Black and White Knights

Problem Code: **BWKNIGHT**

How many ways are there to place a black and a white knight on an  $N * M$  chessboard such that they do not attack each other? The knights have to be placed on different squares. A knight can move two squares horizontally and one square vertically, or two squares

vertically and one square horizontally. The knights attack each other if one can reach the other in one move.

Input :

The first line contains the number of test cases T. Each of the next T lines contains two integers N and M.

Output :

Output T lines, one for each test case, each containing the required answer for the corresponding test case.

Sample Input :

3

2 2

2 3

4 5

Sample Output :

12

26

312

Constraints :

1 <= T <= 10000

1 <= N, M <= 100000

[Home](#) » [Compete](#) » [July Cook Off](#) » Ripple-Carry Adder

## Ripple-Carry Adder Problem Code: RIPPLE

Amortized analysis deals with analyzing the average amount of work done per operation over a series of operations. In some cases, the average amount of work done per operation is dramatically less than the worst case analysis indicates.

A typical example is counting the number of times a bit is flipped in a ripple-carry counter. A ripple-carry counter is an implementation of a binary counter where incrementing from  $B$  to  $B+1$  is done in the following manner. Say the binary number is represented as  $B = b_{n-1}b_{n-2}...b_1b_0$  where  $b_i$  is the bit corresponding to  $2^i$ .  $B$  is increased to  $B+1$  in the following manner:

```
i := 0
while b_i == 1
    b_i := 0
    i := i+1
b_i := 1
```

This doesn't account for overflow when increasing from  $2^{n-1}$  to  $2^n$ , but we'll ignore that error for this problem.

Each time a bit is changed from 0 to 1 or from 1 to 0 we say the bit is "flipped". In the worst case, we may have to flip every bit. However, a standard result says the average number of bits flipped per increment when counting from 0 to  $2^{n-1}$  is less than 2.

Being the curious sort, you decide to explore this result in a slightly more general setting. That is, you want to know how many bits are flipped when the counter is incremented from a number  $a$  to  $b$  where  $a < b$ .

### Input

The first line denotes the number of test cases (about 20).

Each test case consists of three lines. The first contains a single integer  $n$  between 1 and 100,000 denoting the number of bits in the counter. The second line contains the number  $a$  written in binary and the third line contains the number  $b$  written in binary. Both  $a$  and  $b$  are described using exactly  $n$  bits.

### Output

The output for each test case consists of a single line that describes the total number of bits flipped when the counter is increased from  $a$  to  $b$ . This number should be expressed in binary with the most significant bit being 1 (i.e. no leading zeros should pad the output).

### Example

**Input:**

3

1

0

1

2

00

11

3

011

100

**Output:**

1

100

11

[Home](#) » [Compete](#) » [July Cook Off](#) » Head office building

## Head office building Problem Code: BUILDING

CodeChef Inc. owns a rectangular piece of land sized  $w \times h$ . Its edges are parallel to the axes, with the bottom-left corner at  $(0,0)$  and top-right corner at  $(w,h)$ . They intend to build a new head office within this land. The new office will be a square whose center is located at a point with integer coordinates, and whose diagonals are parallel to the axes and have length  $2 \times d$ .

Unfortunately, there are  $n$  barricades on the land, some of which may need to be removed in order for the office to be built. The  $(i)$ th barricade is located at  $(X_i, Y_i)$  and will cost  $C_i$  to remove. The office may only be built once all points it covers (including the boundaries) are free of barricades.

### Request

Help them find the minimum total cost of barricade removal.

## Input

- The first line contains the integers  $w, h, d, n$ , respectively.
- Within following  $n$  lines, the  $i$ -th line contains the integers  $X_i, Y_i, C_i$  respectively.

## Output

Output on a single line an integer which is the minimum cost found.

## Example

### Input:

4 3 1 4

1 3 5

3 3 4

2 2 1

2 1 2

### Output:

2

## Limitations

- $2 \leq w, h \leq 1\ 000$
- $0 \leq n \leq 200\ 000$
- $2 \leq d \leq \min(w, h)$
- $0 < C_i \leq 10\ 000$
- All the listed barricades are inside or on the boundaries of the CodeChef's land.
- There is at most one barricade at each point.

AUG10

[August 2010  
Challenge](#)

01 Aug 2010  
15:00:00

10 days

299

[Home](#) » [Compete](#) » [August 2010 Challenge](#) » Optimizing Production and Sales

# Optimizing Production and Sales

Problem Code: FACTORY

Your boss at the gadget company has recently asked you to help optimize operating costs. Your company is in charge of both manufacturing and selling their world-famous gadgets. Currently, there are a number of factories open in various locations around the world and each store receives their gadgets from precisely one factory.

Each factory has an associated operating cost and the cost of supplying a store from a factory is proportional to the distance between the store and factory. Your job is to close some factories and assign an open factory to each store. The cost of such an assignment is the sum of the factory operating costs of the open factories plus the total cost of assigning each store to a factory. Of course, your objective is to find a cheap solution.

To reiterate, each store must be supplied by exactly one open factory and each factory may supply any number of stores. The operating cost of the factory does not depend on how many stores it is supplying.

## Input

The first line contains a single integer  $T \leq 30$  indicating the number of test cases to follow. The first line of each test case consists of two integers  $F$  and  $S$ , each between 1 and 100, where  $F$  is the number of factories and  $S$  is the number of stores. The next line contains  $F$  non-negative floating point values where the  $i$ 'th value is the operating cost of factory  $i$ . The next  $S$  lines contain  $F$  non-negative floating point values where the  $i$ 'th value on the  $j$ 'th line is the cost of supplying store  $j$  with factory  $i$ .

Since the cost of supplying a store by a factory is proportional to the distance between the store and factory, you may assume that a sort of triangle inequality holds. Say  $d[j,i]$  is the cost of supplying store  $j$  from factory  $i$ . Then for any two stores  $j,j'$  and any two factories  $i,i'$  we have  $d[j,i] \leq d[j,i'] + d[j',i'] + d[j',i]$ . To say it plainly, the shortest distance between store  $j$  and factory  $i$  is the direct route.

All floating point values are between 0 and 1000000 and contain at most two points of precision. Consecutive test cases are separated by a blank line including a blank line immediately preceding the first case.

## Output

For each test case you are to output two lines. The first line consists of  $F$  values, each either 0 or 1. If the  $i$ 'th value is 1 then factory  $i$  is open and if the  $i$ 'th value is 0 then factory  $i$  is closed. The second line consists of  $S$  values, each between 1 and  $F$ . Here, the  $j$ 'th value is the index of an **open** factory from which store  $j$  is supplied.

## Example

**Input:**

2

3 3

1 2.5 3

3 1.7 1.5

1 2.1 1.2

2.1 1.4 2

2 3

0 15

4 0

4 0

4 0

**Output:**

1 0 1

3 1 1

1 0

1 1 1

## Scoring

For each test case, let  $K$  be the cost of the solution that keeps all factories open and assigns each client to any nearest factory (i.e. the current cost of manufacturing and supplying gadgets before your, hopefully cheaper, solution is applied). We will compute the cost of your solution, say  $L$ , and assign a score of  $L/K$  to the test case (we guarantee  $K \geq 1$  for each test case). The final score is then the sum of the scores for each test case. The goal is, of course, to minimize the total score.

For example, the first test case has of  $K = (1 + 2.5 + 3) + (1.5 + 1 + 1.4) = 10.4$  (the first set of parenthesis contains the cost of the factories and the second contains the cost of assigning stores to their cheapest factory). The output solution has  $L = (1 + 3) + (1.5 + 1 + 2.1) = 8.6$ . Therefore, the score for the first test case is  $8.6/10.4 = 0.826923$ . Similarly, the second test case has  $K = (0 + 15) + (0 + 0 + 0)$  and the solution output has  $L = 0 + (4 + 4 + 4) = 12$  so the score is  $12/15 = 0.8$ .

## Test Data

Some of the test data is hand-crafted to make certain heuristics perform poorly and some of the test data is randomly generated according to different distributions. [Home](#) » [Compete](#) » [August 2010 Challenge](#) » Exit code

## Exit code Problem Code: ECODE

Professor X got lost in a maze of an ancient tomb in Egypt. While he was finding the way to escape, he got a message of the tomb builders on the old walls:

- The code to open the exit door is the sequence C of n digits formed  $c_1..c_n$  ( $c_i \in [0,9]$  ).
- For every sequence C, combining with the given integers A,B, call:
  - $h_0=0$
  - $h_i = (h_{i-1} \times A + c_i) \bmod B$
- The smallest sequence C (in lexicological order) satisfying  $h_n=G$  (where G is a given integer) is the exit code which professor X needs.

### Request

Give the integers n,A,B,G, help professor X find out the exit code!

### Input

One and only line contains the integers n,A,B,G, respectively, each of them is separated with at least one space character.

### Output

Output in a single line the exit code found.

### Example

**Input:**  
3 11 111 92

**Output:**  
084

### Limitations

- $1 \leq n \leq 12$
- $10 \leq A, B \leq 10^{15}$
- $0 \leq G < B$

- The input satisfies that the answer always exist.

[Home](#) » [Compete](#) » [August 2010 Challenge](#) » Obstacle Course

## Obstacle Course Problem Code: COURSE

- A number of traffic cones have been placed on a circular racetrack to form an obstacle course. You are asked to determine the largest sized car that can navigate the course. For simplicity, the cones are assumed to have zero width and the car is perfectly circular and infinitely maneuverable. The track itself is the area between 2 concentric circles.
- Formally, the course can be navigated by a car of radius  $c$  if there exists a closed loop around the center of the track which lies between the circles forming the track, and every point on the loop is at least  $c$  distance away from each cone and each boundary of the track.

- **Input:**

- Input begins with an integer  $T$  (about 25), the number of test cases. Each test case begins with 2 integers  $r$  and  $R$  ( $1 \leq r < R \leq 25000$ ). The racetrack is the area between the circles centered at  $(0,0)$  with radii  $r$  and  $R$ . The next line of input contains an integer  $N$  ( $0 \leq N \leq 500$ ), the number of cones.  $N$  lines follow, each containing the coordinates of a cone. The coordinates are integers, and are guaranteed to lie within the track, and be distinct. Cases are separated by a blank line.

- **Output:**

- For each input, output on a single line the diameter of the largest car that can navigate the course, rounded to 3 decimal places.

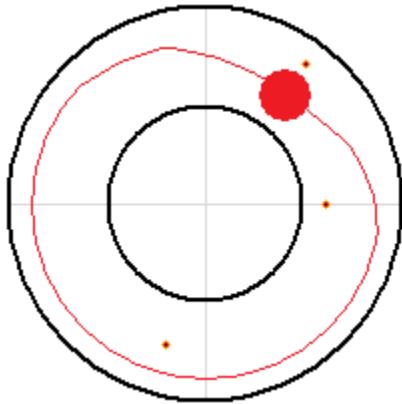
- **Sample input:**

- 1
- 
- 5 10
- 3
- 6 0
- 5 7
- -2 -7

- **Sample output:**

- 2.720

- **Explanation:**
- The image below shows the course corresponding to the sample input. The black circles represent the boundaries of the racetrack, the small dots the cones, and the filled red circle the car. Also shown is one possible path of the car through the course.



[Home](#) » [Compete](#) » [August 2010 Challenge](#) » Swarm of Polygons

## Swarm of Polygons Problem Code: SWARM

There is a regular  $n$ -gon. Some points are marked on each of its sides. There are  $x_1$  point marked on the first side,  $x_2$  – on the second, ...,  $x_n$  – on the  $n$ th. The marked points do not coincide with the vertices of the  $n$ -gon. You can choose no more than one of the marked points from each side and form a convex non-degenerate polygon by connecting all those points with lines. Now your task is to find the number of different  $k$ -gons that can be formed that way.

### Input

The first line of input file contains positive integer  $t$  – the amount of test cases. Next  $t$  lines contain six integers each:  $n, k, a, b, c, m$ . Here  $n$  is the number of sides of the initial  $n$ -gon. The amount of marked points on the first side of this  $n$ -gon is  $x_1 = a$ , the amount of the marked points on the following sides is  $x_i = (b * x_{i-1} + c) \bmod m$ , for  $i > 1$ .

### Constraints

$1 \leq t \leq 30$   
 $3 \leq n \leq 10^9$   
 $3 \leq k \leq 20$   
 $1 \leq b, c, m \leq 10^6$   
 $0 \leq a < m$

### Output

For each test case output the number of k-gons that can be formed modulo 1000000007.

## Example

**Input:**

2

4 3 1 2 2 191

10 5 1 113 157 999991

**Output:**

1228

328836201

[Home](#) » [Compete](#) » [August 2010 Challenge](#) » Stepping Numbers

## Stepping Numbers Problem Code: STEPNUMS

Call a number stepping if adjacent digits, as well as the first and last digits, differ by one. How many n-digit base 11 stepping numbers are there? Give your answer modulo 4294967143.

For example, 8789A9 is a 6-digit base 11 stepping number. 9A0A and 234 are not stepping.

### Input

The first line contains an integer T, the number of test cases (about 20000). Each of the next T lines contains an integer n ( $2 \leq n < 2^{64}$ ).

### Output

Output the answer to each test case on a separate line.

## Example

**Input:**

4

2

4

6

10

**Output:**

19

54

171

1958

[Home](#) » [Compete](#) » [August 2010 Challenge](#) » Genetics

## Genetics Problem Code: GENETICS

Genetic engineering is fun. The scientists have gathered several DNAs and want to create something new out of those. Each DNAs can be represented as a sequence of bases A, G, T, C. Let  $\text{DNA}[a..b]$  be the subsequence of DNA starting from index a finishing in b inclusive, and  $\text{DNA}[a..]$  be the subsequence of DNA starting from index a to the end. Scientist can perform the following operations on the DNAs:

- cross operation – they take DNA1 and DNA2 and numbers k1 and k2. Then two new DNAs are created: DNA3 =  $\text{DNA1}[1..k1]+\text{DNA2}[k2+1..]$  and DNA4 =  $\text{DNA2}[1..k2]+\text{DNA1}[k1+1..]$ .
- mutate operation – they take a DNA, number k and one of the bases. Then they replace the base in position k in DNA with that base.
- also they need to know certain characteristics of those DNAs. So they can perform count operation – they take DNA and numbers k1 and k2 ( $k1 \leq k2$ ). This operation should return the number of A, G, T, C bases in  $\text{DNA}[k1..k2]$ .

The initial DNAs are numbered from 1 to n, where n is the amount of those DNAs. The new DNAs created in the cross operations are numbered with consecutive integers. You are to write a program to help scientists perform those operations.

### Input

The first line of input file contains number  $n$  – the amount of initial DNAs. Each of the following  $n$  lines contains the description of each DNA. The next line contains number  $q$  – the amount of operations to perform. The next  $q$  lines contain the description of each operation in the following form:

- CROSS id1 id2 k1 k2
- MUTATE id k m
- COUNT id k1 k2

## Constraints

$1 \leq n \leq 20$   
 $1 \leq q \leq 30000$

The length of each initial DNA does not exceed 30000. The length of any DNA formed in cross operation won't exceed 2000000000. Total amount of different DNAs won't exceed 10000. It is guaranteed that all operations are correct.

## Output

For each count operation print four integers: the number of each of the bases in the given subsequence of the given DNA.

## Example

**Input:**

```
2
CTCGC
TGC GG
5
MUTATE 1 2 A
COUNT 2 2 4
MUTATE 2 1 G
CROSS 2 1 1 5
COUNT 4 3 6
```

**Output:**

0 2 0 1

0 2 0 2

CO	A	21	2	2
OK	u	Au	ho	3
02	g	g	urs	
	20	20	30	
	10	10	mi	1
	21:	21:	nut	
	30:	30:	es	
	00	00		
	ff			

[Home](#) » [Compete](#) » [August Cook Off](#) » Cleaning Up

## Cleaning Up Problem Code: CLEANUP

After a long and successful day of preparing food for the banquet, it is time to clean up. There is a list of  $n$  jobs to do before the kitchen can be closed for the night. These jobs are indexed from 1 to  $n$ .

Most of the cooks have already left and only the Chef and his assistant are left to clean up. Thankfully, some of the cooks took care of some of the jobs before they left so only a subset of the  $n$  jobs remain. The Chef and his assistant divide up the remaining jobs in the following manner. The Chef takes the unfinished job with least index, the assistant takes the unfinished job with the second least index, the Chef takes the unfinished job with the third least index, etc. That is, if the unfinished jobs were listed in increasing order of their index then the Chef would take every other one starting with the first job in the list and the assistant would take every other one starting with the second job on in the list.

The cooks logged which jobs they finished before they left. Unfortunately, these jobs were not recorded in any particular order. Given an unsorted list of finished jobs, you are to determine which jobs the Chef must complete and which jobs his assistant must complete before closing the kitchen for the evening.

### Input

The first line contains a single integer  $T \leq 50$  indicating the number of test cases to follow. Each test case consists of two lines. The first line contains two numbers  $n, m$  satisfying  $0 \leq m \leq n \leq 1000$ . Here,  $n$  is the total number of jobs that must be completed before closing and  $m$  is the number of jobs that have already been completed. The second line contains a list of  $m$  distinct integers between 1 and  $n$ . These are the indices of the jobs that have already been completed. Consecutive integers are separated by a single space.

### Output

The output for each test case consists of two lines. The first line is a list of the indices of the jobs assigned to the Chef. The second line is a list of the indices of the jobs assigned to his assistant. Both lists must appear in increasing order of indices and consecutive integers should be separated by a single space. If either the Chef or the assistant is not assigned any jobs, then their corresponding line should be blank.

## Example

**Input:**

```
3
6 3
2 4 1
3 2
3 2
8 2
3 8
```

**Output:**

```
3 6
5
1
```

```
1 4 6
2 5 7
```

[Home](#) » [Compete](#) » [August Cook Off](#) » Arranging the Appetizers

## Arranging the Appetizers Problem Code: ARRANGE

The Chef has prepared the appetizers in the shapes of letters to spell a special message for the guests. There are  $n$  appetizers numbered from 0 to  $n-1$  such that if the appetizers are arrayed in this order, they will display the message. The Chef plans to display them in this

order on a table that can be viewed by all guests as they enter. The appetizers will only be served once all guests are seated.

The appetizers are not necessarily finished in the same order as they are numbered. So, when an appetizer is finished the Chef will write the number on a piece of paper and place it beside the appetizer on a counter between the kitchen and the restaurant. A server will retrieve this appetizer and place it in the proper location according to the number written beside it.

The Chef has a penchant for binary numbers. The number of appetizers created is a power of 2, say  $n = 2^k$ . Furthermore, he has written the number of the appetizer in binary with exactly  $k$  bits. That is, binary numbers with fewer than  $k$  bits are padded on the left with zeros so they are written with exactly  $k$  bits.

Unfortunately, this has unforeseen complications. A binary number still "looks" binary when it is written upside down. For example, the binary number "0101" looks like "1010" when read upside down and the binary number "110" looks like "011" (the Chef uses simple vertical lines to denote a 1 bit). The Chef didn't realize that the servers would read the numbers upside down so he doesn't rotate the paper when he places it on the counter. Thus, when the server picks up an appetizer they place it the location indexed by the binary number when it is read upside down.

You are given the message the chef intended to display and you are to display the message that will be displayed after the servers move all appetizers to their locations based on the binary numbers they read.

## Input

The first line consists of a single integer  $T \leq 25$  indicating the number of test cases to follow. Each test case consists of a single line beginning with an integer  $1 \leq k \leq 16$  followed by a string of precisely  $2^k$  characters. The integer and the string are separated by a single space. The string has no spaces and is composed only of lower case letters from `a` to `z`.

## Output

For each test case you are to output the scrambled message on a single line.

## Example

**Input:**

2

2 chef

4 enjoyourapplepie

**Output:**

cehf

eayejpuinpopolre

[Home](#) » [Compete](#) » [August Cook Off](#) » Fetching Cooking Tools

## Fetching Cooking Tools Problem Code: TOOLS

Like any good boss, the Chef has delegated all cooking jobs to his employees so he can take care of other tasks. Occasionally, one of the cooks needs a tool that is out of reach. In some of these cases, the cook cannot leave their workstation to get the tool because they have to closely watch their food. In such cases, the Chef simply fetches the tool for the cook.

Unfortunately, many different cooks have simultaneously requested a tool they cannot reach. Thankfully, no two cooks requested the same tool. Nobody else is available to help the Chef so he has to deliver all of the tools himself. He has to plan a trip around the kitchen in such a way that he picks up each tool and delivers it to the requesting cook. Since the Chef has two hands, he may carry up to two tools at once.

Once the last item has been delivered, the Chef also has to return to his starting position. This must be done as fast as possible so the Chef wants to do this while traveling the minimum possible distance.

### Input

The first line contains a single integer  $T \leq 20$  indicating the number of test cases. Each test case begins with a single integer  $n$ , between 1 and 8, indicating the number of requests being made. The following  $n$  lines describe the locations of cooks and the tools they request. The  $i$ 'th such line begins with two integers  $c_x, c_y$  describing the location of the cook and ends with two more integers  $t_x, t_y$  describing the location of the corresponding requested tool. No tool will be located at the same location as the cook who requests it.

The values  $c_x$ ,  $c_y$ ,  $t_x$ , and  $t_y$  corresponding to any cook or tool will always lie between 0 and 1000 (inclusive). Finally, the kitchen is laid out into square workstations on a grid so the distance between two points  $x, y$  and  $x', y'$  is precisely their Manhattan distance  $|x-x'| + |y-y'|$ .

### Output

The output for each test case consists of a single line. This line should display a single integer indicating the minimum distance the Chef must travel to deliver the tools to the cooks and return to his start location 0,0. Of course, he may only deliver a tool to a cook if he has already picked up the tool and he may not carry more than two tools at a time.

## Example

### Input:

```

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

```

### Output:

```

4
10
6

```

## Notes

In the second case, the Chef dropped of the first tool to the first cook and then picked up the tool for the third cook.

In the last test case, there are three different cooks requesting three different tools. It just so happens that all three tools are at the same location. Still, the Chef can only carry two of the tools from this location at once.

[Home](#) » [Compete](#) » [August Cook Off](#) » Baking Cupcakes

## Baking Cupcakes Problem Code: MUFFINS

The Chef is planning on serving cupcakes for dessert. However, time is running out and the Chef has only enough time to bake one batch. Thankfully, the cupcake baking tin the Chef plans on using has barely enough spaces to ensure each guest receives one cupcake.

To the Chef's dismay, some of the spaces in the cupcake tin are too close to each other. If cupcake batter is placed in two such spaces then they will bake together and neither will have the aesthetically pleasing round shape one expects from gourmet cupcakes.

The Chef wants to impress his guests with this dessert, so he wants all cupcakes to be perfectly round. You must determine if it is possible for the desired number of cupcakes to be baked without any two baking in to each other.

### Input

The first line consists of a single integer  $T \leq 30$  denoting the number of test cases to follow. Each test case begins with a single line consisting of three integers  $n, m$ , and  $g$ . Here,  $n$  is the number of spaces in the cupcake tin,  $m$  is the number of conflicting pairs of spaces in the tin, and  $g$  is the number of guests. Following this line is  $m$  lines describing the conflicting pairs. Each line consists of two distinct integers  $i$  and  $j$ , both between 0 and  $n-1$ . This means spaces  $i$  and  $j$  are conflicting so batter may only be placed in at most one of them.

The bounds are  $1 \leq n \leq 1,000$ ,  $1 \leq m \leq 20,000$ , and  $0 \leq g \leq n$ . Since the cupcake tin has barely more spaces than guests (if any), then we also have  $n-g \leq 15$ .

### Output

The output for each test case consists of a single line containing "Possible" if the Chef can bake all cupcakes perfectly round or "Impossible" if the Chef must settle for some cupcakes being imperfect.

### Example

Input:

2

3 2 2

0 1

1 2

3 3 2

0 1

1 2

2 0

**Output:**

Possible

Impossible

[Home](#) » [Compete](#) » [August Cook Off](#) » Backup Functions

## Backup Functions Problem Code: FUNCTION

One unavoidable problem with running a restaurant is that occasionally a menu item cannot be prepared. This can be caused by a variety of reasons such as missing ingredients or malfunctioning equipment.

There is an additional problem with such situations. A customer who has spent many minutes deciding between menu items can get quite annoyed when they place the order and the server tells them the item is not available. To mitigate this effect, the Chef declares that all servers must have what he calls a "backup function". This is a function  $f$  from menu items to menu items. For each menu item  $x$ ,  $f(x)$  is an alternative menu item that the server may suggest to the customer in the event the customer requests menu item  $x$  when  $x$  is not available.

Of course, if a given item is not available then some other items make better suggestions than others. So, for some pairs of items  $x$  and  $y$  the Chef has determined a numeric value describing the effectiveness of the assignment  $f(x) = y$ . Higher values indicate the proposed substitute is similar to the original and lower values indicate the proposed substitute is not very similar to the original. Such effectiveness values are symmetric meaning that if the effectiveness of assignment  $f(x) = y$  is  $v$ , then the effectiveness of the assignment  $f(y) = x$  is also  $v$ .

You will be given a list of pairs of menu items. Each such pair will come with an associated effectiveness value. You are to compute a backup function  $f$  from these pairs. However, there is one additional constraint. For personal reasons, the Chef is opposed to using two items as backups for each other. Thus, for any two menu items  $x$  and  $y$ , it cannot be that  $f(x) = y$  and  $f(y) = x$ . Your goal is to compute a backup function of maximum possible quality. The quality of the backup function is simply defined as the sum of the effectiveness values of the assignments  $f(a)$  for each item  $a$ .

### Input

The first line contains a single integer  $T \leq 30$  indicating the number of test cases.

Each test case begins with two integers  $N$  and  $M$  where  $3 \leq N \leq 10,000$  and  $3 \leq M \leq 50,000$  where  $N$  is the number of items in the menu. The menu items will be numbered 1 through  $N$ .

$M$  lines follow, each containing three integers  $a, b$ , and  $v$ . Here  $1 \leq a < b \leq N$  and  $|v| \leq 100,000$ . Such a triple indicates that  $f(a) = b$  or  $f(b) = a$  (but not both) may be used in a backup function. Setting either  $f(a) = b$  or  $f(b) = a$  has effectiveness  $v$ . Each pair  $1 \leq a < b \leq N$  will occur at most once in the input.

Test cases will be separated by a single blank line (including a blank line preceding the first test case).

## Output

The output for each test case consists of a single line containing the maximum possible quality of a backup function  $f$  that does not assign any pair of items to each other. To be explicit, the assignment  $f(a) = b$  has effectiveness  $v$  where  $v$  is the value associated with the  $a, b$  pair in the input. Then the quality of a backup function is just the sum of the effectiveness values of the assignment for each item. If no backup function can be constructed from the given input pairs, then simply output "impossible" (note the lowercase 'i').

We may only assign  $f(a) = b$  or  $f(b) = a$  if  $a, b$  is an input pair. Furthermore, a backup function defines a single item  $f(a)$  for every item  $a$  between 1 and  $n$ . Note that  $f(a) = a$  is not valid since  $f(a)$  is supposed to be an alternative item if menu item  $a$  is not available (and all input pairs  $a, b$  have  $a < b$  anyway). Finally, based on the Chef's additional constraint we cannot have a backup function with both  $f(a) = b$  and  $f(b) = a$  for any input pair  $a, b$ .

## Example

Input:

3

3 3

1 2 1

2 3 2

1 3 3

6 7

1 2 0

1 3 0

1 4 0

2 3 0

2 4 0

3 4 0

5 6 0

4 6

1 2 1

1 3 -2

2 3 4

1 4 -8

2 4 16

3 4 -32

**Output:**

6

impossible

19

Note, one possible solution for the first test case is  $f(1) = 2$ ,  $f(2) = 3$ , and  $f(3) = 1$ . The second is impossible since the only possibility for  $f(5)$  is 6 and the only possibility for  $f(6)$  is 5 and both cannot be used in a backup function. Finally, a possible solution for the last test case is  $f(1) = 2$ ,  $f(2) = 3$ ,  $f(3) = 1$ , and  $f(4) = 2$ .

SE	<u>Sept</u>	01	1	
PT	<u>emb</u>	Se	0	
	<u>er</u>	p	d	5
10	<u>2010</u>	20	a	0
	<u>Chall</u>	10	y	
	<u>enge</u>	15:	s	8
		00:		
		00		

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Brush Fire

## Brush Fire Problem Code: FIRE

Help! A brush fire has started near your house and you are the only one who can help extinguish it. The only tool you have at your disposal is a single fireproof barrier that can protect one bush at a time, provided the bush has not yet caught fire already. For simplicity, we will assume the flames spread according to the following discrete time model.

Initially, a single bush, say  $s$ , is on fire. You choose a bush, say  $k$ , to save. The flames then leap from  $s$  to every bush near  $s$  with the exception of bush  $k$  if it was near  $s$ . Say these new burning bushes form a set  $B$ . Bush  $s$  is then reduced to ashes and will never burn again. Now that the bushes in  $B$  are on fire, you may move the protective barrier from bush  $k$  to some other bush  $k'$ . The flames leap from bushes in  $B$  to every bush that has not yet burned (i.e. not  $s$  or in  $B$ ) and is close enough to some bush in  $B$  except, perhaps, bush  $k'$ . Say this new set of burning bushes is  $B'$ . The bushes in set  $B$  are reduced to ashes and will never burn again. Now, you may move the protective barrier from  $k'$  to another bush to protect it from the flames that will spread from bushes in  $B'$  and so on. This process repeats until there are no more bushes on fire.

The bushes are arranged in a peculiar manner. Before the fire started, each bush was close enough to spread a fire to at most three other bushes. Moreover, the bush that was initially on fire is actually close enough to only at most two other bushes. It is understood that for any two bushes  $A$  and  $B$ , if bush  $A$  is close enough to bush  $B$  then bush  $B$  is also close enough to bush  $A$ . Finally, the graph underlying the "close enough" relation has no cycles.

For whatever reason, some of the bushes hold some sentimental value to you. For this reason, you want to save all of these bushes.

### Input

The first line of the input contains a single integer  $T \leq 40$  indicating the number of test cases that will follow. Each test case begins with three integers  $n$ ,  $s$ , and  $t$  satisfying  $1 \leq n \leq 10,000$  and both  $s$  and  $t$  are between 1 and  $n$ . Here,  $n$  is the number of bushes (indexed from 1 to  $n$ ),  $s$  is the index of the bush that is initially on fire, and  $t$  is the number of bushes you want to save.

Following this, there are  $n$  lines describing the "close enough" relation. The  $i$ th line starts with an integer  $0 \leq c_i \leq 3$  meaning bush  $i$  is close enough to  $c_i$  other bushes. The remaining  $c_i$  integers on this line are the indices of the bushes that bush  $i$  is close to. As mentioned earlier, the input will be such that if bush  $i$  is close enough to bush  $j$ , then bush  $j$  is also close enough to bush  $i$ . Finally, we also guarantee that  $c_s \leq 2$ .

The last line of input for each test case consists of  $t$  distinct integers between 1 and  $n$ . These correspond to indices of bushes you want to save.

Test cases are separated by a single blank line (including a blank line preceding the first test case).

## Output

For each test case, you are to output a single line consisting of either "yes" or "no", depending on whether or not it is possible to save all of the indicated bushes.

## Example

Input:

3

3 1 2

2 2 3

1 1

1 1

2 3

3 1 1

2 2 3

1 1

1 1

2

7 1 3

2 2 3

3 1 4 5

3 1 6 7

1 2

1 2

1 3

1 3

4 5 7

**Output:**

no

yes

yes

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Multiples of 3

## Multiples of 3 Problem Code: MULTQ3

There are N numbers  $a[0], a[1]..a[N - 1]$ . Initially all are 0. You have to perform two types of operations :

1) Increase the numbers between indices A and B by 1. This is represented by the command "0 A B"

2) Answer how many numbers between indices A and B are divisible by 3. This is represented by the command "1 A B".

Input :

The first line contains two integers, N and Q. Each of the next Q lines are either of the form "0 A B" or "1 A B" as mentioned above.

Output :

Output 1 line for each of the queries of the form "1 A B" containing the required answer for the corresponding query.

Sample Input :

4 7

1 0 3

0 1 2

0 1 3

1 0 0

0 0 3

1 3 3

1 0 3

Sample Output :

4

1

0

2

Constraints :

1 <= N <= 100000

1 <= Q <= 100000

0 <= A <= B <= N - 1

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Cake Production Line

## Cake Production Line Problem Code: CAKES

Today, the Chef has been asked to bake a variety of cakes for some very important regular customers. Each cake has multiple components that may be prepared in parallel by different bakers. For example, the bakers working for the chef can simultaneously bake the cake, whip the icing, boil the fondant, make the decorations, etc. Each of these tasks take time to accomplish and may this time vary between different cakes (e.g. some cakes may have multiple layers, others may require more icing than usual, etc).

Once all of the pieces of the cake have been prepared by the bakers, the Chef can assemble the final product in virtually no time at all. So, a cake is said to be completed once its last component has been finished by the bakers. Each baker specializes in exactly one task so if one baker finishes early, he cannot help another baker.

The Chef wants to prepare the cakes as quickly as possible. To measure this, he has assigned a "weight" to each customer according to how much he values their patronage. The "weighted completion time" of a cake is the time it takes to complete it times the weight of its associated customer. The Chef wants to try and minimize the sum of the weighted completion times of all cakes. To do this he determines, for each baker, a permutation of the customers. The baker then prepares all of the components he creates for the requested cakes in the order he is given. He starts working on the first cake in the list and as soon as he finishes creating his component for one cake, he immediately starts working on the next cake in the list until he has processed all of the cakes.

A default order is usually specified. It is the order where each baker simply processes the cakes in the same order they appear on the initial list. The Chef suspects that there may be a better ordering of the jobs. Help him find a schedule with better weighted completion time!

### Input

The first line consists of a single integer  $T \leq 30$  indicating the number of test cases to follow. Each test case begins with two integers  $n, m$  on a single line, both between 1 and 200. Here,  $n$  is the number of different cakes that have been ordered and  $m$  is the number of different components that must be assembled per cake. Thus, the problem statement says that there are also  $m$  different bakers, one per job type.

Following this are  $n$  different lines, each containing  $m+1$  integers. The first integer on the  $i$ 'th line is the weight of customer  $i$ , a positive integer between 1 and 10,000. Of the remaining  $m$  integers, the  $j$ 'th integer indicates the amount of time that must be spent by baker  $j$  on cake  $i$ , again a positive integer between 1 and 10,000.

### Output

For each test case you are to output  $m$  different lines, each describing the order the respective baker prepares their component for the cakes. Specifically, the  $i$ 'th line of the output for a given test case should be a permutation of the numbers 1 through  $n$ . The first number is the first cake the  $i$ 'th baker works on, the second number is the second cake the  $i$ 'th baker works on, and so on.

## Example

**Input:**

2

3 2

1 1 2

4 2 4

5 10 1

2 2

1 2 3

4 5 6

**Output:**

1 2 3

2 3 1

2 1

2 1

## Scoring

For each test case, we will compute the weighted completion time of the schedule where each baker simply processes the cakes in the order 1, 2, ...,  $n$ . Call this value  $K$ . We will also compute the weighted completion time of the schedule specified by your output, call this value  $L$ . The score for this test case is then  $L/K$  and the total score for the entire data set is  $L/K$ . Your goal is to minimize this quantity.

For example, in the first test case, the default ordering for each baker is 1,2,3. So the first cake is completed after 2 units of time, the second cake is completed after 6 units of time and the third cake is completed after 13 units of time. Therefore, the weighted completion time is  $1*2 + 4*6 + 5*13 = 91$ . However, following the schedule in the sample output we see that the second cake is finished after 4 units of time, the first cake is finished after 7 units of time and the third cake is finished after 13 units of time for a weighted completion time of  $1*7 + 4*4 + 5*13 = 88$ . Therefore, the score for this particular test case is 88/91.

Similarly, the weighted completion time of the default ordering of the second test case is  $1*3 + 4*9 = 39$  whereas the specified ordering in the sample output has a weighted completion time of  $1*9 + 4*6 = 33$  so the total score for this case is 33/39.

## Test Data

Some test data is hand-crafted to cause certain heuristics to perform poorly and some data is randomly generated according to varying distributions.

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Trees Again

## Trees Again Problem Code: TREES

Given a tree, you need to count how many subtrees exist such that the length of the longest path in the subtree is at most K.

Input :

The first line contains the number of test cases T. T test cases follow. For each test case, the first line contains N and K. The following N - 1 lines contain two integers ai and bi, indicating an edge between nodes ai and bi in the tree. There is a blank line after each test case.

Output :

Output T lines, one corresponding to each test case, containing the required answer.

Sample Input :

2

3 1

0 1

1 2

6 3

0 1

1 2

2 3

2 4

3 5

Sample Output :

5

23

Constraints :

1 <= T <= 2000

2 <= N <= 60

0 <= ai,bi < N

1 <= K <= N - 1

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Bytelandian Shoppers

## Bytelandian Shoppers Problem Code: BYTESHOP

There are N shoppers today at the Bytelandian Shopping Mall. There are M different items at sale at the mall. In how many ways can shoppers purchase R items in all such that everyone buys atleast one item, and every item is brought by atleast one person ? A shopper can purchase any particular item at most once.

Input :

The first line contains the number of test cases T. T lines follow containing 3 integers: N,M and R.

(1 <= T <= 100. 1 <= N,M <= 200. 1 <= R <= N \* M)

Output :

Output  $T$  lines, one for each test case, containing the output for the corresponding test case.

Output all values modulo 1000000007

Sample Input :

3

1 1 1

2 1 1

2 3 3

Sample Output :

1

0

6

[Home](#) » [Compete](#) » [September 2010 Challenge](#) » Communicating Servers

## Communicating Servers Problem Code: SERVERS

The servers working at the Chef's restaurant communicate over a distance using wireless devices to relay requests as quickly as possible. Each server has exactly one device. However, unforeseen technical difficulties make it such that some pairs of devices cannot communicate directly. Still, if a server A wants to communicate with another server B but their devices don't communicate directly, then server A may ask relay their request to server B using intermediate devices/servers.

There are two additional complications. First, the Chef is worried that relaying requests using intermediate servers may get confusing. Thus, for any two servers who can communicate, either directly or indirectly through other servers, there should be a unique way to pass messages between these servers. Second, each pair of devices that can

communicate directly does so over a specified frequency. However, if too many pairs of devices communicate over the same frequency, then the messages may be corrupted.

Because of these two problems, the Chef has ordered that some pairs of devices be forbidden to communicate. This may result in some pairs of servers who could communicate before to be unable to communicate, but the Chef thinks this negative effect is outweighed by the guarantee that the above problems don't occur.

Your goal is to determine the maximum number of pairs of devices that may be allowed to communicate directly without having to go through intermediate devices. To restate the constraints, if  $S$  is such a set of pairs of devices then there should be at most one way to relay a message between any two servers using only pairs of devices in  $S$ . Furthermore, for each frequency  $f$ , the number of pairs of devices in  $S$  that communicate over frequency  $f$  is less than some prescribed value  $c(f)$ .

## Input

The first line of input contains a single integer  $T \leq 30$  indicating the number of test cases. Each test case begins with three integers  $n, m$ , and  $k$  on a single line. Here,  $n$  is the number of servers,  $m$  is the number of pairs of devices that are able to communicate directly, and  $k$  is the number of available frequencies. The next line consists of  $k$  integers, each between 1 and  $m$ , where the  $i$ 'th such integer is the value  $c(i)$  specifying how many devices may communicate using frequency  $i$  without worrying about messages being corrupted.

Following this is  $m$  lines where each line describes a pair of devices that may communicate directly. Each line begins with two distinct integers  $u, v$ , each between 1 and  $n$ , and ends with an integer  $f$ , between 1 and  $k$ . This means that devices  $u$  and  $v$  may communicate directly using frequency  $f$ . No pair of devices will appear more than once in the input. There will be a blank line separating test cases including a blank line preceding the first case.

Bounds:  $1 \leq n \leq 40$ ,  $1 \leq m \leq 200$ , and  $1 \leq k \leq m$ .

## Output

For each test case you are to output a single line consisting of a single integer. This integer should be the size of  $S$  where  $S$  is a maximum size subset of communicating pairs satisfying the constraints outlined in the last paragraph of the problem description.

## Example

**Input:**

3

4 4 2

1 1

1 2 1

2 3 2

3 4 1

4 1 2

4 4 1

4

1 2 1

2 3 1

3 4 1

4 1 1

5 7 2

3 1

1 2 1

2 3 1

3 1 1

4 1 2

4 2 2

5 1 2

5 2 2

**Output:**

2

3

3

**Notes On The Sample Data**

One solution to the first test cases uses communicating pairs (1,2) and (2,3). We can't have 3 pairs since one of the frequencies would then be used by two devices. A solution for the second test case uses pairs (1,2), (2,3), and (3,4). All four pairs may not be used since there would be multiple ways for 1 and 3 to communicate (using either 2 or 4 as an intermediate device).

Finally, one solution to the last test case uses (1,2), (2,3), and (4,1). One cannot have 4 pairs for the following reason. We can't use all of (1,2),(2,3), and (3,1) since this creates more than one way for 1 and 2 to communicate. On the other hand, we can't use two of (4,1),(4,2),(5,1), or (5,2) since this would violate the bound on the frequencies.

O	<u>Octo</u>	01	1	
	<u>ber</u>	Oct	0	
C	<u>201</u>	201	d	3
T	<u>0</u>	0	a	6
10	<u>Chal</u>	15:	y	5
	<u>leng</u>	00:	s	
	<u>e</u>	00		

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Count palindromes

**Count palindromes** Problem Code: COUNTPAL

Each palindrome can be always created from the other palindromes, if a single character is also a palindrome. For example, the string "bobseesanna" can be created by some ways:

- \* bobseesanna = bob + sees + anna
- \* bobseesanna = bob + s + ee + s + anna
- \* bobseesanna = b + o + b + sees + a + n + n + a

...

We want to take the value of function CountPal(s) which is the number of different ways to use the palindromes to create the string s by the above method.

**Input**

The string s

**Output**

The value of function CountPal(s), taking the modulo of 1 000 000 007  
 $(10^9+7)$

## Limitations

$0 < |s| \leq 1000$

## Example

**Input:**

bobseesanna

**Output:**

18

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Logging Game

## Logging Game Problem Code: LOGGERS

Logging can be a very mundane job, but Alice and Bob have devised a game to help them pass the time. They take turns choosing a log, and cutting it into 2 smaller logs. The sum of the lengths of the 2 logs equals the length of the original log. The only restriction is that neither of the resulting logs may be shorter than 1 meter in length (but exactly 1 meter is fine). In other words, non-integer lengths are allowed. Alice makes the first cut, and the first logger who cannot make a legal cut loses.

### Input

Input begins with an integer  $T$ , the number of test cases (less than 450).  $T$  test cases follow, each on its own line. Each test case begins with an integer  $N$ , the number of logs at the start of the game.  $N$  integers follow, giving the initial lengths of the logs. There are at most 7 logs, and the total length of the logs will not exceed 250 meters. Note that the initial lengths of the logs are integers, but logs may be cut to non-integer lengths.

### Output

For each test case, output a single line containing the name of the winner of the game, assuming both loggers choose their cuts optimally.

### Sample Input

1 2

2 3 4

3 7 8 9

## Sample Output

Alice

Alice

Bob

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Integer Combinations Of Vectors

## Integer Combinations Of Vectors Problem Code: INTCOMB

You are given a collection of  $n$  integer-valued vectors  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$  (i.e. integer arrays) of some fixed dimension. An integer combination of these vectors is a vector of the form  $\mathbf{v} = c_1\mathbf{v}_1 + \dots + c_n\mathbf{v}_n$  where each  $c_i$  is an integer. Here, the  $j$ 'th component  $\mathbf{v}[j]$  is precisely  $c[1]\mathbf{v}_1[j] + \dots + c[n]\mathbf{v}_n[j]$ .

Define the length of a vector  $\mathbf{x} = (x[1], x[2], \dots, x[n])$  to be the square root of  $x[1]^2 + x[2]^2 + \dots + x[n]^2$ . Your task is to find a short integer combination of vectors  $\mathbf{v}_1, \dots, \mathbf{v}_n$  whose length is non-zero.

### Input

The first line contains a single integer  $T \leq 30$  indicating the number of test cases to follow. Each test case begins with two integers  $n$  and  $d$ , both between 1 and 100, where  $n$  is the number of vectors in the test case and  $d$  is their common dimension.

Then  $n$  lines follow, each containing  $d$  integers. The  $j$ 'th number on the  $i$ 'th line is the  $j$ 'th component of vector  $\mathbf{v}_i$  (i.e.  $\mathbf{v}_i[j]$ ). Every such integer is between -100 and 100 (inclusive). None of the input vectors will be a zero vector. This means that for each  $i$ , at least one  $j$  is such that  $\mathbf{v}_i[j] \neq 0$ .

Test cases are separated by a single blank line including a blank line preceding the first test case.

### Output

For each test case, you are to output a single line consisting of  $n$  integers where the  $i$ 'th integer is the value of the coefficient  $c[i]$  in some integer combination of the input vectors. The value must satisfy  $-100 \leq c[i] \leq 100$ .

Furthermore, the integer combination of the input vectors using these coefficients must produce a non-zero vector. That is, the vector must have positive length.

## Example

**Input:**

2

2 1

14

35

4 5

1 4 1 2 5

1 2 3 4 5

0 1 2 3 7

1 -2 3 -4 5

**Output:**

-3 1

-1 0 1 0

## Scoring

The score for a test case is the length of the integer combination as given by the output coefficients divided by the length of the shortest input vector. The total score for the entire data set is simply the sum of the scores of the individual test cases.

## Example

The integer combination represented by the output in the first test case is  $-3*(14) + 1*(35) = (-7)$ . The length of the vector (-7) is simply 7. The length of the shortest vector in the input is 14 so the score for this case is 0.5.

The integer combination represented by the output for the second test case is  $-1*(1,4,1,2,5) + 1*(0,1,2,3,7) = (-1,-3,1,1,2)$ . The length of this is the square root of  $1^2 + 3^2 + 1^2 + 1^2 + 2^2 = 16$ , so the length is 4. The shortest vector in the input is the first one having length equal to the square root of  $1^2 + 4^2 + 1^2 + 2^2 + 5^2 = 47$  which is roughly 6.855. Thus, the score for the second case is  $4/6.855 = 0.5835$ .

## Input Distribution

Some of the test data is randomly generated by simply selecting values for the components of the vectors uniformly at random in the range -100 to 100 and keeping the vector if it is non-zero.

Other test data is generated by generating  $n-1$  of the vectors as above, computing a random integer combination of these vectors and then creating the final vector so that subtracting it from the random integer combination results in a very short vector. This final vector may be any of the  $n$  vectors in the input.

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Reduce string

## Reduce string Problem Code: STREDUC

Given a string  $s$  of length  $l$ , and a set  $S$  of  $n$  sample string(s). We do reduce the string  $s$  using the set  $S$  by this way:

- Wherever  $S_i$  appears as a consecutive substring of the string  $s$ , you can delete (or not) it.
- After each deletion, you will get a new string  $s$  by joining the part to the left and to the right of the deleted substring.

## Request

By that way, try to reduce the given string  $s$  to get a new string of minimum length. You can do delete for unlimited times.

## Input

- The first line contains the string  $s$ .
- The second line contains the integer  $n$ .
- Within the last  $n$  lines, the  $i$ -th line contains the string  $S_i$ .

## Output

Output on a single line an integer which is the minimum length found.

## Example

**Input:**  
aaabcccd

3

abc

ac

aaa

**Output:**  
2

## Limitations

- $0 < l \leq 250$
- $0 < n \leq 30$
- $0 < |S_i| \leq 20$

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Get a Clue

## Get a Clue Problem Code: CLUE

A group of 5 logicians are playing a game of Clue (also known as Cluedo). The game is played with 18 cards: 6 "suspect" cards, 6 "weapon" cards, and 6 "room" cards (the original game has 9 room cards, but the logicians have removed 3 in order for everyone to receive the same number of cards). At the beginning of the game, 1 of each type of card is chosen at random and placed in an envelope, whose contents are kept secret. The remaining cards are then dealt out randomly to the players, 3 cards each. The goal of the game is to be the first player to determine which cards are in the envelope. To make things more interesting, the logicians have modified the rules to the game as follows:

First of all, the cards are each given a unique number: The "suspect" cards are given numbers 1-6, the "weapon" cards numbers 7-12, and the "room" cards 13-18. At the start of play, each logician announces the sum of his cards to the group. Then, starting with the first logician and continuing in a circle, each logician will either pass, or declare himself the winner by announcing the contents of the secret envelope. After the fifth logician's turn, it becomes the first logician's turn again. The logicians are very smart, and will never pass if it's possible for them to deduce the contents of the envelope, nor will they ever attempt to guess the contents of the envelope when they don't know with certainty. Furthermore, all of the logicians are aware of the fact that none of them will pass if it's possible to win.

Your task is to determine, given the distribution of the cards, which logician will win (if any).

**Input:**

Input begins with an integer  $T$  (about 100), the number of test cases. Each test case consists of 5 lines of 3 integers each, indicating the cards held by each of the 5 logicians. Each integer will be between 1 and 18, inclusive, and exactly one integer from each of the ranges 1-6, 7-12, and 13-18 will be absent from each test case. The first line corresponds to the logician whose turn is first, the second line to the logician whose turn is second, and so on. Each test case is separated by a blank line

**Output:**

For each test case output a single integer, on a line by itself, indicating the (1-based) index of the logician who wins the game, or -1 if the game has no winner (i.e. continues forever).

**Sample Input:**

2

6 10 16

3 2 11

8 9 12

15 17 18

4 5 14

1 4 5

7 10 11

3 6 9

12 14 15

16 17 18

**Sample Output:**

1

## Explanation:

In the first example, the cards in the envelope are 1, 7, and 13. All of the logicians immediately know this, however the first logician wins because it must be one's turn in order to win.

In the second example, the cards in the envelope are 2, 8, and 13. Note that initially the second player does not have enough information to determine which cards are in the envelope. However, the fact that the first player passes is enough for the second player deduce the first player's cards (if the first player had been holding 2 3 5, for example, he would have won instead of pass), which in turn allows him to determine the secret cards and win.

[Home](#) » [Compete](#) » [October 2010 Challenge](#) » Post Office

## Post Office Problem Code: POST

It's the 1st of the month again, which means it's time for Dave to pay his bills. Dave needs to go to the post office to mail his rent cheque, and wants to take the shortest route possible. Dave wonders how many different shortest routes he can take.

Dave lives at the southwest corner of town (0,0) and the post office is at the northeast corner of town (W,H). It takes Dave exactly one minute to move from (x, y) to (x, y+1), (x, y-1), (x+1, y) or (x-1, y). Dave cannot move further south or west than his house, nor can he move further north or east than the post office (that is, Dave's x coordinate must stay between 0 and W, and his y coordinate must stay between 0 and H at all times). Additionally, there are N intersections that are being worked on by construction crews, and Dave must avoid these intersections.

The number of shortest routes may be very large, so print the result modulo 1000000037. If it is impossible for Dave to reach the post office, the answer is 0.

### Input:

Input begins with an integer T, the number of test cases. Each test case begins with 3 integers W, H, N. N lines follow, each containing 2 integers ( $x_i, y_i$ ), the coordinates of an intersection being worked on by a construction crew. Intersections are guaranteed to be unique, and neither (0,0) nor (W,H) will be under construction. A blank line separates each test case.

### Output:

For each test case, output a line containing the number of shortest paths to the post office, modulo 1000000037.

**Sample input:**

3

2 2 1

1 1

1 1 2

0 1

1 0

7 1 2

1 0

4 1

**Sample output:**

2

0

6

**Constraints**

$1 \leq T \leq 30$   
 $1 \leq W, H \leq 10^7$   
 $0 \leq x_i \leq W$   
 $0 \leq y_i \leq H$   
 $0 \leq N \leq \max(100, \min(W, H, 1000))$

CO	Th	15	2	1
OK	eo	Oc	ho	€
03	ct	t	urs	0

ob	10	mi
er	21:	nut
C	30:	es
oo	00	
k-		
O-		
f-		
A-		
C-		
M-		
I-		
P-		
C-		
W-		
a-		
m-		
U-		
p-		
IIT-		
-		
K)		

[Home](#) » [Compete](#) » [The October Cook-Off \(ACM ICPC Warm Up IIT-K\)](#) » [Cooling Pies](#)

## Cooling Pies Problem Code: COOLING

The chef has just finished baking several pies, and it's time to place them on cooling racks. The chef has exactly as many cooling racks as pies. Each cooling rack can only hold one pie, and each pie may only be held by one cooling rack, but the chef isn't confident that the cooling racks can support the weight of the pies. The chef knows the weight of each pie, and has assigned each cooling rack a maximum weight limit. What is the maximum number of pies the chef can cool on the racks?

### Input:

Input begins with an integer  $T \leq 30$ , the number of test cases. Each test case consists of 3 lines. The first line of each test case contains a positive integer  $N \leq 30$ , the number of pies (and also the number of racks). The second and third lines each contain exactly positive  $N$  integers not exceeding 100. The integers on the second line are the weights of the pies, and the integers on the third line are the weight limits of the cooling racks.

### Output:

For each test case, output on a line the maximum number of pies the chef can place on the racks.

### Sample input:

2

3

10 30 20

30 10 20

5

9 7 16 4 8

8 3 14 10 10

### Sample output:

3

4

[Home](#) » [Compete](#) » [The October Cook-Off \(ACM ICPC Warm Up IIT-K\)](#) » Birthday Candles

## Birthday Candles Problem Code: CANDLE

The chef is preparing a birthday cake for one of his guests, and his decided to write the age of the guest in candles on the cake. There are 10 types of candles, one for each of the digits '0' through '9'. The chef has forgotten the age of the guest, however, so doesn't know whether he has enough candles of the right types. For example, if the guest were 101 years old, the chef would need two '1' candles and one '0' candle. Given the candles the chef has, your task is to determine the smallest positive integer that cannot be represented with those candles.

### Input:

Input will begin with an integer  $T \leq 100$ , the number of test cases. Each test case consists of a single line with exactly 10 integers, each between 0 and 8, inclusive. The first integer of each test case represents the number of '0' candles the chef has, the second integer represents the number of '1' candles the chef has, and so on.

### Output:

For each test case, output on a single line the smallest positive integer that cannot be expressed with the given candles.

### Sample input:

3

2 1 1 4 0 6 3 2 2 2

0 1 1 1 1 1 1 1 1 1

2 2 1 2 1 1 3 1 1 1

### Sample output:

4

10

22

[Home](#) » [Compete](#) » [The October Cook-Off \(ACM ICPC Warm Up IIT-K\)](#) » Product of Digits

## Product of Digits Problem Code: DIGITS

The chef may be losing his mind. He has demanded you solve the following task: given an integer  $P$ , find the smallest positive integer whose product of digits, modulo 1000000007, is  $P$ . You can't imagine what this has to do with cooking, but you'd better do what he says or you won't get to sample his culinary delights.

### Input:

Input will begin with a positive integer  $T \leq 15$ , the number of test cases.  $T$  lines follow, each containing an integer  $P$  between 0 and 1000000006, inclusive.

### Output:

For each test case, print on a single line the smallest positive integer whose product of digits, modulo 1000000007, is  $P$ . It is guaranteed that such an integer exists with fewer than 700 digits.

### Sample input:

4

16

486

220703118

0

**Sample output:**

28

699

555555555555

10

[Home](#) » [Compete](#) » [The October Cook-Off \(ACM ICPC Warm Up IIT-K\)](#) » Two Chefs

## Two Chefs Problem Code: TWOCHefs

Two chefs are working in a kitchen with two ovens. Each chef has a list of dishes to prepare. Each chef must complete the dishes on his list, and in the order in which they appear on his list. The chefs have already prepared the items, and need only cook them. Each oven can only cook one dish at a time, and once a dish enters an oven it must remain in the oven for the full cooking time. A dish is considered completed when it is done cooking. That is, dishes need not enter the ovens in order, but must finish cooking in order (it is okay for 2 dishes from the same chef to finish cooking simultaneously). The chefs cannot clean up and go home until both of them have completed all their dishes, and they want to go home as soon as possible. Your task is to determine the minimal amount of time it will take the chefs to complete all dishes.

**Input:**

Input consists of 3 lines. The first line contains 2 integers  $N_1$  and  $N_2$ , the number of dishes to be completed by the first and second chefs, respectively. The second line contains  $N_1$  integers giving the cooking times of the first chef's dishes, in the order in which they must be completed. The third line contains  $N_2$  integers giving the cooking times of the second chef's dishes, in the order in which they must be completed.

**Output:**

Output on a single line the minimum amount of time (in minutes) for both chefs to complete all dishes.

**Example****Input 1:**

4 4

20 20 25 25

40 10 10 40

**Ouput 1:**

100

One way to finish in 100 minutes is for the one chef to cook all his dishes in one oven, and the other chef to cook all his dishes in the other oven. Note that although the dishes could be cooked in only 95 minutes, they cannot be completed in the correct order in only 95 minutes.

**Input 2:**

3 1

15 16 21

10

**Output 2:**

31

One oven will cook the 10 minute and 21 minute dishes, the other the 15 and 16 minute dishes. The first chef's third dish enters an oven before his second dish, but the dishes finish cooking at the same time and therefore are still considered to have completed in the correct order.

**Constraints:**

$1 \leq N_1 \leq 50$

$1 \leq N_2 \leq 50$

All cooking times are between 1 and 60 minutes, inclusive.

[Home](#) » [Compete](#) » [The October Cook-Off \(ACM ICPC Warm Up IIT-K\)](#) » The Perfect Chocolate Candy

## The Perfect Chocolate Candy Problem Code: CANDY

The chef is a fan of candies with chocolate and caramel, and has devoted much of his life to finding the perfect ratio of chocolate to caramel. He recently discovered the perfect ratio, but to his dismay, none of his favourite candy shoppes sell candies with exactly that ratio. So the chef crafted a plan to buy several candies, and melt them together into a larger candy so that the resulting candy will have the perfect ratio. How many candies will he have to buy? The chef may buy multiple candies of the same type.

### **Input:**

Input begins with an integer  $N$ , the number of different candies available at the various shoppes.  $N$  lines follow, each containing 2 integers  $\text{Chocolate}_i$  and  $\text{Caramel}_i$  indicating the grams of chocolate and caramel, respectively, contained in the  $(i)$ th candy. Finally, there is a line containing 2 integers  $\text{desiredChocolate}$  and  $\text{desiredCaramel}$ , indicating the chef's desired ratio of chocolate to caramel.

### **Output:**

For each test case, output on a single line the minimum number of candies the chef will have to buy. If it is impossible to produce a candy with the desired ratio, output -1 instead.

### **Example**

#### **Input 1:**

3

4 5

2 4

4 1

1 1

#### **Output 1:**

3

In this example, the chef purchases one of each candy and combines them to form a candy with 8 grams chocolate and 8 grams caramel.

**Input 2:**

3

2 3

4 6

10 15

5 8

**Output 2:**

-1

**Constraints:**

$1 \leq N \leq 15$

$1 \leq \text{Chocolate}_i \leq 500$

$1 \leq \text{Caramel}_i \leq 500$

$1 \leq \text{desiredChocolate} \leq 500$

$1 \leq \text{desiredCaramel} \leq 500$

For all candies,  $\text{Chocolate}_i/\text{Caramel}_i \neq \text{desiredChocolate}/\text{desiredCaramel}$

No two candies will be identical.

NOV10	<a href="#">November 2010 Challenge</a>	01 Nov 2010 15:00:00	10 days	307
-------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [November 2010 Challenge](#) » Randomly Testing Circuits

## Randomly Testing Circuits Problem Code: CIRCUITS

AND gates and OR gates are basic components used in building digital circuits. Both gates have two input lines and one output line. The output of an AND gate is 1 if both inputs are 1, otherwise the output is 0. The output of an OR gate is 1 if at least one input is 1, otherwise the output is 0.

You are given a digital circuit composed of only AND and OR gates where one node (gate or input) is specially designated as the output. Furthermore, for any gate G and any input node I, at most one of the inputs to G depends on the value of node I.

Now consider the following random experiment. Fix some probability  $p$  in  $[0, 1]$  and set each input bit to 1 independently at random with probability  $p$  (and to 0 with probability  $1 - p$ ). The output is then 1 with some probability that depends on  $p$ . You wonder what value of  $p$  causes the circuit to output a 1 with probability  $1/2$ .

## Input

The first line indicates the number of test cases to follow (about 100).

Each test case begins with a single line containing a single integer  $n$  with  $1 \leq n \leq 100$  indicating the number of nodes (inputs and gates) in the circuit. Following this,  $n$  lines follow where the  $i$ 'th line describes the  $i$ 'th node. If the node is an input, the line simply consists of the integer 0. Otherwise, if the node is an OR gate then the line begins with a 1 and if the node is an AND gate then the line begins with a 2. In either case, two more integers  $a, b$  follow, both less than  $i$ , which indicate that the outputs from both  $a$  and  $b$  are used as the two input to gate  $i$ .

As stated before, the circuit will be such that no gate has both of its inputs depending on the value of a common input node.

Test cases are separated by a blank line including a blank line preceding the first test case.

## Output

For each test case you are to output a single line containing the value  $p$  for which the output of node  $n$  is 1 with probability exactly  $1/2$  if the inputs are independently and randomly set to value 1 with probability  $p$ . The value  $p$  should be printed with exactly 5 digits after the decimal.

## Example

**Input:**

4

1

0

3

0

0

1 1 2

3

0

0

2 1 2

5

0

0

0

2 1 2

1 3 4

**Output:**

0.50000

0.29289

0.70711

0.40303

[Home](#) » [Compete](#) » [November 2010 Challenge](#) » [Chefs Bad Day](#)

## Chefs Bad Day Problem Code: GRIDCHEF

### Statement

Chef is not having a good time these days. Just Yesterday he made terrible food and hence got beaten up. He has hence decided that he wont prepare food himself but instead collect food from hotels in the city.

However, he suffered injury on his head ( people can get really angry when served with bad food ) and is suffering from short term memory loss. He is standing in a city in the form of a grid where cell(i,j) of the grid contains a hotel which has  $G_{ij}$  amount of food to offer to the chef. Every second, Chef goes to one of the neighbors in the grid. ( Note: Please note that any cell(i,j) can have atmost 4 neighbors).

Unfortunately, due to short term memory loss, he throws away whatever he is given every alternate second. That is he throws away food given at  $t = 0$  , keeps food given at  $t = 1$  , throws away food given at  $t = 2$  , and so forth.

The grid also has some added constraints. No hotel ( or cell ) can be visited more than once. Since, Chef has weak memory, he must return to his kitchen ( which is also his starting point ) after collecting the food.

You have to help the Chef choose a path such that the amount of food collected is maximized [ The path should be such that no two cell in the path are same except for the start and end of the path which **have to be** the same ].

**The Chef can choose the kitchen ( starting point ) to be located in any of the four corners of the grid.**

### Input

The first line contains  $t$  (  $\leq 100$  ) which denotes the number of test cases. For each test case, the first line contains 2 space separated integers  $n,m$  which denote the dimension of the matrix. Then  $n$  lines follow, each containing  $m$  space separated integers.  $j$ th integer on the  $i$ th line denotes  $G_{ij}$

**Warning : Enormous Input/Output.**

### Output

For each test case, the first line of output should contain a number denoting the maximum amount of food the Chef can collect. The second line of output should contain the length of the path Chef takes (  $L$  ) for the mentioned amount of food. Then follow  $L$  lines, where the  $i$ th line contains 2 space separated integers ( 0 based ) denoting the cell number of the  $i$ th cell in the path.

If there are multiple such optimal paths, any path would suffice.

**Note :** The first and last cell of the path are the same ( The Chef returns to his starting point ) .

## Sample Input

1

2 2

2 3

3 2

## Sample Output

6

5

0 0

0 1

1 1

1 0

0 0

## Constraints

$1 < m * n \leq 100000$

$0 \leq G_{ij} \leq 10000$

[Home](#) » [Compete](#) » [November 2010 Challenge](#) » Bombing

## Bombing Problem Code: BOMBING

There are  $n+1$  ( $0 < n \leq 10^9$ , indexed  $0..n$ ) houses lying on a straight street. One day, they are bombed by an enemy. CodeChef is designing defence systems to protect the street from bombing. Each defense system can protect all houses in some interval, and a house can be protected by more than one system. A defence system can be also moved at any time. By agents, CodeChef knows the bombing schedule of the enemy. Each time they bomb a house and CodeChef wants to know how many systems are protecting that house.

## Input

- The first line contains two integers  $n, m$  ( $0 < m \leq 250\,000$ ).
- Each of the  $m$  following lines follows one of these:
  - $P\ u\ v$ : CodeChef adds a defence system protecting all houses from the  $u$ -th house to the  $v$ -th house ( $0 \leq u \leq v \leq n$ ). All systems will be indexed by order of appearance in the input, starting from 1.
  - $M\ i\ d$ : CodeChef moves the  $i$ -th system by  $d$  houses,  $d < 0$  means moving nearer the 0-th house, and  $d > 0$  means moving away (that is, if the  $i$ -th system covers houses  $u$  through  $v$ , it is moved to instead cover houses  $w=u+d$  through  $t=v+d$ ). After being moved the system will protect from the  $w$ -th house to  $t$ -th house where  $0 \leq w \leq t \leq n$ .
  - $B\ x$ : The enemy bombs the  $x$ -th house.

## Output

For each time of bombing by the enemy (by the appearance in the input), output on a single line a number of systems protecting that house.

## Example

**Input:**

7 5

P 1 4

P 3 5

B 3

M 2 1

B 3

**Output:**

2

1

## Balanced Walks Problem Code: BALANCED

Little Jack and little Jill are on a scavenger hunt you have devised. They are going to visit a set of locations and you are going to leave exactly one object at each location. There are precisely two different types of objects, say type A and type B. Furthermore, Jack only likes objects of type A and Jill only likes objects of type B so Jack will never collect a type B object and Jill will never collect a type A object.

Jack and Jill will travel together and visit these locations one at a time. If at any point during their journey one person has many more of their preferred item than the other, then the other will get jealous and they may fight. To help avoid such fights, you want to assign items to locations to minimize the maximum difference between the number of items they hold over the whole journey.

Specifically, if  $R = (r_1, r_2, \dots, r_n)$  is the order the locations will be visited and  $P$  is an assignment of an object to each location, then we define  $d_R(P, t)$  to be the difference between the number of objects Jack has and the number of objects Jill has after visiting the first  $t$  locations. Finally, let  $d_R(P)$  denote the maximum of  $d_R(P, t)$  over all times  $t$ . The goal would be to find an assignment  $P$  to minimize  $d_R(P)$ .

This sounds like an easy task, but there is one further complication. Both Jack and Jill have decided on an order they want to visit the items. Jack wants to follow order  $S$  but Jill wants to follow order  $T$ . It looks like they are a long way from resolving this dispute but you need to place the items now! To minimize the effect of the worst-case scenario, you want to place the items in some manner  $P$  to minimize the maximum of  $d_S(P)$  and  $d_T(P)$ .

### Input

The first line indicates the number of test cases to follow (at most 30). Each test case consists of three lines. The first contains a single integer  $n$  between 1 and 20,000. The next two lines each contain a permutation of the integers from 0 to  $n-1$  where consecutive integers are separated by a space. The first permutation, say  $S$ , is the order that Jack wants to visit the locations and the second permutation, say  $T$ , is the order that Jill wants to visit the locations.

### Output

The output for each test case consists of a single line. This line should contain a single string of length  $n$  consisting only of uppercase characters 'A' and 'B'. This string describes a placement  $P$  of items to locations where the  $i$ 'th character describes which item is at location  $i$  (the indices range from 0 to  $n-1$ ).

The placement  $P$  described by this string should minimize the maximum of  $d_S(P)$  and  $d_T(P)$ . If there are many strings describing optimum placements, then output the lexicographically least such string.

### Example

**Input:**

2

3

1 0 2

2 1 0

4

0 2 1 3

1 2 3 0

**Output:**

ABA

AABB

**Explanation of Sample Data**

In the first test case, following the first permutation 1 0 2 results in the sequence of objects being BAA while following the second permutation results in the sequence of objects being ABA. In either case, the maximum difference between the number of As seen and the number of Bs seen over the whole trip is at most 1.

In the second test case, following permutation 0 2 1 3 results in the sequence being ABAB whereas following permutation 1 2 3 0 results in the sequence being ABBA. Again, the maximum difference in either case is at most 1.

[Home](#) » [Compete](#) » [November 2010 Challenge](#) » Graph Challenge

# Graph Challenge

 Problem Code: GRAPHCH**Statement**

You are given a modified graph with N vertices and M edges. Each vertex is a rectangle [ dimension of each vertex may be different ]. Your task is to place these vertices in 2-d space such that :

- No two vertices overlap [ remember, they are rectangles ]
- All rectangles have their sides parallel to the axis.
- Rectangles cannot be rotated.

For every edge in the graph, add the manhattan distance between the centres of the vertex for each edge as the cost of a solution ( C ).

Now, there might be multiple ways to achieve this. So, you have to strive to keep C as low as possible.

## Input

First line contains  $t$  ( $\leq 100$ ) equal to the number of test cases. For each test case, the first line contains 2 space separated integers (  $N$  and  $M$  respectively ). Then  $M$  lines follow, each line containing 2 integers  $x$  and  $y$  ( $0 \leq x, y < N$ ,  $x \neq y$ ) denoting an edge between vertex  $x$  and  $y$ . Then follow  $N$  lines, where line number  $i$  contains 2 integers  $a$  and  $b$  denoting the dimension of the  $i$ th vertex [ Here,  $a$  denotes the length parallel to  $x$ -axis and  $b$  denotes the length parallel to  $y$ -axis ]

( **Note** : If the same pair  $x, y$  appears multiple times, it denotes multiple edges and hence, each pair contributes to the cost ).

**NOTE** : Please note that all solutions will be tested on another set of test data **after the contest** which will follow the same pattern for test case generation ( as mentioned in 'Test Case Generation' section ). **The final score for a solution will be the score of solution on latter test data.**

## Output

For each test case, print  $N$  lines , each containing 2 floating point numbers  $X$  and  $Y$ , denoting the co-ordinates of the centre of vertex  $i$ .

**Note** :  $-10^9 \leq X, Y \leq 10^9$  . Solutions not following the mentioned constraints will be adjudged as wrong answer.

**Please note that the answers may not be optimal**

## Scoring

The Score for a solution =  $(C+1)$ .

Your total score is the sum of your score for all the test cases.

You have to keep the score as low as possible.

## Constraints

$2 \leq N \leq 50$

$1 \leq M \leq 200$  [ The edges are randomly generated ]

$1 \leq a, b \leq 100000$

There can be multiple edges between a pair of vertices.

Additionally for 50% of the cases  $2 \leq N \leq 20$ . Also, for about 50% of cases, all vertices are squares [  $a = b$  ] of same size.

## Test Case Generation

There are about 100 cases in total. For first 50 cases, the number of vertex (  $n$  ) is chosen as a random number in the interval [3,20]. The number of edges (  $m$  ) is chosen as a random number in the range  $[2, n*(n-1)/2]$ . Then all edges are chosen as random pair of integers. All vertex are given the same dimension with the probability 0.5 else they are given a random dimension in  $[1,100000] * [1,100000]$ .

For other 50 cases,  $n$  is chosen as a random integer in the interval [3,50].  $m$  is chosen randomly from  $[2, \min(n*(n-1)/2, 200)]$ . Rest of the procedure remains the same.

## Sample Input

1

2 1

0 1

2 2

2 2

## Sample Output

2 2

10 3

**Score for sample output : 10.0 ( Better answers may be possible )**

[Home](#) » [Compete](#) » [November 2010 Challenge](#) » Renting Spare Oven Time

## Renting Spare Oven Time Problem Code: OVENTIME

The Chef has thought of another way to generate revenue for his restaurant. He has a large oven to bake his goods, but he has noticed that not all of the racks are used all of the time. If a rack is not used, then the Chef has decided to rent it out for others to use. The

Chef runs a very precise schedule; he knows in advance exactly how many racks will be free to rent out at any given time of the day.

This turns out to be quite a profitable scheme for the Chef. Apparently many people were interested in the idea of renting space in an industrial oven. The Chef may need to turn down some of the requests because of lack of oven space.

Each request comes in the form of a start time  $s$ , an end time  $e$ , and a value  $v$ . This means that someone has requested to use a single rack in the oven from time  $s$  until time  $e-1$  (i.e. the item is removed just before time  $e$  so another request may use that rack starting at time  $e$ ). If this request is satisfied, the Chef receives  $v$  dollars in payment.

Help the Chef determine the maximum possible profit he can receive from the collection of requests. That is, find the largest total profit of a subset  $S$  of requests such that at any point in time the number of requests in  $S$  that require a rack during that time does not exceed the number of free racks at that time.

## Input

The first line of input contains a single integer indicating the number of test cases  $T \leq 50$ . Each test case begins with two integers  $n, m$  with  $1 \leq n \leq 100$  and  $1 \leq m \leq 50$ . This means there are  $n$  requests to process and the oven is available from time 0 to time  $m$ .

Each of the following  $n$  lines describes a request and consists of three integers  $s$ ,  $e$ , and  $v$ . The numbers satisfy  $0 \leq s < e \leq m$  and  $1 \leq v \leq 1000$ . This means that the request requires use of a single rack in the oven during time units  $s, s+1, \dots, e-1$  and will pay  $v$  dollars if the request is satisfied.

Finally, a line follows with  $m$  integers  $c_0, c_1, \dots, c_{m-1}$ , each between 1 and 25. These indicate that at each time  $i$ ,  $0 \leq i < m$ , there are  $c_i$  racks available to rent.

Test cases are separated by a single blank line including a blank line preceding the first test case.

## Output

The output for each test case consists of a single integer indicating the maximum total value the Chef can receive from a subset of requests which can all be satisfied. Recall that we say a subset  $S$  of requests can be satisfied if at any time  $0 \leq t < m$  that the number of requests in  $S$  that require the oven at time  $t$  does not exceed  $c_t$ .

## Example

**Input:**

1

4 4

0 1 2

1 2 2

2 4 2

0 3 5

2 1 1 1

**Output:**

7

## Explanation of Sample Output

In the given test case, a profit of 7 can be obtained by satisfying the first and last request in the input. Time 0 has 2 free racks and both are used by these requests. Times 1 and 2 have only 1 free rack but these times are not required by the first request so there are enough racks to accommodate the requests at these times.

Now, the last time 3 has a free rack that is not being used. The second last request requires a rack at this time, but it cannot be added because it also requires a rack at time 2, which is already being used by the last request.

COOK04	<a href="#">The November CookOff</a>	27 Nov 2010 21:30:00	2 hours 30 minutes	248
--------	--------------------------------------	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [The November CookOff](#) » [Cutting Recipes](#)

## Cutting Recipes Problem Code: RECIPE

The chef has a recipe he wishes to use for his guests, but the recipe will make far more food than he can serve to the guests. The chef therefore would like to make a reduced version of the recipe which has the same ratios of ingredients, but makes less food. The chef, however, does not like fractions. The original recipe contains only whole numbers of ingredients, and the chef wants the reduced recipe to only contain whole numbers of ingredients as well. Help the chef determine how much of each ingredient to use in order to make as little food as possible.

### Input

Input will begin with an integer  $T$ , the number of test cases. Each test case consists of a single line. The line begins with a positive integer  $N$ , the number of ingredients.  $N$  integers follow, each indicating the quantity of a particular ingredient that is used.

## Output

For each test case, output exactly  $N$  space-separated integers on a line, giving the quantity of each ingredient that the chef should use in order to make as little food as possible.

## Constraints

$T \leq 100$

$2 \leq N \leq 50$

All ingredient quantities are between 1 and 1000, inclusive.

## Sample Input 1

3

2 4 4

3 2 3 4

4 3 15 9 6

## Sample Output 1

1 1

2 3 4

1 5 3 2

[Home](#) » [Compete](#) » [The November CookOff](#) » One Dimensional Game of Life

## One Dimensional Game of Life Problem Code: LIFE

In Conway's Game of Life, cells in a grid are used to simulate biological cells. Each cell is considered to be either alive or dead. At each step of the simulation each cell's current status and number of living neighbors is used to determine the status of the cell during the following step of the simulation.

In this one-dimensional version, there are  $N$  cells numbered 0 through  $N-1$ . The number of cells does not change at any point in the simulation. Each cell  $i$  is adjacent to cells  $i-1$  and  $i+1$ . Here, the indices are taken modulo  $N$  meaning cells 0 and  $N-1$  are also adjacent to each other. At each step of the simulation, cells with exactly one living neighbor change

their status (alive cells become dead, dead cells become alive). For example, if we represent dead cells with a '0' and living cells with a '1', consider the state with 8 cells:

01100101

- Cells 0 and 6 have two living neighbors.
- Cells 1, 2, 3, and 4 have one living neighbor.
- Cells 5 and 7 have no living neighbors.

Thus, at the next step of the simulation, the state would be:

00011101

Given some state of the game, your task is to determine the state immediately preceding it. In some cases there may be more than one answer or no possible answer.

## Input

Input will begin with an integer  $T < 100$ , the number of test cases. Each test case consists of a single line, with between 3 and 50 characters, inclusive. Each character will be either '0' or '1'. Each '0' represents a dead cell, and each '1' represents an alive cell.

## Output

For each test case, output the state of the game that precedes the given state. If there is no possible solution, print "No solution" (quotes for clarity only). If there are multiple possible solutions, print "Multiple solutions" (quotes for clarity only).

## Sample Input

4

00011101

000

000001

11110

## Sample Output

01100101

Multiple solutions

No solution

10010

[Home](#) » [Compete](#) » [The November CookOff](#) » N Knights Problem

## N Knights Problem Problem Code: KNIGHTS

Dave recently mastered the problem of placing N queens on a chessboard so that no two queens attack each other. Now he wants to see how many knights he can place on a chessboard so that no two knights attack each other. Normally this would be a simple task, but some of the squares of the chessboard have been marked as unusable and hence cannot have a knight placed on them.

Recall that a knight can attack another knight if their vertical distance is 2 and their horizontal distance is 1, or if their vertical distance is 1 and their horizontal distance is 2. Only one knight may be placed on each square of the chessboard

### Input

The first line of input contains an integer T ( $0 < T \leq 50$ ), the number of test cases to follow.

Each test case will begin with a line containing 2 integers M and N ( $0 < M, N \leq 30$ ), denoting the number of rows and columns, respectively. M lines follow, each containing exactly N characters. The j-th character of the i-th line is '.' if a knight may be placed in the j-th column of the i-th row, and '#' if the square is unusable.

### Output

For each test case, output on a single line the maximum number of knights that may be placed on the chessboard such that no two attack each other.

### Sample input:

```
2
2 4
....
....
5 5
..#..
#..#.
```

```
##...
```

```
...##
```

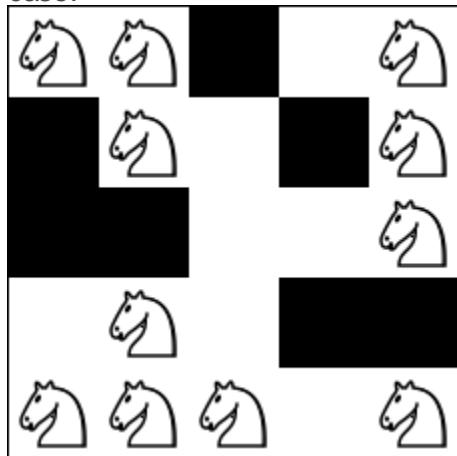
```
.....
```

### Sample output:

```
4
```

```
11
```

The following image represents the chessboard and a possible solution to the second test case:



[Home](#) » [Compete](#) » [The November CookOff](#) » Choosing Cook Off Problems

## Choosing Cook Off Problems Problem Code: COOKOFF

Dave is having trouble choosing problems for the upcoming cook-off. The admins will be mad at him if he doesn't select problems with a gradually increasing level of difficulty. To make his task easier, Dave has assigned a difficulty rating to each problem he has written. A difficulty gap is defined as the absolute difference in difficulty rating between 2 problems, where there is no other problem with a difficulty rating between them. For example, if problems are chosen with difficulty ratings of 4, 7, 13, and 19, then the difficulty gaps are 3, 6, and 6. Dave has already chosen 2 problems for the cook-off, and wants to choose the remaining problems in a way that minimizes the largest difficulty gap.

### Input

Input begins with an integer  $T$ , the number of test cases. Each test case begins with 4 integers  $N$   $M$   $C_1$   $C_2$  on a line.  $N$  is the number of problems to choose from (not including

the two already chosen problems).  $M$  is the number of problems that must be chosen (in addition to the two that have already been chosen).  $C_1$  and  $C_2$  are the difficulty ratings of the 2 problems that have already been chosen. Following this is a line with exactly  $N$  integers, giving the difficulty ratings of the remaining problems. Again, the 2 already chosen problems are not included in the values of  $N$  and  $M$ . A blank line separates each test case, including a blank line after the number of test cases.

## Output

For each test case, output on a single line the minimum possible value of the largest difficulty gap.

## Sample Input:

3

3 1 0 100

16 45 61

5 0 50 80

20 40 60 80 100

6 3 0 100

21 64 34 55 64 89

## Sample Output:

55

30

34

## Constraints

$T \leq 250$

$0 \leq M \leq N \leq 1000$

All difficulty ratings are between 0 and 1000000, inclusive.

[Home](#) » [Compete](#) » [The November CookOff](#) » Dinner Party Socializing

## Dinner Party Socializing Problem Code: SOCIAL

The chef is having a dinner party. He has  $N$  chairs and has invited  $N$  guests. The chef knows that if the guests are left to their own devices, they tend to sit in the same chairs and socialize with the same people all night. To prevent this, the chef has developed a plan to help people socialize. He will assign each chair a follow-up chair. At predetermined intervals during the party, the chef will ring a bell, instructing all guests to move from their current chair to its follow-up chair.

The chef will assign follow-up chairs randomly, with the restriction that no chair will be its own follow-up chair, and no two chairs will have the same follow-up chair. That is, the chef randomly chooses one arrangement out of all assignments satisfying the two conditions. The chef wonders, after a certain number of ringings of the bell, what the expected number of guests who will be back in their original chairs will be.

### Input

Input will begin with an integer  $T$ , the number of test cases. Each test case consists of a single line with 2 integers  $N$  and  $R$ , the number of chairs and number of ringings, respectively.

### Output

For each test case, output on a single line the expected number of guests who will be back in their original seats after exactly  $R$  ringings, rounded to 5 decimal places.

### Sample Input

4

2 1

2 2

4 2

5 3

### Sample Output

0.00000

2.00000

1.33333

1.36364

## Constraints

$T \leq 300$

$2 \leq N \leq 50$

$1 \leq R \leq 1000000$

DEC10	<a href="#">December 2010 Challenge</a>	01 Dec 2010 15:00:00	10 days	367
-------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [December 2010 Challenge](#) » A String Game

## A String Game Problem Code: ASTRGAME

Teddy and Tracy like to play a game based on strings. The game is as follows. Initially, Tracy writes a long random string on a whiteboard. Then, each player starting with Teddy makes turn alternately. Each turn, the player must erase a contiguous substring that exists in the dictionary. The dictionary consists of  $N$  words.

Of course, the player that can't erase any substring in his turn loses the game, and the other player is declared the winner.

Note that after a substring  $R$  is erased, the remaining substring becomes separated, i.e. they cannot erase a word that occurs partially to the left of  $R$  and partially to the right of  $R$ .

Determine the winner of the game, assuming that both players play optimally.

### Input

The first line contains a single integer  $T$ , the number of test cases.  $T$  test cases follow. The first line of each testcase contains a string  $S$ , the string Tracy writes on the whiteboard. The next line contains a single integer  $N$ .  $N$  lines follow. The  $i$ -th line contains a single string  $w_i$ , the  $i$ -th word in the dictionary.

### Output

For each test case, output a single line containing the name of the winner of the game.

### Example

**Input:**

3

codechef

2

code

chef

foo

1

bar

mississippi

4

ssissi

mippi

mi

ppi

**Output:**

Tracy

Tracy

Teddy

**Constraints**

- $1 \leq T \leq 5$
- $1 \leq N \leq 30$
- $1 \leq |S| \leq 30$
- $1 \leq |w_i| \leq 30$
- $S$  and  $w_i$  contain only characters 'a'-'z'

## Byteknights and Byteknaves Problem Code: BYTEISLE

A long school holiday has come, and you decided to visit the famous Byte Island. You know there are only two types of Bytelandians: Byteknights and Byteknaves. A Byteknight always tells the truth, whereas a Byteknav always lies.

It is known that there are  $N$  Bytelandians in the island, and now you meet all of them. You are curious about their types. Because you are a smart logician, you don't want to ask them questions that immediately reveal their types (that's too boring). Instead, to each Bytelandian you ask, "How many Byteknights are there here?"

To your surprise, they also don't answer your questions directly. Instead, the  $i$ -th Bytelandian answers of the form "The number of Byteknights here is between  $a_i$  and  $b_i$ , inclusive". You record all answers in your pocket note.

Now that you have collected all information you need, determine the type of each Bytelandian.

### Input

The first line contains a single integer  $T$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains a single integer  $N$ .  $N$  lines follow. The  $i$ -th line contains two integers  $a_i$  and  $b_i$ .

### Output

For each test case, output two lines. In the first line, output a single integer indicating the number of different solutions, modulo 1000000007. In the next line, output the lexicographically smallest solution. A solution is a string consisting of  $N$  characters, where the  $i$ -th character of the string is '1' if the  $i$ -th Bytelandian is a Byteknight, or '0' in case of a Byteknav. It is guaranteed that there is at least one valid solution.

### Example

**Input:**

```
3
1
0 1
4
1 4
```

2 4

3 4

4 4

3

1 2

0 0

1 3

**Output:**

1

1

5

0000

1

101

## Constraints

- $1 \leq T \leq 5$
- $1 \leq N \leq 50000$
- $0 \leq a_i \leq b_i \leq N$

[Home](#) » [Compete](#) » [December 2010 Challenge](#) » The Hungry Bear

## The Hungry Bear Problem Code: HBEAR

Little Bear had just come back from school. He was very hungry that time and wanted at least  $K$  units of honey to satisfy his hunger. He immediately went to a rectangular field of size  $N \times M$  to collect some honey. He wanted to collect honey only in a special sub-rectangle which has size  $S \times T$  where  $S \leq A$  and  $T \leq B$ , for some fixed  $A$  and  $B$ . He has  $Q$  queries, each query contains  $A, B$ . For each query, he wanted to know how many special sub-rectangles contain at least  $K$  units of honey.

## Input

The first line of the input contains integers  $N$ ,  $M$  and  $K$  ( $1 \leq N, M \leq 300$ ,  $1 \leq K \leq N \cdot M$ ). The next  $N$  lines contain  $M$  characters (either 'H' or '.'). 'H' means there is honey in that cell, while '.' means there is no honey in that cell. The next line contains an integer  $Q$  ( $1 \leq Q \leq 100\ 000$ ). The next  $Q$  lines contain integers  $A$  and  $B$  ( $1 \leq A \leq N$ ,  $1 \leq B \leq M$ ).

## Output

The output contains  $Q$  lines, each line contains the number of special sub-rectangles which satisfy the requirement. (The amount of honey is at least  $K$  inside the special sub-rectangle)

## Example

Input:

5 5 4

H.H..

..H..

H...H

HHHH.

HH..H

2

1 1

2 3

Output:

0

4

### Explanation for 2nd query

#: means the chosen cell

The possibilities are:

1. Size 2 x 3

H.H..

..H..

###.H

##H..

HH..H

2. Size 2 x 3

H.H..

..H..

H...H

##H..

###.H

3. Size 2 x 3

H.H..

..H..

H...H

H##..

H###H

4. Size 2 x 2

H.H..

..H..

H...H

##HH.

##..H

Note that in 2nd query, the possible special sub-rectangles are of size 1x1, 1x2, 1x3, 2x1, 2x2, 2x3. Also note that 2x3 is different from 3x2

[Home](#) » [Compete](#) » [December 2010 Challenge](#) » Delivering Bread

## Delivering Bread Problem Code: DELIVERY

Stores from all around the city purchase their bread from the Chef. Normally, the Chef has a delivery man who brings the bread from the Chef's kitchen to the stores each day. Unfortunately, the delivery man is ill today so the Chef must deliver the bread himself.

The Chef can only deliver bread to a store if he reaches it before the store closes for the day. To complicate matters more, the city is a mess of one way streets so it is difficult to determine a good route that visits as many stores as possible before they close.

You may assume it takes 0 time to deliver bread to a store once the Chef has reached the store. Thus, the total time it takes the Chef to deliver the bread is simply amount of time he spends driving. Finally, we note that all stores close at the same time in this city.

### Input

The first line consists of a single integer  $T$  indicating the number of test cases that will follow (at most 40). Each test case begins with a line consisting of three integers  $n, m, b$ . Here,  $n$  indicates the number of stores (including the Chef's bakery),  $m$  indicates the number of one-way streets and  $b$  is the number of time units until all stores close.

Then  $m$  lines follow, each consisting of three integers  $u, v, d$  with both  $u$  and  $v$  between 0 and  $n-1$  and  $d$  between 0 and 1,000,000,000. This means there is a one way street from store  $u$  to store  $v$  that takes precisely  $d$  units of time to travel. The input will be such that

there is at most one street from a store  $u$  to a store  $v$ , though there may be a one way street from  $u$  to  $v$  and one from  $v$  to  $u$ . Also, there is never a one way street from a store to itself. Finally, the Chef's bakery is always store 0 and this is where the Chef starts driving.

Bounds:  $1 \leq n \leq 300$ ,  $0 \leq m \leq 10,000$ , and  $0 \leq b \leq 1,000,000,000$ .

## Output

The output for each test case consists of a single line. This line should begin with a sequence of integers between 0 and  $n-1$  and end with -1. A sequence  $s_1, s_2, \dots, s_k$  followed by a -1 means the chef travels along a single one way street from store 0 (the bakery) to store  $s_1$ , then from store  $s_1$  to store  $s_2$ , and so on until store  $s_k$  is reached. That is, there should be a one way street from store 0 to store  $s_1$  and, for each  $0 < j < k$  there should be a one way street from store  $s_j$  to store  $s_{j+1}$ .

This should be done such that the total time taken along all streets should be at most  $b$ . Finally, since the Chef doesn't want the driving instructions to be too complicated the output should be such that  $k \leq 10,000$ . A store may appear multiple times in this output list.

The goal is to visit as many distinct stores as possible before they close. Note that the optimum answer is not required. This problem is scored and any output that meets the above specifications will be considered valid. The details on how the score is calculated is described below the sample data.

## Example

**Input:**

2

3 4 2

0 1 1

0 2 1

1 2 1

2 1 2

5 7 1 1

0 1 2

1 0 3

0 2 1

2 3 2

3 2 3

2 4 1

4 2 2

**Output:**

1 2 -1

1 0 2 3 -1

## Scoring

The score for each test case is simply the number of distinct stores visited by the Chef (recall the Chef starts at store 0 so it is counted even if it does not appear in the output list). The total score for a test file is then the sum of the scores of each test case. Finally, there are multiple input files and the displayed score will be the average of the scores of the individual test files.

## Explanation of Output

The output for the first test case indicates the Chef drives from 0, to 1, and then to 2. The total time spent is 2 units so the Chef reaches the last store just in time and the score for this test case is 3.

For the second test case, the Chef travels along the sequence of stores 0,1,0,2,3 before stopping which takes  $2+3+1+2=8$  time units. Thus, he reaches exactly 4 distinct stores so the score for this test case is 4.

## Test Data

Some test data is hand-crafted to defeat simple heuristics and some is randomly generated from a variety of distributions.

[Home](#) » [Compete](#) » [December 2010 Challenge](#) » Odd Binomial Coefficients

# Odd Binomial Coefficients Problem Code: ODDBIN

If  $P(x)$  is a polynomial in  $x$  with integer coefficients, let  $W(P(x))$  = number of odd coefficients of  $P(x)$ .

Given  $a_1, a_2, \dots, a_m$ , find  $W( (1+x)^{a_1} + (1+x)^{a_2} + \dots + (1+x)^{a_m} )$ .

## Input

First line contains  $TC$ , the number of test cases.

Each test case consists of a single line in the format:

$m a_1 a_2 \dots a_m$

## Limits

$1 \leq m \leq 15$

$0 \leq a_i < 2^{60}$

$1 \leq TC \leq 1000$

## Output

Output one line per test case, the value  $W( (1+x)^{a_1} + (1+x)^{a_2} + \dots + (1+x)^{a_m} )$ .

## Example

Input:

4

1 1

1 3

2 1 3

3 1 2 3

Output:

2

4

2

2

## Explanation

$(1+x) + (1+x)^3 = 2 + 4x + 3x^2 + x^3$ . Hence the output for "2 1 3" is 2. (2 odd coefficients)

$(1+x) + (1+x)^2 + (1+x)^3 = 3 + 6x + 4x^2 + x^3$ . Hence the output for "3 1 2 3" is 2.

[Home](#) » [Compete](#) » [December 2010 Challenge](#) » Even Coefficients

## Even Coefficients Problem Code: EVENPOLY

In this problem, you are given a polynomial  $P$  over multiple variables with integer coefficients and your task is simply to determine if all of the coefficients are even or not.

Unfortunately, you will not be given an explicit description of the polynomial. Rather, you will be given a square matrix whose entries are polynomials and the polynomial  $P$  is the determinant of this matrix.

Here, we are defining the determinant of a matrix by the usual cofactor expansion formula. That is, say the matrix  $M$  is of size  $d \times d$  and denote the polynomial located at row  $i$ , column  $j$  by  $M_{i,j}$  for any  $i,j$  between 1 and  $d$ . For a permutation  $X$  on the set  $\{1,2,\dots,d\}$ , let  $\text{sgn}(X)$  be  $(-1)^{\text{inv}(X)}$  where  $\text{inv}(X)$  is the number of pairs  $1 \leq i < j \leq d$  with  $X(i) > X(j)$  (i.e. the number of inversions of  $X$ ). Then the determinant of  $M$  is the sum, over all permutations  $X$ , of the products  $\text{sgn}(X)M_{1,X(1)}M_{2,X(2)}\dots M_{d,X(d)}$ .

## Input

The first line consists of a value  $k \leq 30$  indicating the number of test cases. Each test case begins with a single line containing an integer  $d$ ,  $1 \leq d \leq 40$  indicating the order of the matrix. Then,  $d$  lines follow where each contains  $d$  strings describing the polynomials. The  $j$ 'th string on the  $i$ 'th line describes entry  $M_{i,j}$  of the polynomial matrix. The  $d$  strings on a single line are separated by a single space.

Each polynomial is given by a string of length between 1 and 100. A monomial is either the string '0', or the string '1', or a string of lowercase characters 'a', 'b', ..., 'z'. A polynomial is then a sequence of monomials separated by '+' with no spaces. The monomials '0' or '1' are simply the respective constant polynomials and a monomial consisting of a string of

lowercase letters is simply the product of variables a, b, ..., z where the power of a letter, say L, is exactly the number of times it appears in the string. For example, "1+aba" is the string representing the polynomial  $1+a^2b$  and "0+xy+yx" represents the polynomial  $2xy$ .

## Output

The output for each test case consists of a single line containing the message "All Even" or "Some Odd" to indicate if all coefficients of the polynomial represented by the determinant of the matrix are even or if there is at least one odd coefficient, respectively.

## Example

### Input:

3

2

x+1 0

0 x

3

x y z

y z x

xx+xy xy+xz xx+xz

1

a+a

### Output:

Some Odd

All Even

All Even

## Explanation of Sample Data

The determinant in the first test case is simply  $x^2+x$  which has two odd coefficients. A tedious calculation shows that the determinant in the second test case is actually 0. The last test case has determinant 2a, so all coefficients are even.

COOK05

[The December 2010 Cook-Off](#)19 Dec 2010  
21:30:00

2 hours 30 minutes

227

[Home](#) » [Compete](#) » [The December 2010 Cook-Off](#) » The Morning Commute

## The Morning Commute Problem Code: COMMUTE

The Chef commutes to work every day using the city's underground metro. The schedule for the trains has recently been changed and he wants to know how long it will take to travel from the station nearest to his house and the station nearest to his restaurant.

The Chef doesn't want to change the route he took before, so he simply has to find out how long it will take to reach his restaurant along his usual route. This route is given by a sequence of stations  $s_0, s_1, \dots, s_n$  where  $s_0$  is the station where the Chef enters the metro and  $s_n$  is the station where the Chef exits the metro.

Trains are scheduled to run between every two consecutive stations  $s_{i-1}$  and  $s_i$ . Such a schedule is specified by three integers  $x_i$ ,  $l_i$ , and  $f_i$ . This means that the first train on this line starts operating at time  $x_i$ . The time it takes this train to travel from  $s_{i-1}$  and  $s_i$  is exactly  $l_i$  units. Finally, a train departs from station  $s_{i-1}$  every  $f_i$  minutes following the previous train. That is, a train departs at time  $x_i, x_i+f_i, x_i+2f_i$ , and so on.

The Chef is very experienced at navigating the metro so the time it takes him to transfer between trains at a given station is essentially zero. Thus, if the Chef arrives at a station, say  $s_i$ , the moment that the train from  $s_i$  to  $s_{i+1}$  is scheduled to depart, he skillfully hops on this next train. However, if the Chef arrives when no train to  $s_{i+1}$  is scheduled to depart, he must wait until the scheduled departure time.

Help the Chef figure out how long it will take him to travel from station  $s_0$  to station  $s_n$ . You may assume that the Chef is already at station  $s_0$  at time 0.

### Input

The first line consists of a single integer denoting the number of test cases (at most 50). Each test case begins with a line containing a single integer  $n$  between 1 and 1000 indicating the number of lines the Chef must traverse (so there are  $n+1$  stations  $s_0, s_1, \dots, s_n$ ). The next  $n$  lines describe the train schedules between stations, one per line. The  $i$ 'th such line gives the values  $x_i$ ,  $l_i$ , and  $f_i$  for the train that travels between stations  $s_{i-1}$  and  $s_i$ .

The  $x_i$  values will be between 0 and 1000 and the  $l_i$  and  $f_i$  values will be between 1 and 1000.

### Output

For each test case you are to output a single integer denoting the minimum time  $t$  for which the Chef can reach station  $s_n$  using the given route. Remember, the Chef starts at  $s_0$  at time 0.

## Example

### Input:

3

2

0 4 7

0 6 5

2

0 1 2

6 2 10

2

1 2 3

0 2 3

### Output:

11

8

5

[Home](#) » [Compete](#) » [The December 2010 Cook-Off](#) » Pairing Chefs

## Pairing Chefs Problem Code: PAIRING

The Chef's latest idea is that some cooks might work better in pairs. So, he is going to experiment by pairing up some of his employees to see if the quality of the food prepared

in his kitchen increases. However, only some pairs of employees are compatible. Two employees that are not compatible cannot be paired together.

For each pair of compatible employees, the Chef has assigned a number estimating how well the overall quality of the food might increase. Of course, each employee can only be paired with at most one other employee. Furthermore, it is ok to not pair some employees. So, your goal is to help the Chef decide how to pair the employees to maximize the total amount that the overall quality of food increases.

## Input

The first line contains a single integer denoting the number of test cases (at most 50). Each test case begins with two integers  $n$  and  $m$ . Here,  $n$  is the number of employees (between 2 and 1000) and  $m$  is the number of compatible pairs of employees (between 1 and 10,000). The employees are numbered from 0 to  $n-1$ . The next  $m$  lines describe a pair of compatible employees, one per line. The  $i$ 'th such line contains two distinct integers  $u_i, v_i$  between 0 and  $n-1$ . Strangely enough, the Chef estimates that picking the  $i$ 'th pair  $u_i, v_i$  will increase the quality of food prepared in his kitchen by exactly  $2^i$ .

No pair of employees will be given more than once in the input. That is, for distinct indices  $i$  and  $j$ , we do not have both  $u_i = u_j$  and  $v_i = v_j$ , nor do we have both  $u_i = v_j$  and  $v_i = u_j$ .

## Output

The output for each test case consists of the indices of the pairs of employees that are used in a maximum total value pairing (the indices are between 0 and  $m-1$ ). These indices should be given in increasing order with a single space between consecutive numbers. If there is more than one possible output, then any will do.

## Example

Input:

2

4 5

0 1

1 2

2 3

1 3

3 0

4 3

0 1

2 3

2 1

**Output:**

1 4

2

[Home](#) » [Compete](#) » [The December 2010 Cook-Off](#) » Preparing Dishes

## Preparing Dishes Problem Code: PREPARE

The Chef and his assistant are, by far, the two best cooks in the kitchen. They can prepare many dishes in far less time than the other cooks in the kitchen. Also, the Chef and his assistant are able to prepare many dishes in succession, but any other cook can only prepare one dish before they need a break.

Now, the supper rush has arrived and many orders for various dishes have just arrived. For each order, the Chef has an estimate on how long it will take him or his assistant to prepare the dish as well as an estimate of how long it will take any other cook to prepare the dish. So, the Chef must decide which dishes will be prepared by himself or his assistant and which dishes will be prepared by other cooks. Furthermore, he must divide the dishes not sent to other cooks between himself and his assistant.

Even though every other cook can only prepare one dish, there are many such cooks so the Chef can delegate as many dishes as he wants to these other cooks. These cooks will immediately start working on their dish. The remaining dishes are partitioned between the Chef and the assistant and the time it takes each of them to finish the dishes they have been assigned is the sum of the times of each assigned dish.

Help the Chef decide which dishes should be given to other cooks and how to divide the remaining dishes between himself and his assistant to minimize the time that the last dish is prepared.

### Input

The first line contains a single integer indicating the number of test cases (at most 50). Each test case begins with a single integer  $n$  indicating how many dishes must be prepared. The next  $n$  lines describe the dishes, one per line. The first integer on the  $i$ 'th line

indicates how long it will take any other cook to prepare the dish and the second indicates how long it will take the Chef or his assistant to prepare the dish.

Bounds: Both n and all times given in the input are between 1 and 1000.

## Output

The output for each test case consists of a single integer t which is the minimum time such that there is a way to give some dishes to other cooks and partition the remaining dishes between the Chef and his assistant so that all dishes are prepared within t units of time.

## Example

**Input:**

2

7

10 1

10 2

10 1

10 2

10 1

3 1

3 1

3

1 2

1 3

1 4

**Output:**

4

1

[Home](#) » [Compete](#) » [The December 2010 Cook-Off](#) » Seeding The Pattern

## Seeding The Pattern Problem Code: SEEDS

The Chef has decided to take a break from the restaurant. Rather than sitting on some boring tropical beach for hours on end, he has decided to join an archaeological dig. His latest discovery is an ancient tablet containing a list of numbers that seem to fit some pattern. At the top of the list are  $d$  numbers  $a_0, \dots, a_{d-1}$ , each of which is either 0 or 1. The rest of the list is only partially readable, but it seems to contain an extremely long list of 0/1 numbers  $x_0, x_1, \dots$

Only some of the  $x_i$  values are readable and the rest are too damaged to make out. The Chef conjectures that these numbers fit the following pattern. For any number  $n \geq 0$ , the Chef conjectures that  $x_{n+d} = a_0x_n + \dots + a_{d-1}x_{n+d-1}$ . Because of the nature of this pattern, he is calling the first  $d$  integers  $x_0, \dots, x_{d-1}$  the "seeds". Finally, this ancient civilization appears to be only concerned about whether a number is odd or even. Because of this, his conjecture is that the numbers fit the previous pattern when all results are reduced modulo 2.

Your job is to help the Chef determine the validity of his conjecture. Specifically, you are to determine if there is a way to assign either a 0 or a 1 to all unreadable  $x_i$  entries so that the pattern holds. If so, you must determine if there is more than one way to do this or if there is exactly one way to do this. If there is exactly one way, you must specify the values of the "seeds".

### Input

The first line contains a single integer indicating the number of test cases to follow (at most 50). Each test case begins with two integers  $d$  and  $k$ . The next line contains the  $d$  values  $a_0, \dots, a_{d-1}$  that are written at the top of the tablet. Finally,  $k$  lines follow where the  $j$ 'th line contains two integers  $i(j)$  and  $x$ . This specifies the value of the readable number  $x_{i(j)}$ . The  $i(j)$  values will appear in strictly increasing order.

Bounds: Both  $d$  and  $k$  are between 1 and 20. All values  $a_i$  and  $x_{i(j)}$  are either 0 or 1. Finally, each index  $i(j)$  is between 0 and  $2^{31}-1$ .

### Output

The output for each test case is a single line containing one of three messages. If there is no way to fill in the missing values  $x_i$  so that the pattern holds, then you should output "no solutions". If there are multiple ways to assign values to the missing  $x_i$  so that the pattern holds, then output "multiple solutions".

Finally, if there is only one way to assign values to the missing  $x_i$  values, then you should output a line containing the  $d$  values  $x_0, \dots, x_{d-1}$  with a single space between consecutive numbers. Since these values are calculated modulo 2, then they are either 0 or 1.

## Example

### Input:

3

2 2

1 1

0 0

2 1

2 2

1 1

0 0

3 0

2 2

1 0

0 0

2 1

### Output:

0 1

multiple solutions

no solution

## Newspaper Puzzle Problem Code: PUZZLES

When things slow down in the kitchen, the Chef likes to do puzzles he finds in the newspaper until more orders arrive. Today, a new type of puzzle has been featured and the Chef needs help solving it.

In this puzzle, a series of groups of non-zero integers are given where each group has exactly three integers. Furthermore, for each positive integer  $k$ , the number of groups including  $k$  plus the number of groups including  $-k$  does not exceed 3. The goal is to assign, for each positive integer  $k$ , either the character 'T' or 'F' so that every group of three integers has

- at least one negative integer  $k$  such that  $|k|$  is assigned 'F', or
- at least one positive integer  $k$  such that  $k$  is assigned 'T'

Finally, you are guaranteed that the absolute values the three integers in any particular group are all distinct. Now, newspaper puzzles are only fun if they are solvable so you are guaranteed that there is a solution. Your task is to find such an assignment of 'T' or 'F' values to the positive integers meeting the above requirements.

### Input

The first line consists of a single integer denoting the number of test cases (at most 30). The first line for each test case consists of two integers  $C$  and  $V$ . Following this are  $C$  lines, one per group of integers in the puzzle. Each such line consisting of 3 non-zero integers between  $-V$  and  $V$ . Furthermore, the absolute values of these three integers are distinct.

Finally, the input is such that for every integer  $k$ , the total number of groups containing either  $k$  or  $-k$  is at most 3.

Bounds:  $1 \leq C \leq 1000$  and  $3 \leq V \leq 3000$ .

### Output

The output for each test case is a single line consisting of  $V$  characters 'T' or 'F'. These characters should appear consecutively with no spaces between. The  $i$ 'th such character is what positive integer  $i$  is assigned. This should be done so that each group of three integers has the property described above.

If an integer  $k$  between 1 and  $V$  does not have either  $k$  or  $-k$  appearing in a group of three integers then you should still assign it a character 'T' or 'F'. Finally, if there are multiple solutions then any will do.

### Example

**Input:**

3 3

1 2 3

1 -2 -3

-1 2 -3

2 7

5 -2 1

3 -1 2

**Output:**

TFF

TFTFFTF

## Explanation of Output

In the first case, the first two groups include 1 which is assigned 'T', and the last group contains -3 and 3 is assigned 'F' so all groups meet the criteria. In the second case, the first group has 1 which is assigned 'T' and the second group has 3 which is also assigned 'T' so all groups in this test case also meet the criteria.

JAN11	<a href="#">January 2011 Challenge</a>	01 Jan 2011 15:00:00	10 days	400
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Count Relations

## Count Relations Problem Code: COUNTREL

**Read problems statements in [Mandarin Chinise.](#)**

Let A be a set of the first **N** positive integers : $A=\{1,2,3,4,\dots,N\}$

Let B be the set containing all the subsets of A.

Let A be a set of the first **N** positive integers : $A=\{1,2,3,4,\dots,N\}$

Let B be the set containing all the subsets of A.

Professor Eric is a mathematician who defined two kind of relations R1 and R2 on set B. The relations are defined as follows:

$R1 = \{ (x, y) : x \text{ and } y \text{ belong to } B \text{ and } x \text{ is not a subset of } y \text{ and } y \text{ is not a subset of } x \text{ and the intersection of } x \text{ and } y \text{ is equal to empty set } \}$

$R2 = \{ (x, y) : x \text{ and } y \text{ belong to } B \text{ and } x \text{ is not a subset of } y \text{ and } y \text{ is not a subset of } x \text{ and the intersection of } x \text{ and } y \text{ is not equal to empty set } \}$

Now given the number  $N$ , Professor Eric wants to know how many relations of kind  $R1$  and  $R2$  exists. Help him.

**NOTE :**  $(x, y)$  is the same as  $(y, x)$ , i.e the pairs are **unordered**.

### **Input format:**

The first line contains the number of test cases  $T$ . Each of the test case is denoted by a single integer  $N$ .

### **Output format:**

Output  $T$  lines, one for each test case, containing two integers denoting the number of relations of kind  $R1$  and  $R2$  respectively, modulo 100000007.

### **Example**

#### **Sample Input:**

3

1

2

3

#### **Sample Output:**

0 0

1 0

6 3

**Constraints:**

$1 \leq T \leq 1000$

$1 \leq N \leq 10^{18}$

**Explanation:**

Let  $A = \{1, 2\}$

Then  $B = \{\text{Phi}, \{1\}, \{2\}, \{1, 2\}\}$

$\text{Phi} = \text{Empty Set}$

So  $R1 = \text{Either } \{\{\{1\}, \{2\}\}\} \text{ or } \{\{\{2\}, \{1\}\}\}$

and  $R2 = \text{No relation exists}$

So, there is 1 relation of kind  $R1$  and 0 relation of kind  $R2$ .

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Flu Shot Lineup

## Flu Shot Lineup Problem Code: FLUSHOT

A new strain of flu has broken out. Fortunately, a vaccine was developed very quickly and is now being administered to the public. Your local health clinic is administering this vaccine, but the waiting line is very long.

For safety reasons, people are not allowed to stand very close to each other as the flu is not under control yet. However, many people were not aware of this precaution. A health and safety official recently examined the line and has determined that people need to spread out more in the line so that they are at least  $T$  units away from each other. This needs to be done as quickly as possible so we need to calculate the minimum distance  $D$  such that it is possible for every person to move at most  $D$  units so the distance between any two people is at least  $T$ . Specifically,  $D$  should be the minimum value such that there are locations  $x'_i$  so that  $|x_i - x'_i| \leq D$  for each person  $i$  and  $|x'_i - x'_j| \geq T$  for any two distinct people  $i, j$ . Furthermore, since nobody can move past the receptionist we must also have that  $x'_i \geq 0$ .

The location of each person is given by the number of meters they are standing from the receptionist. When spreading out, people may move either forward or backward in line but nobody may move past the location of the receptionist.

### Input

The first line of input contains a single integer  $K \leq 30$  indicating the number of test cases to follow. Each test case begins with a line containing an integer  $N$  (the number of people)

and a floating point value  $T$  (the minimum distance that should be between people). The location of each person  $i$  is described by single floating point value  $x_i$  which means person  $i$  is  $x_i$  meters from the receptionist. These values appear in non-decreasing order on the following  $N$  lines, one value per line.

Bounds:  $1 \leq N \leq 10,000$  and  $T$  and every  $x_i$  is between 0 and 1,000,000 and is given with at most 3 decimals of precision.

## Output

For each test case, you should output the minimum value of  $D$  with exactly 4 decimals of precision on a single line.

## Example

### Input:

3

2 4

1

2

2 2

1

2

4 1

0

0.5

0.6

2.75

### Output:

2.0000

0.5000

1.4000

## Explanation of Sample Data

In the first test case, the first person can move to location 0 and the second to location 4 with the maximum distance moved being 2. In the second case, person 1 can move to location 0.5 and person 2 can move to location 2.5 for a maximum distance moved being 0.5. Finally, in the last output the first person does not move, the second moves to location 1, the third to location 2, and the fourth to location 3. The maximum distance moved by any person was done by the third person who moved 1.4 meters to their destination.

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Splitting Giant Subs

## Splitting Giant Subs Problem Code: SPLIT

A submarine sandwich (or a "sub", for short) is a very long and skinny sandwich. The Chef loves to make extremely long subs that he affectionately calls "giant subs". Since a giant sub is so long, the Chef likes to place different sandwich ingredients along different sections of the sub. For instance, the first 20 cm of the sub may have a roast beef filling, the second 20 cm may have tuna, the third 20 cm of the sub might only contain vegetables, and so on. The only restriction on how the ingredients are placed is that each ingredient must appear in an even number of 20 cm segments, for aesthetic reasons.

Now, Jack and Jill are going to have an eating competition. They have ordered one of the Chef's giant subs and will race to see who can eat half of the sub first. To be fair, both Jack and Jill must consume the same number of each ingredient. So, they will cut the sub into small pieces and partition these pieces so that for any sandwich ingredient, both Jack and Jill receive the same number of sections of the sub including this ingredient.

Both Jack and Jill are very anxious to get started with the competition, so they want to do this as fast as possible. Help them cut the sub into smaller parts using the least number of cuts!

### Input

The input begins with a single line consisting of a single integer  $k \leq 30$  that indicates the number of test cases to follow. The first line of each test case begins with a single even integer  $n$  between 2 and 1000 indicating the number of sections in the giant sub.

The last line of each test case contains  $n$  integers between 1 and 500 describe the ingredients used in the sub. The  $i$ 'th such integer indicates which ingredient is used in the  $i$ 'th section of the sub. You are guaranteed that each integer/ingredient appears an even number of times.

## Output

The output for each test case should consist of two lines. The first line indicates the number of cuts that should be made. Say this number is  $c$ . Then the second line consists of  $c$  integers between 1 and  $n-1$  appearing in strictly increasing order. The  $i$ 'th such integer, say  $t_i$ , indicates that a cut should be made in the sub to separate section  $t_i$  and  $t_{i+1}$  (the first section of the sub has index 1).

These pieces will then be distributed to Jack and Jill in the following manner. Jack will receive all sections from the start of the sub until the cut after section  $t_1$ . Then Jill will receive all sections between the cuts made after section  $t_1$  and  $t_2$ . Then Jack receives the sections between the cuts after sections  $t_2$  and  $t_3$ , etc. Thus, the pieces of the sub are given to Jack and Jill in an alternating fashion. The final piece of the sub (the part after section  $t_c$ ) is given to whoever did not receive the section of the sub that ended with section  $t_c$ .

The output should be such that Jack and Jill receive the same number of each ingredient after these cuts are made (note that this is always possible). Your goal should be to do this with as few cuts as possible. This is a scored problem so your output does not have to be optimal, it may contain more cuts than is necessary.

## Example

**Input:**

3

8

1 2 3 2 3 2 1 2

6

1 1 2 2 3 3

8

500 400 2 400 500 500 2 500

**Output:**

1

4

3

1 3 5

4

1 3 4 6

## Scoring:

The score for each test case is simply  $c$ , the number of cuts made. The score for all test cases in a file is the sum of the scores for the individual test cases in that file. There are multiple files used to test this problem and your overall score is the average of the scores for each file. Your goal should be to make this score as small as possible.

## Test Data:

Some of the test data is created specifically to make certain heuristics perform poorly. Other test data is generated randomly from varying distributions.

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Chess Pushing Game

# Chess Pushing Game Problem Code: CHESSGM

Dave is playing a game with three pawns on a long board. The board can be seen as an (semi-)infinitely long sequence of grid squares numbered starting from 0 with width 3. Thus, the squares on the board can be referred to using  $(i,j)$  where  $i \geq 0$  and  $j = 0, 1$  or  $2$ . For this problem, the distance between two pawns is the distance along the first coordinate. For example, distance between pawns at  $(0,2)$  and  $(10,0)$  is 10.

The pawns are numbered 0, 1 and 2. Pawn  $j$  starts off at square  $(0, j)$  initially. In one move, a pawn can move one square to the right (increasing first coordinate, while second coordinate remains same). For example, a pawn can move from  $(4, 2)$  to  $(5, 2)$ . Another restriction is that if  $p < q$ , then pawn  $q$  can never be to the right (higher first coordinate) of pawn  $p$ . Yet another restriction is that at any point of time, pawns  $p$  and  $p+1$  (for  $p=0$  or  $p=1$ ) should not be separated by a distance more than  $D$ .

At the end of  $K$  moves, all the pawns should end up at the same first coordinate. Dave wants to know how many ways there are to complete this game. Help him!

## Input

The first and only line contains two integers  $D, K$  ( $0 \leq D \leq 7$ ;  $0 \leq K \leq 10^9$ ).

## Output

Output an only integer which is the result calculated, by modulo of 1,000,000,007.

## Example

**Input:**

1 3

**Output:**

1

**Input:**

2 6

**Output:**

5

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Restaurant Expansion

## Restaurant Expansion Problem Code: RESTEXP

You own a large local restaurant. Because now it is very popular, you want to expand your restaurant in a neighboring country.

The country consists  $N$  cities numbered 1 to  $N$ . There are  $N-1$  roads connecting the cities. The country is efficiently organized, so there is exactly one path between every pairs of cities. No road will connect a city to itself.

You want to build new restaurants in some cities. From an amateur survey, for each city  $i$  you know its profit index  $S_i$ , that is the profit you will get if you build a new restaurant in city  $i$ .

Initially, you have  $C$  chefs in city 1. You have  $D$  days to accomplish your expansion. On each day, you may perform exactly one of these options:

- Do nothing.
- Transfer any number of chefs from a city to another city, provided that the two cities are connected by a road.
- Build a new restaurant in a city, provided that there is at least one chef in the city. After that, all chefs in the city cannot be transferred again. You can only build one restaurant per city.

After D days has passed, you get the profit associated with a city if the city has a restaurant. Of course, you want the maximum possible total profit. Arrange your expansion plan in order to maximize your total profit.

## Input

The first line contains three integers N, C, and D. The next line contains a sequence of N integers  $S_i$ . The next  $N-1$  lines contains two integers  $u_i$  and  $v_i$ , where  $u_i$  and  $v_i$  are the two cities connected by the  $i$ -th road.

## Output

In the first line, output the maximum possible total profit. The next D lines contain your expansion plan. In each of the next D lines, output exactly one of the following (quotes for clarity).

- "**nothing**", if you plan to do nothing on that day.
- "**transfer a b c**", if you plan to transfer c chefs from city a to city b on that day.
- "**build a**", if you plan to build a new restaurant in city a on that day.

If there are several expansion plans that lead to the same maximum profit, you may output any.

## Example

### Input

4 2 5

-10 5 2 6

1 2

2 3

2 4

### Output

11

transfer 1 2 2

transfer 2 4 1

nothing

build 4

build 2

## Constraints

- $1 \leq N \leq 30$
- $1 \leq C \leq 30$
- $1 \leq D \leq 30$
- $-1000 \leq S_i \leq 1000$
- $1 \leq u_i \leq N$
- $1 \leq v_i \leq N$
- $u_i \neq v_i$
- There is exactly one path between every pair of cities

[Home](#) » [Compete](#) » [January 2011 Challenge](#) » Chef attic window

## Chef attic window Problem Code: WINDOW

Chef lives in a big house. Almost all is perfect in it except for one thing. Chef has an attic, but there is no window in it. Now, he wants to build a window there.

The attic wall has the form of a right triangle. We can imagine it as a triangle on a coordinate plane with vertices  $(0; 0)$ ,  $(N; 0)$ ,  $(N; N * A / B)$ , where  $N, A, B$  are some positive integers.

As a modern guy Chef want to build a modern  $(L, K)$ -window on the attic wall. A modern  $(L, K)$ -window is made up of  $L+1$  vertical lines and  $K+1$  horizontal lines such that all their intersections lie on the attic wall. Here  $L, K$  are positive integers.

Formally he can choose  $L+1$  integers  $0 \leq x[0] < x[1] < \dots < x[L] \leq N$  and  $K+1$  integers  $0 \leq y[0] < y[1] < \dots < y[K] \leq N * A / B$  such that all points  $(x[i], y[j])$  lie on the wall. Then, rectangular grid with all these points as nodes is required window.

Chef is interested in knowing how many ways he can choose the window. But since this number can be very large he wants to find it modulo **900000011**. Help him.

### Input

The first line contains a single positive integer  $T \leq 50$ , the number of test cases.  $T$  test cases follow. The only line of each test case contains five positive integers  $N, A, B, K, L$ , where  $N, A, B \leq 10^{18}$  and  $K, L \leq 10$ .

### Output

For each test case, output a single line containing the number of ways to choose a window modulo **900000011**.

## Example

Input:

3

4 2 3 1 1

4 3 2 2 2

1001 101 97 3 2

Output:

5

4

579415965

## Explanation

In the first case we need calculate the number of rectangles which lie in triangle with vertices **(0; 0)**, **(4; 0)**, **(4; 8/3)**. There exist 5 such rectangles. Among them 3 have size **1 x 1**. One has size **2 x 1** and one have size **1 x 2**.

In the second case we have two rectangles **2 x 2** and one rectangle **2 x 3** where we can choose middle horizontal segment in two ways. So we have in total 4 possible choices

COOK06	<a href="#">The January 2011 Cook-Off</a>	23 Jan 2011 21:30:00	2 hours 30 minutes	528
--------	---	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [The January 2011 Cook-Off](#) » Holes in the text

## Holes in the text Problem Code: HOLES

Chef wrote some text on a piece of paper and now he wants to know how many holes are in the text. What is a hole? If you think of the paper as the plane and a letter as a curve on the plane, then each letter divides the plane into regions. For example letters "A", "D", "O", "P", "R" divide the plane into two regions so we say these letters each have one hole.

Similarly, letter "B" has two holes and letters such as "C", "E", "F", "K" have no holes. We say that the number of holes in the text is equal to the total number of holes in the letters of the text. Help Chef to determine how many holes are in the text.

## Input

The first line contains a single integer **T  $\leq 40$** , the number of test cases. **T** test cases follow. The only line of each test case contains a non-empty text composed only of uppercase letters of English alphabet. The length of the text is less than 100. There are no any spaces in the input.

## Output

For each test case, output a single line containing the number of holes in the corresponding text.

## Example

Input:

2

CODECHEF

DRINKEATCODE

Output:

2

5

[Home](#) » [Compete](#) » [The January 2011 Cook-Off](#) » Chef team

## Chef team Problem Code: CHEFTEAM

Chef has **N** subordinates. In order to complete a very important order he will choose exactly **K** of them. He can't choose less than **K** since it will be not enough to complete the order in time. On the other hand if he chooses more than **K** subordinates he can't control them during the operation. Help him to find the number of ways he can choose the team to complete this very important order.

## Input

The first line contains a single positive integer **T  $\leq 100$** , the number of test cases. **T** test cases follow. The only line of each test case contains two integers **N** and **K**, where **0  $\leq N, K < 2^{64}$** . It is guaranteed that the answer will be less than **2<sup>64</sup>**.

## Output

For each test case, output a single line containing the number of ways to choose the required team.

## Example

Input:

3

2 1

3 3

10 5

Output:

2

1

252

[Home](#) » [Compete](#) » [The January 2011 Cook-Off](#) » Whole submatrix

## Whole submatrix Problem Code: WINDOW2

After Chef successfully built a modern  $(L, K)$ -window on the attic wall he decided to expand the notion of the  $(L, K)$ -window in some other areas. Now he considers a rectangular grid that contains only zeroes and ones and has size  $N \times M$ . He considers the  $(L, K)$ -window here as any submatrix of size  $L \times K$  that contains only ones. Formally he defines  $(L, K)$ -window as any  $(K+L)$ -tuple  $(R_1, \dots, R_L, C_1, \dots, C_K)$  such that  $1 \leq R_1 < \dots < R_L \leq N$ ,  $1 \leq C_1 < \dots < C_K \leq M$  and  $A[R_i][C_j] = 1$  for all  $1 \leq i \leq L$ ,  $1 \leq j \leq K$ . Here  $A[r][c]$  is the  $c$ -th element of the  $r$ -th row of considered rectangular grid.

Why does Chef call some  $(K+L)$ -tuple of numbers by the window? Just mark all points  $(R_i, C_j)$  ( $1 \leq i \leq L$ ,  $1 \leq j \leq K$ ) on the plane and join by line segments all pairs of points that has equal abscises or ordinates and you will see that this picture is like a window.

Now Chef considers some particular  $N \times M$  grid and wants to calculate the total number of  $(L, K)$ -windows in this rectangular grid. Help him. Since this number can be very large calculate the result modulo **1000000080798150871**.

## Input

The first line contains a single positive integer  $T \leq 100$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains four positive integers  $N, M, L, K$ , where  $L, N \leq 1000, K, M \leq 3$ . Next  $N$  lines describe the rectangular grid considered by Chef. Each of these lines contains  $M$  symbols. Every symbol is either one or zero.

## Output

For each test case, output a single line containing the total number of  $(L, K)$ -windows for the given grid modulo **1000000080798150871**.

## Example

**Input:**

```
2
3 2 2 1
11
01
10
3 3 2 2
111
101
111
```

**Output:**

```
2
5
```

## Explanation

In the first case it is just the number of pairs of cells with value 1 that have the same column number.

In the second case we have the following **(2, 2)**-windows:

(First row, Second row, First column, Third column)  
 (First row, Third row, First column, Second column)  
 (First row, Third row, First column, Third column)  
 (First row, Third row, Second column, Third column)  
 (Second row, Third row, First column, Third column)

[Home](#) » [Compete](#) » [The January 2011 Cook-Off](#) » Time of collisions

## Time of collisions Problem Code: COLLTIME

Identical small balls are located on a straight line and can move along this line only. Each ball moves with a constant velocity, but velocities of different balls may be different. When two balls meet, a perfectly elastic collision occurs. It's a common-known physical fact that when two equal-mass physical bodies, say **A** and **B**, collide perfectly elastically, they swap their velocities. That is, **A**'s new velocity is **B**'s old velocity and **B**'s new velocity is **A**'s old velocity. Your task is to find the sum of all moments of time when collisions occur. If three or more balls collide at the same moment of time in the same place then the following will occur. Let the collision involves **k >= 3** balls then you should consider this collision as simultaneous **k\*(k-1)/2** collisions. It means that you should count the moment of time of this collision **k\*(k-1)/2** times in your answer. The new velocities of these balls after all collisions have occurred will be as follows: the **1**-st and **k**-th balls swap their velocities, **2**-nd and **k-1**-th balls swap their velocities and so on. Explicitly, for each **i** between **1** and **k/2** balls **i** and **k+1-i** swap their velocities. Here we enumerate balls in order they were on the line in the moment of time just before the collision.

### Input

The first line contains a single positive integer **T <= 10**, the number of test cases. **T** test cases follow. The first line of each test case contains the number of balls **N (1 <= N <= 50000)**. Each of the following **N** lines contains 2 space-separated integers - the starting coordinate and the velocity of corresponding ball. All start coordinates are not greater than **1000000** in absolute value and all velocities are not greater than **20** in absolute value. All start coordinates are different.

### Output

For each test case, output a single line containing number **S** - the sum of moments of time of all collisions. If there are infinitely many collisions then print "**INF**". If there are only finitely many of them then it is guaranteed that the answer **S** is rational and you should print it in the form of a mixed number. That is if **S** is an integer then just print **S**, otherwise if it is less than one and equals to irreducible fraction **P/Q** then print "**P/Q**", otherwise print "**A P/Q**" where **A** is an integer part of **S** and **P/Q** is its fractional part in irreducible form. Note that for all fractions **P/Q** in output specifications we have **P < Q**.

### Example

**Input:**

3

2

-1 1

1 -1

2

0 1

1 -1

3

-1 1

2 -1

3 -5

**Output:**

1

1/2

2 5/12

**Explanation**

In the third test case we have collisions in the following moments of time: 1/4, 2/3 and 3/2.

[Home](#) » [Compete](#) » [The January 2011 Cook-Off](#) » Divisors number divisibility

**Divisors number divisibility** Problem Code: DIVNODIV

For a given positive integer **N** find the number of all positive integers **X** such that **N** divides **X** and the number of all positive divisors of **X** (including 1 and **X**) is equal to **N**. If there exist infinitely many such numbers print **-1**.

## Input

The first line contains a single positive integer **T  $\leq 100$** , the number of test cases. **T** test cases follow. The only line of each test case contains a positive integer **N**, where **N  $\leq 10^{18}$** .

## Output

For each test case, output a single line containing the answer for the corresponding test case.

## Example

### Input:

3

6

113

144

### Output:

2

1

-1

## Explanation

In the first test case only numbers 12 and 18 have 6 divisors and are divisible by 6.

In the second test case the only number is  $113^{112}$ .

In the third test case there are infinitely many of them.

FEB11

[February 2011  
Contest](#)

01 Feb 2011  
15:00:00

10 days

426

[Home](#) » [Compete](#) » [February 2011 Contest](#) » Coloring Colorable Graphs

# Coloring Colorable Graphs Problem Code: THREECLR

The Chef has a list of dishes he must prepare and he is able to work on many of them simultaneously (with the help of his employees, of course). For example, he can heat soup, bake a cake, marinate some meat, let some wine aerate, etc. all at the same time since no two of them require a common cooking tool. Unfortunately, his kitchen is dangerously low in its supply of tools and he has no more than one copy of each cooking tool. This means that some dishes cannot be prepared at the same time such as slicing two different cuts of meat since there is only one meat slicer.

So, the Chef has scheduled the preparation of the dishes into rounds. In each round, he and his staff prepare some dishes of which no two require a common cooking tool. For simplicity's sake, he will not start preparing dishes from a certain round until all dishes from the previous round are completed.

The Chef received the orders for today's dishes early in the day and he has had a lot of time to think about how to optimally schedule them. In fact, he even found a way to prepare all of the dishes in only 3 rounds! Unfortunately, he lost the piece of paper that contained this schedule and he has to start preparing dishes right away.

He needs your help and he needs it fast! Your job is to assign, to each dish, a round in the schedule such that no two dishes scheduled in the same round require a common cooking tool. It doesn't matter if it uses more than three rounds. Of course, fewer rounds are preferred, but the Chef would be happy with any schedule right now.

## Input

The first line contains an integer  $T$  denoting the number of test cases (at most 50). Each test case begins with two integers  $n$  and  $m$  where  $n$  denotes the number of dishes to be prepared and  $m$  denotes the number of pairs of dishes that require a common cooking tool. Following this are  $m$  lines, each containing two distinct integers  $u, v$  between 1 and  $n$ . This indicates that dishes  $u$  and  $v$  require a common cooking tool so they cannot be scheduled in the same round. No pair of dishes will appear more than once in the input.

The input will be such that there is a way to assign each dish to one of three rounds so that no two dishes in the same round require a common cooking tool. Of course, you won't be given this schedule.

Bounds:  $1 \leq n \leq 500$  and  $0 \leq m \leq 10,000$ .

## Output

The output for each test case is a single line containing  $n$  integers. These integers must be between 1 and  $n$  and the  $i$ 'th such integer indicates the round that dish  $i$  is scheduled in. The scheduling must be done so that no two dishes that require a common cooking tool appear in the same round. However, any output that conforms to these constraints will be accepted.

## Scoring

The score for a single test case is simply the largest integer that appears in the output for that test case. The total score over all test cases is the sum of the scores for the individual test cases. There are multiple test files, so your final score is the average of the scores over all test files.

## Example

Input:

3

4 5

1 2

3 1

1 4

3 2

2 4

10 15

1 2

2 3

3 4

4 5

5 1

6 8

8 10

10 7

7 9

9 6

1 6

2 7

3 8

4 9

5 10

3 2

1 2

2 3

**Output:**

1 2 3 3

1 2 1 2 3 4 4 3 3 2

1 3 1

## Sample Output Score

The score for the first test case is 3 since only three rounds are used. The score for the second test case is 4 since four rounds are used. Note that this is not optimum as the assignment 1 2 3 1 2 2 1 1 3 3 is a valid way to schedule the dishes in only three rounds. Finally, the last test case has score 3. Even though only two rounds are used (rounds 1 and 3), the output specification says that the score will be the largest integer used as a round in the output.

[Home](#) » [Compete](#) » [February 2011 Contest](#) » Bogosort

## Bogosort Problem Code: BOGOSORT

Recently Johnny have learned bogosort sorting algorithm. He thought that it is too ineffective. So he decided to improve it. As you may know this algorithm shuffles the sequence randomly until it is sorted. Johnny decided that we don't need to shuffle the whole sequence every time. If after the last shuffle several first elements end up in the right places we will fix them and don't shuffle those elements furthermore. We will do the same for the last elements if they are in the right places. For example, if the initial sequence is (3, 5, 1, 6, 4, 2) and after one shuffle Johnny gets (1, 2, 5, 4, 3, 6) he will fix 1, 2 and 6 and proceed with sorting (5, 4, 3) using the same algorithm. Johnny hopes that this optimization

will significantly improve the algorithm. Help him calculate the expected amount of shuffles for the improved algorithm to sort the sequence of the first  $n$  natural numbers given that no elements are in the right places initially.

## Input

The first line of input file is number  $t$  - the number of test cases. Each of the following  $t$  lines hold single number  $n$  - the number of elements in the sequence.

## Constraints

$1 \leq t \leq 150$   
 $2 \leq n \leq 150$

## Output

For each test case output the expected amount of shuffles needed for the improved algorithm to sort the sequence of first  $n$  natural numbers in the form of irreducible fractions.

## Example

**Input:**

3

2

6

10

**Output:**

2

1826/189

877318/35343

# Glass Measurement Problem Code: GLASS

## Statement

Chef has  $n$  glasses each with a capacity of  $x_i$ .

There are  $Q+1$  people playing a game including the Chef himself. Everyone at his/her turn gives a positive integer  $M_i$  and asks the Chef if volume  $M$  of water can be transferred using the  $n$  glasses with Chef. Chef at his own turn wants to give  $M$  as the highest integer volume which cannot be constructed using these  $n$  glasses.

So, you need to help chef with this game.

Assume that to construct a volume  $M$ , we have a reserve with infinite volume of liquid. If Chef uses  $i^{\text{th}}$  glass, he has to transfer a volume equal to  $x_i$  in one turn. He can use a glass multiple times. However, transfer of liquid between glasses is not permitted. For example, you cannot obtain volume of 2 by tranferring liquid from glass of volume 5 to glass of volume 3.

It is always possible to construct volume 0.

## Input

First line contains  $t$  ( $\leq 20$ ) which is the number of test cases. For each test case, the first line contains  $n$  (the number of glasses). The next line contains  $n$  space separated integers denoting the capacity of each glass. The next line contains a single integer  $Q$  - the number of players excluding the Chef. Then follows a line containing 3 space separated integers  $a, b, c$ . Calculate  $M_i$  for the  $i^{\text{th}}$  person as  $(a*i+b) \% c$  ( $i \geq 1$ ).

## Output

For each test case, the first line of output should contain the highest value  $M$  desired by Chef [ If there is no positive  $M$ , then output -1 ]. Then follows a line containing 2 integers  $A$  and  $B$  where  $A$  denotes the number of  $M_i$  which cannot be constructed using the given glasses and  $B$  denotes the number of  $M_i$  which can be constructed.

## Sample Input

1

2

3 5

2

8 1 10

## Sample Output

7

1 1

## Constraints

$1 < n \leq 10$   
 $1 \leq x_i \leq 10^7$   
 $\text{MIN}(x_i) \leq 100000$   
 $1 \leq Q \leq 1000000$   
 $1 \leq a, c \leq 10^{12}$   
 $0 \leq b \leq 10^{12}$

## Sample Test Case Explanation

There are 2 glasses of size 3 and 5. Volumes 1, 2, 4, 7 cannot be created by the combinations of glass suggested in the problem. Hence, the highest volume is 7. The 2 queries are for 9 and 7. 9 can be created using the glasses and 7 cannot.

[Home](#) » [Compete](#) » [February 2011 Contest](#) » Counting Terms

## Counting Terms Problem Code: TERM

When the Chef is not busy with cooking delicious foods, one of his favorite pastime is to solve mathematical problem. But as you know, he is not very good with programming. So when he fails to solve a problem he usually asks John to write a program that will solve it. Recently the chef became interested in finding the expansion of  $(x_1+x_2+\dots+x_k)^n$ . Specifically he is more interested in number of terms in it. Being unable to solve the problem he asked John for help. The conversation between John and chef goes as

Chef: Can you write a program that will calculate  $f(n,k) = \text{number of terms in } (x_1+x_2+\dots+x_k)^n$  ?

John: That is a very easy problem.

Chef: Oh, really? Then calculate the sum over ( $i=0$  to  $n$ ) ( $f(i,k)$ )

John: That is also an easy problem.

Chef: Then go finish it. And did I mention that I am calculating modulo a prime number  $p$ .

John: What?

Chef: Say  $k=2$ ,  $n=4$ ,  $p=3$ . Then  $(x_1+x_2)^4 \% 3 = (x_1^4+4 x_1^3 x_2+6 x_1^2 x_2^2+4 x_1 x_2^3+x_2^4) \% 3 = (x_1^4+x_1^3 x_2+x_1 x_2^3+x_2^4) \% 3$ . So if  $p=3$  then  $f(4,2)=4$ .

John: OK, I am doing it.

As John started to solve the problem he realized it is not that easy. Now he asked you, his best friend, for help.

As the answer can be quite large print the output modulo 1000003.

## Input

Input file starts with a line containing a number  $T$ .  $T$  lines follow each describing a test case. Each test case consists of a line containing three space separated integers  $n, k, p$ .

## Output

Output  $T$  lines, each line containing the Sum over ( $i=0$  to  $n$ )  $f(i,k)$  for the given prime  $p$  modulo 1000003.

## Limit:

$1 \leq T \leq 10000$

$0 \leq n \leq 10^{15}$

$1 \leq k \leq 10^{15}$

$1 \leq p \leq 100$

$p$  prime number

## Example

### Input:

7

10 2 2

10 2 3

10 3 2

10 3 3

4 2 3

3 2 3

0 2 2

### Output:

37

42

85

112

12

8

1

[Home](#) » [Compete](#) » [February 2011 Contest](#) » Milestones

## Milestones Problem Code: MSTONES

Once upon a time, there was a Kingdom of Seven Roads. Besides a fancy name, it actually had exactly 7 straight roads. Its residents wanted to keep track of the distances they traveled so they placed milestones along some roads. The roads slowly deteriorated and disappeared but some milestones remained. Archeologists documented remaining milestones and want to reconstruct the kingdom, starting with its main road. Help them by finding the maximum number of collinear milestones.

### Input

The first line contains a single integer  $T$ , the number of test cases. The first line of each testcase contains the number of documented milestones  $N$ . Following lines give the coordinates  $(X_i, Y_i)$  of those milestones. Coordinates of all milestones will be different.

### Output

For each test case, output the maximum number of collinear milestones.

### Constraints

- $T \leq 30$
- $1 \leq N \leq 10\,000$
- $-15\,000 \leq X_i, Y_i \leq 15\,000$

### Example

**Input:**

2

5

0 0

1 0

2 0

1 1

3 1

2

1 1

10 10

**Output:**

3

2

[Home](#) » [Compete](#) » [February 2011 Contest](#) » Rush Hour

## Rush Hour Problem Code: RUSHOUR

Rush Hour is a popular puzzle game which is now available on many mobile devices.

There is a board of size  $M \times N$ . The board is full of cars and trucks. Each car has size  $1 \times 2$  and each truck has size  $1 \times 3$ . Cars and trucks can be either horizontal or vertical.

At each step, we can move a horizontal car or truck along its row if there is no other objects in its way. Similarly we can move a vertical car or truck along its column.

Among the cars, one belongs to the president. It is always horizontal. The object of the game is to get the president's car to the exit gate at the rightmost column in its row.

You need to write a solver for Rush Hour, which takes a map of the game and returns the minimum number of steps to solve it.

As an example, look at the figures below. The president's car is painted in yellow. It takes 5 steps to solve this game instance.

## Input

The first line contains the number of test cases (about 10). Each test case has the following form.

The first line contains three numbers M, N and K. M and N are the size of the board. K are the total number of cars and trucks. ( $6 \leq M, N \leq 12$ ,  $3 \leq K \leq 26$ )

Each line in the next M lines contains N characters. Each character can be '.' or 'a'..'z'. '.' stands for an empty square. 'a'..'z' represents the ID of the car or truck that occupies the square. The president's car is always 'a'.

It is guaranteed that the president's car initial position is not at the rightmost column.

The requirement of the game is guaranteed by the input, that is each object can only have size 1x2 or 1x3.

You can assume that there is always a solution and **the optimal number of steps to solve any puzzle instance given in the input is at most 10**.

Each test case's input is separated by a blank line.

## Output

For each test case, print in a single line the optimal number of steps to solve the corresponding puzzle.

## Test case generator

Note that in general, solving Rush Hour puzzle is very hard, so in this problem you are only required to solve random generated instances of the puzzle. We briefly describe how we generated the test cases.

Given M, N, K, for each car, we let it be horizontal or vertical with equal probability. The car's size is randomly picked from 2 to 3. The car's position is also chosen randomly. To make sure that a solution exists, we start with a state in which the president's car is already at the exit gate. After that, we perform some random steps (around 10000). The final state is used as the input data.

## Example

**Input**

2

6 6 13

b.c..e

b.cdde

fgahaal

fghjjl

.giikk

mmm...

6 6 10

.....

.bb.cc

aad.hi

fed.hi

feg..j

f.g..j

**Output**

5

6

The first sample test case is explained in the figure.

COOK07

[February Cook - Off Challenge](#)20 Feb 2011  
21:30:00

2 hours 30 minutes

301

[Home](#) » [Compete](#) » [February Cook - Off Challenge](#) » Three Way Communications

## Three Way Communications Problem Code: COMM3

The Chef likes to stay in touch with his staff. So, the Chef, the head server, and the sous-chef all carry two-way transceivers so they can stay in constant contact. Of course, these transceivers have a limited range so if two are too far apart, they cannot communicate directly.

The Chef invested in top-of-the-line transceivers which have a few advanced features. One is that even if two people cannot talk directly because they are out of range, if there is another transceiver that is close enough to both, then the two transceivers can still communicate with each other using the third transceiver as an intermediate device.

There has been a minor emergency in the Chef's restaurant and he needs to communicate with both the head server and the sous-chef right away. Help the Chef determine if it is possible for all three people to communicate with each other, even if two must communicate through the third because they are too far apart.

### Input

The first line contains a single positive integer  $T \leq 100$  indicating the number of test cases to follow. The first line of each test case contains a positive integer  $R \leq 1,000$  indicating that two transceivers can communicate directly without an intermediate transceiver if they are at most  $R$  meters away from each other. The remaining three lines of the test case describe the current locations of the Chef, the head server, and the sous-chef, respectively. Each such line contains two integers  $X, Y$  (at most 10,000 in absolute value) indicating that the respective person is located at position  $X, Y$ .

### Output

For each test case you are to output a single line containing a single string. If it is possible for all three to communicate then you should output "yes". Otherwise, you should output "no".

To be clear, we say that two transceivers are close enough to communicate directly if the length of the straight line connecting their  $X, Y$  coordinates is at most  $R$ .

### Example

**Input:**

3

1

0 1

0 0

1 0

2

0 1

0 0

1 0

2

0 0

0 2

2 1

**Output:**

yes

yes

no

[Home](#) » [Compete](#) » [February Cook - Off Challenge](#) » Breaking Into Atoms

## Breaking Into Atoms Problem Code: ATOMS

Let  $X$  be the set of all integers between 0 and  $n-1$ . Suppose we have a collection  $S_1, S_2, \dots, S_m$  of subsets of  $X$ . Say an atom  $A$  is a subset of  $X$  such that for each  $S_i$  we have either  $A$  is a subset of  $S_i$  or  $A$  and  $S_i$  do not have any common elements.

Your task is to find a collection  $A_1, \dots, A_k$  of atoms such that every item in  $X$  is in some  $A_i$  and no two  $A_i, A_j$  with  $i \neq j$  share a common item. Surely such a collection exists as we could create a single set  $\{x\}$  for each  $x$  in  $X$ . A more interesting question is to minimize  $k$ , the number of atoms.

## Input

The first line contains a single positive integer  $t \leq 30$  indicating the number of test cases. Each test case begins with two integers  $n, m$  where  $n$  is the size of  $X$  and  $m$  is the number of sets  $S_i$ . Then  $m$  lines follow where the  $i$ 'th such line begins with an integer  $v_i$  between 1 and  $n$  (inclusive) indicating the size of  $S_i$ . Following this are  $v_i$  distinct integers between 0 and  $n-1$  that describe the contents of  $S_i$ .

You are guaranteed that  $1 \leq n \leq 100$  and  $1 \leq m \leq 30$ . Furthermore, each number between 0 and  $n-1$  will appear in at least one set  $S_i$ .

## Output

For each test case you are to output a single integer indicating the minimum number of atoms that  $X$  can be partitioned into to satisfy the constraints.

## Example

**Input:**

2

5 2

3 0 1 2

3 2 3 4

4 3

2 0 1

2 1 2

2 2 3

**Output:**

3

4

# Integer Sequences Problem Code: SEQUENCE

For a fixed integer  $n$ , let  $x_1, x_2, \dots, x_{2^n}$  be a sequence that contains each of the  $2^n$  different  $n$  bit integers exactly once. To be precise, we say an integer is an  $n$  bit integer if it can be expressed in binary with exactly  $n$  bits (with, perhaps, some leading zeros). We say the sequence is *gradual* if two consecutive numbers differ in exactly one bit when written in binary and the first and last numbers also differ in exactly one bit in their binary representations.

Your job is to generate such a sequence. This is normally a fairly standard exercise, but there is one extra constraint. For some reason, there are two particular  $n$  bit numbers  $a$  and  $b$  such that it is illegal to have  $a$  and  $b$  appear consecutively (in either order) in the sequence and it is illegal to have one of  $a$  or  $b$  at the start of the sequence and the other at the end.

For example, if  $n = 2$ ,  $a = 1$ , and  $b = 3$  then the sequence  $0, 2, 3, 1$  is not allowed since  $a$  and  $b$  are consecutive. The sequence  $1, 0, 2, 3$  is also not allowed since  $a$  is at the beginning and  $b$  is at the end. Finally, the sequence  $0, 1, 2, 3$  is also not allowed since two bits change when going from 1 to 2 and when going from 3 to 0.

## Input

The first line contains a single positive integer  $t \leq 30$  denoting the number of test cases to follow. Each test case consists of three numbers  $n, a, b$ . Here,  $n$  is the number of bits (between 1 and 15, inclusive) and  $a$  and  $b$  are distinct integers between 0 and  $2^n - 1$  (inclusive).

## Output

The output for each test case is a single line consisting of a sequence of  $2^n$  integers. This should be such that the sequence is gradual according to the above description and contains all  $n$  bit integers. Furthermore, numbers  $a$  and  $b$  should not appear consecutively (in either order) nor does the start of the sequence contain one of  $a$  or  $b$  and the end contain the other. If there are many such sequences then any will do. On the other hand, if there is no such sequence then simply print a single line containing -1.

## Example

**Input:**

3

2 0 1

3 1 2

3 5 7

**Output:**

-1

0 1 3 2 6 7 5 4

1 3 7 6 2 0 4 5

[Home](#) » [Compete](#) » [February Cook - Off Challenge](#) » Dance Floor Energy

## Dance Floor Energy Problem Code: ENERGY

The Chef is catering a dinner that will be followed by a dance. Unfortunately, the person in charge of organizing the dance has just become ill. The Chef still wants the dance to be successful since it is associated with the dinner he prepared. If the dance is not successful, then it may inadvertently leave a negative impression of the Chef's catering business on the guests.

The Chef has never organized a dance before and now he needs your help. The most complicated task the Chef has to undertake is the following. A dance is only successful if many people are on the dance floor. Thankfully, the guests are shy and are willing to be ordered to dance with other guests they like. The Chef knows which guests each person is willing to dance with and his goal is to have, at any moment in time, the maximum possible number of dancing pairs. A pair of guests will only dance if each is interested in dancing with the other.

To complicate matters, each guest may come and go at different parts of the dance. Specifically, each guest enters the dance only one time, stays for a non-zero length of time, and then leaves the dance and does not return. The Chef is only interested in maximizing the number of dancing pairs at any given moment in time. He is not concerned with repeatedly swapping people to new dance partners, so if a new person enters the dance then the Chef may break up many different dancing couples at this time and form new pairs to maximize the number of dancing pairs.

Say  $B$  is the total number of boys on the guest list and  $G$  is the total number of girls on the guest list. Your goal is the following. For each integer  $m$  between 0 and  $\min(B, G)$ , you should determine the total time that the maximum number of couples that the Chef can assign to the dance floor is  $m$ .

### Input

The first line contains a single positive integer  $T \leq 30$  describing the number of test cases. Each test case begins with three positive integers  $B$ ,  $G$ , and  $L$ .  $B$  is the number of boys that will attend the dance and  $G$  is the number of girls.  $L$  indicates the length of the dance.

B lines then follow, each describing a boy. Each such line begins with three integers S,T,N where  $0 \leq S < T \leq L$  and  $0 \leq N \leq G$ . Here, S is number of seconds after the dance starts that the boy enters and T is the number of seconds after the dance starts when the boy leaves. Then N distinct integers between 0 and G-1 follow on the same line that describe the girls that the boy is interested in dancing with.

Then G lines follow, each describe a girl in the same manner. The only difference is that  $0 \leq N \leq B$  and the last N integers on a line corresponding to a girl are distinct integers between 0 and B-1 describing the boys that the girl is interested in dancing with.

Bounds: Both B and G are integers between 1 and 200 and L is an integer between 1 and 1,000,000,000.

## Output

You should output a single line for each test case consisting of  $\min(G, B) + 1$  integers separated by a single space. Starting at index  $m = 0$ , the  $m$ 'th such integer denotes the total time that the maximum number of dancing pairs that can be formed is  $m$ . Notice that the sum of the numbers on the output line should be exactly L.

Recall that a pair between a particular boy and a particular girl can only be formed at a given moment of time if both are present at that time and each is interested in dancing with the other.

## Example

**Input:**

4

2 3 10

0 10 2 0 1

1 6 3 0 2 1

4 5 2 0 1

3 8 1 1

2 8 1 0

3 3 2 0

0 1 2 3 0 1 2

1 13 3 0 1 2

2 14 3 0 1 2

3 15 3 0 1 2

4 16 3 0 1 2

5 17 3 0 1 2

4 3 4 0

0 17 3 0 1 3

5 3 4 2 2 3

21 4 0 3 0 1 2

1 3 5 2 3 1

0 2 7 2 0 3

11 4 0 4 0 1 2 3

5 2 9 3 0 2 1

1 1 1 0

0 5 1 0

5 1 0 1 0

**Output:**

7 2 1

9 2 2 7

0 16 18 6

10 0

## Test Case Annotation

For the first test case, we have:

From time 0 to 3, no pairs can be formed.

From time 3 to 4, one pair can be formed.

From time 4 to 5, two pairs can be formed.

From time 5 to 6, one pair can be formed.

From time 6 to 10, no pairs can be formed.

So, the total time with  $m = 0$  is 7, the total time with  $m = 1$  is 2, and the total time with  $m = 2$  is 1.

[Home](#) » [Compete](#) » [February Cook - Off Challenge](#) » Icing Crowns

## Icing Crowns Problem Code: CROWNS

The Chef has been asked to decorate a birthday cake for a little child. The parents of the child requested that he draw a crown out of icing on the cake. The Chef has already added some icing blossoms and now it is time to add the crown. If we imagine the blossoms as being points on the x-y plane, he draws this crown according to the following rules. For a point  $p$ , we will let  $x(p)$  and  $y(p)$  denote the x and y coordinates of  $p$ , respectively.

1) The crown is a polygon defined by a sequence of an odd number of points (i.e. icing blossoms)  $p_1, p_2, \dots, p_k$ . The polygon has line segments from  $p_i$  to  $p_{i+1}$  for each  $1 \leq i < k$  as well as a line segment from  $p_k$  to  $p_1$ . If we translate the points so that  $p_1$  is at the origin and then rotate the points about the origin so that  $p_k$  lies on the positive x axis, then  $x(p_i) < x(p_{i+1})$  for each  $1 \leq i < k$  and  $y(p_i) > 0$  for each  $1 < i < k$ . Furthermore, for any index  $1 < i \leq k$  we have that  $y(p_i) > y(p_{i-1})$  if  $i$  is even and  $y(p_i) < y(p_{i-1})$  if  $i$  is odd. To paraphrase, after translating and rotating the points we have that the x coordinates of the points are strictly increasing, the y coordinates of all but the two endpoints are strictly positive, and the difference between y coordinates of successive points alternates between positive and negative.

2) No blossoms may be contained strictly in the interior of the crown.

You can find some pictures of crowns following the sample data for this problem.

Subject to these restrictions, the Chef wants to draw the crown with the largest possible area. Thankfully, his job is made slightly simpler since no three blossoms are collinear.

## Input

The first line consists of a single positive integer  $T \leq 20$  indicating the number of test cases. Each test case begins with an integer  $N$ , between 3 and 50, indicating the number of blossoms. Then  $N$  lines follow, each containing two integers  $X$  and  $Y$  describing the location of a blossom. These integers will have absolute value at most 10,000.

As promised in the problem statement, no three blossoms will be collinear. Also, no two blossoms will have both the same  $X$  and  $Y$  coordinates.

## Output

The output for each test case is a single floating point number with 1 decimal of precision indicating the largest possible area of a crown drawn according to the restrictions above.

## Example

**Input:**

4

4

0 0

0 1

1 0

1 1

6

0 0

100

3 1

8 6

1 7

4 7

5

-4 10

-10 -1

-2 3

8 4

0 5

7

2 -6

6 3

10 8

-3 -9

9 2

0 0

-9 -9

#### Output:

0.5

36.5

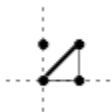
57.0

66.5

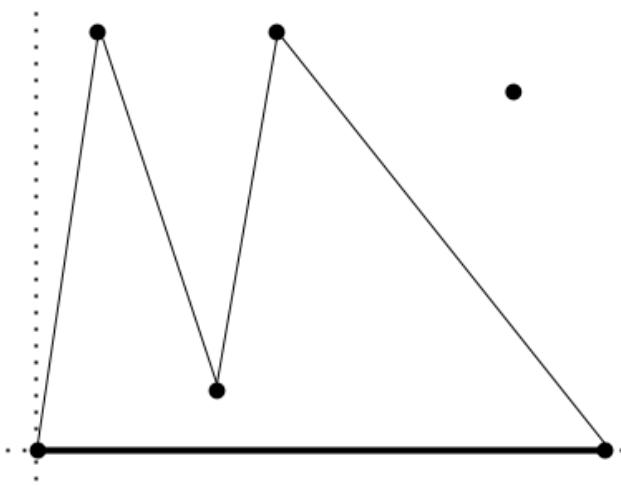
#### Figures

The following figures illustrate the solutions for each test case. The dotted lines indicate the x and y axis. The bold line on a crown indicates that the endpoints of this line are the endpoints  $p_1$  and  $p_k$  of the crown.

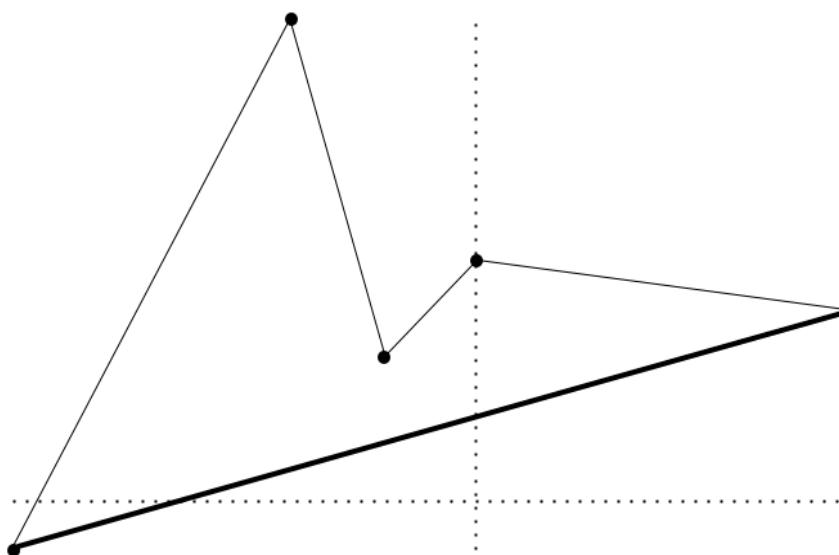
Case 1. Notice that having endpoints  $(1,0)$  and  $(0,0)$  would not form a crown since, after translating and rotating the points so  $(1,0)$  is on the origin and  $(0,0)$  is on the positive  $x$  axis, every other point would share an  $x$  coordinate with one of the endpoints.



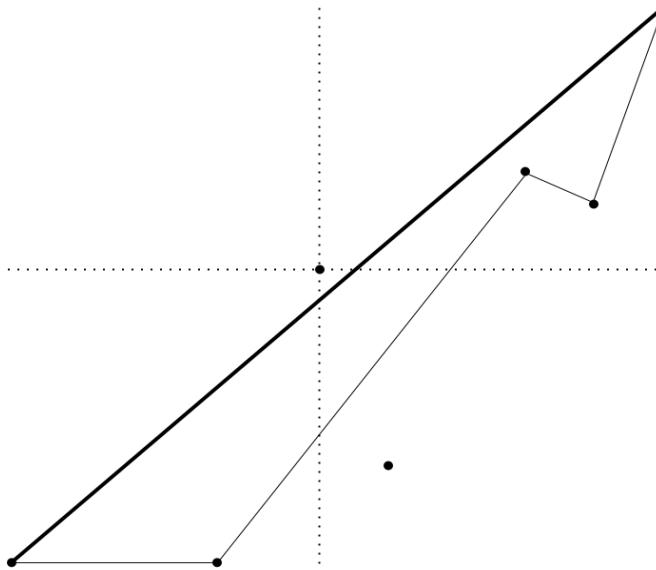
Case 2. We can't add the missing point to the crown since then we would have two successive decreases in the  $y$  coordinates of consecutive points.



Case 3.



Case 4.



MARCH11

[March 2011  
Challenge](#)

12 Mar 2011  
18:00:00

1 day

384

[Home](#) » [Compete](#) » [March 2011 Challenge](#) » K-Unique Sequence

## K-Unique Sequence Problem Code: KUNIQUE

A sequence of  $N$  integers is called  $K$ -unique if, when it is split into  $N/K$  subsequences of  $K$  contiguous elements, each subsequence consists of  $K$  distinct integers.

For example, consider this sequence of 6 integers.

$(2, 10, 2, 8, 3, 6)$

This sequence is 2-unique, because when you split it into 3 subsequences each of 2 contiguous elements as below,

$(2, 10), (2, 8), (3, 6)$ ,

each subsequence consists of 2 distinct integers.

You are given a sequence of  $N$  integers, and a positive integer  $K$ . Find a permutation of the sequence that is  $K$ -unique.

### Input

The first line contains a single integer  $T$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains two integers  $N$  and  $K$ . The next line contains a sequence of  $N$  integers  $a_i$ , where  $a_i$  is the  $i$ -th element of the sequence.

## Output

For each test case, output a single line containing a permutation of the sequence that is K-unique. The elements of the sequence should be separated by a single space. If there are several solutions, output the lexicographically smallest one. If there is no solution, output -1.

## Constraints

- $1 \leq T \leq 5$
- $1 \leq N \leq 50000$
- $1 \leq K \leq N$
- $N$  will be divisible by  $K$
- $0 \leq a_i \leq 1000000000$

## Example

### Input:

3

1 1

42

4 2

4 7 7 7

6 2

2 10 2 8 3 6

### Output:

42

-1

2 3 2 6 8 10

# The Amazing All-In-One Cooking Machine - Challenge

**Problem Code: ALLINONE**

The Chef has recently sold a number of his amazing all-in-one cooking machines! These machines can be used to prepare virtually any recipe you want. Unfortunately, there seems to be a major flaw in many of these devices.

The problem can be fixed very quickly with the right replacement part (it takes essentially 0 time). Since the Chef endorsed these devices so much, he takes it upon himself to perform all of the repairs. He has a list of clients that need their devices repaired as well as a map of the city so he can find the most efficient route between any two clients.

The Chef wants to repair these devices as fast as possible. However, since customer satisfaction is very important, his goal is to visit the clients in some order that minimizes the average time each client waits to be served. Specifically, we let  $T(i)$  denote the amount of time it takes for the Chef to reach client  $i$  from the moment he starts traveling given that he visits all clients that appear before client  $i$  on the list in the given order. Then the Chef's goal is to minimize the average of the waiting times, which is  $(T(1) + T(2) + \dots + T(k))/k$  if there are  $k$  clients.

## Input

The first line contains a single positive integer  $T \leq 30$  indicating the number of test cases to follow. Each test case begins with three integers  $N$ ,  $M$  and  $K$  written on a single line. Here,  $N$  is the number of intersections in the city,  $K$  is the number of clients and  $M$  is the number of streets to follow. The intersections are numbered from 0 to  $N-1$  and the Chef begins at intersection 0.

Following this is a single line containing  $K$  integers describing the locations of the clients. The  $i$ 'th such integer (between 1 and  $N-1$ ) indicates the intersection where the  $i$ 'th client resides. Note that any particular intersection may contain more than one client. Note also that no clients are located at intersection 0.

Then  $M$  lines follow, each describing a street. A street is given by three integers  $U, V$ , and  $D$  where  $U, V$  are distinct integers between 0 and  $N-1$  describing the endpoints of the street and  $D$  is a positive integer that indicates how long it takes to travel from  $U$  to  $V$  along this road. No two intersections will be the endpoint of more than one road. Furthermore, it will always be possible reach all clients from intersection 0.

Bounds:  $2 \leq N \leq 300$ ,  $1 \leq K \leq 10000$ , and  $1 \leq M \leq 2000$ . Furthermore, each street distance  $D$  is between 1 and 1000.

## Output

For each test case you are to output a permutation of the integers from 1 to  $K$  on a single line. This is the order in which the Chef will visit the clients. It does not need to be optimal,

any permutation from 1 to K will be accepted. However, your goal should be to try and make the average waiting times of the clients as small as you possibly can in your solution. Your solution will be scored according to the following scoring mechanism.

## Scoring

Starting from intersection 0, the Chef will visit these clients in the order they appear in the output. If we define  $T(0) = 0$ , then the time it takes to reach client  $i \geq 1$ , say  $T(i)$ , is precisely  $T(i-1)$  plus the shortest time that the Chef can reach the intersection containing client  $i$  in this list from the intersection containing client  $i-1$  in this list. The score for a particular test case is then  $(T(1) + T(2) + \dots + T(K))/K$ . The overall score over all test cases in a test file is simply the sum of the scores of each individual test case.

## Example

**Input:**

3

4 3 3

1 2 3

0 1 1

0 2 1 0

1 3 1 0 0

5 6 1 0

1 2 3 2 3 4 3 2 3 2

0 1 2

0 2 4

0 3 3

3 4 2

4 2 1

2 1 1

3 2 1 0

1 1 1 1 1 1 1 1 2

0 1 5

0 2 1

**Output:**

1 3 2

2 4 8 10 3 5 7 9 1 6

1 2 3 4 5 6 7 8 9 1 0

## Explanation of Sample Data

The optimal solution for the first test case is actually 1 2 3, but 1 3 2 is a valid ordering according to the problem description so it is accepted. Client 1 only waits 1 unit of time to be served. The time to travel from client 1 to client 3 is 100 units of time so client 3 must wait a total of 101 units of time to be served. Then, traveling from client 3 to client 2 takes 111 units of time so client 2 waits a total of 212 units of time to be served. Thus, the average amount of time it takes for a client to be served is then  $(1 + 101 + 212)/3 = 314/3$  and that will be the score for the output on this test case.

The other two test cases are analyzed similarly. In the third test case, all of the clients at location 1 are visited first. It takes 5 units of time to travel to location 1 and serve the first client. However, since we assume it takes 0 time to serve a client we can take care of the rest of the clients at location 1 immediately. So, each client at location 1 waited only 5 units of time to be served meaning the total time waited by all clients at this location is 45 units. Finally, it takes 6 units of time to travel to the sole client at location 2 so the average time waited by the clients is then 56/10.

Overall, the score for the entire set of data (being the sum of the scores of individual test cases) is 116.066667.

## Test Data

Some test data has been hand crafted to make certain heuristics perform poorly and some has been generated randomly from a variety of distributions.

[Home](#) » [Compete](#) » [March 2011 Challenge](#) » Omax

## Omax Problem Code: OMAX

Given a matrix sized  $m \times n$ , we define an O-region as a non-empty sub-block with another non-empty sub-block inside removed, such that the removed sub-block has no common boundaries with the outside. That is, the leftmost column of the removed sub-block must be different from the leftmost column of the outer sub-block, and similarly the rightmost columns, topmost rows, and bottommost rows are different. The removal of the inner sub-block leaves a region which looks like the letter O.

The value of an O-region is the sum of all numbers in the outer sub-block but not in the removed one. Your task is finding the maximal value of all O-regions appearing in the given matrix.

### Input

There are several test cases (fifteen at most), each formed as follows:

- The first line contains two positive integers  $m, n$  ( $3 \leq m, n \leq 77$ ).
- $m$  lines follow, each containing  $n$  integers (each having an absolute value of  $10^5$  at most) describing the matrix.

The input is ended with  $m = n = 0$ .

### Output

For each test case, output on a line an integer which is the respective maximal value found.

### Example

#### Input:

```
3 3
1 1 1
1 3 -1
1 1 1
4 5
1 -2 3 -4 5
-5 4 -3 2 -1
1 -3 5 -7 9
```

-9 7 -5 3 -1

0 0

**Output:**

6

19

[Home](#) » [Compete](#) » [March 2011 Challenge](#) » Squares Game

## Squares Game Problem Code: SQUAGAME

Fat Tony and Fit Tony are playing the square painting game. There are  $n$  squares drawn on a plane. The sides of the squares are parallel to the axes. Squares don't intersect, but some of them can be inside others. In his turn a player can choose any square and paint its internal area black. All squares inside the painted one are also painted black. The player can't paint the squares that were already painted. The loser is the player who can't make a turn. Determine the winner of the game if both players play optimally and Fat Tony's turn is the first. Also if Fat Tony can win the game determine which square he has to paint on his first turn in order to win. If there are many squares which guarantee victory to Fat Tony choose the one with the smallest number.

### Input

The first line of input contains  $t$  - the number of test cases. Each test case starts with  $n$  - the number of squares. Next  $n$  lines consist of three integers each  $x, y, a$  - the coordinates of the lowest left corner of the square and the length of its sides. The squares in the input are numbered from 1 to  $n$  in the order they are listed.

### Constraints

$1 \leq t \leq 10$

$1 \leq n \leq 50000$

The total number of squares over all test cases in one file won't exceed 250000.

$x, y$  won't exceed  $10^8$  in absolute value.

$a$  will be positive and less than  $10^8$ .

### Output

For each test case print "Fat  $x$ ", where  $x$  - is the number of square which needs to be painted on the first turn in order to win (if there are many such square chose the one with the smallest number), if Fat Tony wins or "Fit" if Fit Tony wins.

### Example

**Input:**

2

5

0 0 10

15 15 1

1 1 3

5 5 1

14 14 3

2

1 1 1

3 3 1

**Output:**

Fat 2

Fit

[Home](#) » [Compete](#) » [March 2011 Challenge](#) » Buying Land

## Buying Land Problem Code: BUYLAND

After recent success of cooking reality shows, Chef decided to borrow the idea and host his own "Devil's Kitchen" show. But his ambitions are even bigger. He plans to build a whole complex of hotels, gyms and other entertainment facilities along with one main restaurant where he'll host the show. But first of all, he needs to purchase some land to build on. The country is divided into a grid with  $R$  rows and  $C$  columns of smaller parcels. Chef wants to buy a rectangular piece of land  $H$  parcels high and  $W$  parcels wide. He also has a preferred layout in mind. The restaurant will be located at  $(i,j)$ . For every parcel he defined its preferred height relative to the height of the restaurant parcel. For a particular piece of land, he computes height differences relative to the restaurant and compares them to the preferred layout. This comparison is done as a sum of squared differences over all  $H \times W$  parcels. Given the map of the country, help him out by finding the  $K$  most suitable pieces of land.

## Input

Input starts with a description of the country. It is described from north to south and from west to east. All coordinates are 1-based, which means that the parcel farthest to the north-west is at (1,1). Dimensions R and C are given in the first line, followed by R lines of C integers, describing the heights of individual parcels. Following is the description of the chef's preferred layout, starting with dimensions H and W. Position of the restaurant within the layout (i,j) is given in the next line. The j-th number in the i-th line of preferred layout will always be 0. Input ends with number K, the number of solutions you are required to find. It will be possible to buy the piece of land on at least K different positions. All numbers describing heights will have absolute value at most 30.

## Output

Output exactly K lines containing the K best pieces of land chef can buy, ordered from best to worst. Each piece of land should be described by numbers BR, BC, S, where (BR,BC) represents the position of the upper-left corner of the purchased piece of land and S the corresponding score. In case of a tie, the chef prefers solutions to the north (direction of decreasing row numbers). If there is still a tie, choose the solution farthest to the west (direction of decreasing column numbers).

## Constraints

- $1 \leq R, C \leq 666$
- $1 \leq H \leq (R+1)/2$
- $1 \leq W \leq (C+1)/2$
- $1 \leq K \leq 1000$

## Example

### Input:

```
5 8
6 1 8 6 0 0 8 1
6 5 8 6 8 6 8 1
6 5 8 6 8 6 8 2
6 5 8 6 8 6 8 2
6 5 8 6 8 6 0 2
3 4
-1 2 2 2
```

-1 2 0 2

-1 2 2 2

2 3

4

**Output:**

2 2 8

3 2 8

2 4 11

3 4 75

## Explanation

If the chef decides to buy the land from (3,4) to (5,7), he will end up with the following layout:

0 2 0 2  
0 2 0 2  
0 2 0 -6

Comparing it to the preferred layout gives the penalty of  $1+0+4+0+1+0+0+0+1+0+4+64=75$ , which is 4th best.

[Home](#) » [Compete](#) » [March 2011 Challenge](#) » [Ski Resort](#)

## Ski Resort Problem Code: SKIRES

Ski season is in full swing and the Chef decided it's time to expand his business. He invested into a restaurant at the bottom of a ski slope. Things were going great until one of his employees got injured while skiing home from work. Chef would like to make sure this doesn't happen again. He wants to alter the surrounding landscape to prevent his employees from skiing to their homes in the nearby town. Flattening a large area is obviously out of the question and that's where you come in.

Chef was kind enough to model the surrounding area as a grid with H rows and W columns. Area outside this grid is not suitable for skiing. You are given the elevation of each cell in this grid and the location of his restaurant (R<sub>r</sub>,C<sub>r</sub>) and the town (R<sub>t</sub>,C<sub>t</sub>). As long as there exists a non-increasing path from the restaurant to the town, the employees will not hesitate to ski home. A sequence of cells is called a path if every consecutive pair of cells shares a common edge. You are allowed to **increase** the elevation of any cell (except for the restaurant and town cell) by an integer amount. Find the minimum total number of units of elevation increase which allows Chef to prevent his employees from skiing home.

## Input

The first line contains a single integer  $T$ , the number of test cases. The first line of each testcase contains numbers  $H$  and  $W$ .  $R_r, C_r, R_t, C_t$  are given in the second line. Following lines describe the layout of the ski resort. The  $j$ -th number in the  $i$ -th line represents elevation  $E_{i,j}$  of cell  $(i,j)$ . Top left corner of the grid has coordinates  $(1,1)$ . All numbers in the input are integer.

## Output

For each test case, output minimum total elevation so that chef can achieve his goal or  $-1$  if that is not possible.

## Constraints

- $T \leq 10$
- $1 \leq H, W \leq 50$
- $0 \leq E_{i,j} \leq 200\,000$

## Example

**Input:**

2

4 2

1 1 3 2

10 11

6 6

4 4

3 4

1 5

1 2 1 3

3 4 2 5 5

Output:

4

-1

COOK08	<a href="#">March Cook-off Challenge</a>	20 Mar 2011 21:30:00	2 hours 30 minutes	334
--------	--	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [March Cook-off Challenge](#) » Money Transformation

## Money Transformation Problem Code: MONTRANS

You are standing near a very strange machine. If you put **C** cents in the machine, the remaining money in your purse will transform in an unusual way. If you have **A** dollars and **B** cents remaining in your purse after depositing the **C** cents, then after the transformation you will have **B** dollars and **A** cents. You can repeat this procedure as many times as you want unless you don't have enough money for the machine. If at any point **C**  $> B$  and **A**  $> 0$ , then the machine will allow you to break one of the **A** dollars into **100** cents so you can place **C** cents in the machine. The machine will not allow you to exchange a dollar for **100** cents if **B**  $\geq C$ .

Of course, you want to do this to maximize your profit. For example if **C=69** and you have **9** dollars and **77** cents then after you put **69** cents in the machine you will have **8** dollars and **9** cents (**9.77**  $\rightarrow$  **9.08**  $\rightarrow$  **8.09**). But I should warn you that you can't cheat. If you try to throw away **9** cents before the transformation (in order to obtain **99** dollars and **8** cents after), the machine will sense you are cheating and take away all of your money. You need to know how many times you should do this transformation in order to make a maximum profit. Since you are very busy man, you want to obtain the maximum possible profit in the minimum amount of time.

### Input

The first line contains a single integer **T**  $\leq 40$ , the number of test cases. **T** test cases follow. The only line of each test case contains three nonnegative integers **A**, **B** and **C** where **A**, **B**, **C**  $< 100$ . It means that you have **A** dollars and **B** cents in your purse and you need to put **C** cents in the machine to make the transformation.

### Output

For each test case, output a single line containing the minimal number of times you should do this transformation in order to make a maximal profit. It is guaranteed that the answer is less than **10000**.

### Example

**Input:**

2

9 77 69

98 99 69

**Output:**

4

0

**Explanation**

In the first test we have the following sequence: **9.77, 8.09, 40.07, 38.39, 70.37, 68.69, 0.68**. After last step we have not enough money for further transformations. The maximal profit will be after **4** transformations.

[Home](#) » [Compete](#) » [March Cook-off Challenge](#) » Snake Snaky

## Snake Snaky Problem Code: SNAKY

Chef loves to play the computer game "Snake Snaky". In this game the player controls a long thin creature, resembling a snake that crawls on the rectangular field with width **N** and height **M** surrounded by walls. Snake itself consists of **L** links each of which occupies one cell of the field. Any two consecutive links of the snake must be in adjacent cells of the field and the snake should not have self-intersections (i.e. no two links of the snake can not be located in the same cell).

Snake moves all the time and the player can't stop it, but you can change the direction of its movement using cursor keys. The longer the player can control the snake avoiding self-intersections and collision with the wall the more points he earns. The snake moves in such a way that the first link (head) will occupy a cell adjacent to its previous position in the direction corresponding to the current pressed cursor key. Each successive snake link in a new position occupies the cell which was occupied by the previous link in the old position. Thus the connection of the snake is preserved. The cell which was occupied by the last link (tail) in the old position will be free. All movements made simultaneously so the cell that was occupied by the tail can be occupied by the head in the new position. If no key is pressed, the head of the snake will move in the same direction as in the previous move.

In the middle of a game, the Chef leaves the computer but does not shut down the game so the snake is now moving on its own. Determine the number of movements of the snake before it collides with a wall or some part of its body.

## Input

The first line contains a single integer  $T \leq 40$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains five positive integers  $N, M, x, y, L$ . Here  $N$  and  $M$  are sizes of the field ( $1 \leq N, M \leq 1000$ ),  $x$  and  $y$  are coordinates of the cell which is occupied by the snake tail ( $1 \leq x \leq N, 1 \leq y \leq M$ ) and  $L$  is the snake length ( $2 \leq L \leq N \cdot M$ ). The next line contains the sequence of  $L-1$  symbols corresponding to the direction the snake moved in the last  $L-1$  steps before the Chef left. Symbol 'U' means that the corresponding movement was up, 'D' - down, 'L' - left, 'R' - right. The positive x-axis is directed right and the positive y-axis is directed up. The coordinate of lower left cell is (1, 1). It is guaranteed that the input describes a valid snake that did not intersect itself or collide with a wall before the Chef left. The size of the input file does not exceed **4 MB**.

## Output

For each test case output "**WALL**" if the snake will come into collision with the wall. Otherwise snake will come into collision with part of its body and you must output "**BODY**". After that, print a space followed by the number of movements before the collision.

## Example

Input:

2

10 10 3 2 6

URDDL

6 6 1 6 13

RRRRRDDDLLU

Output:

WALL 2

BODY 1

[Home](#) » [Compete](#) » [March Cook-off Challenge](#) » Grouping Chefs

## Grouping Chefs Problem Code: GROUPING

The Chef's last idea was that some cooks might work better in pairs. But now he thinks that it is better to form one group of Chefs in order to make larger increase of foods quality. All

pairs of employees in this group must be compatible. We call such group friendly. Chef provides you the list of compatible pairs of employees. Also for each employee, the Chef has assigned a number estimating how well the overall quality of the food might increase if this employee will be in the group. Strangely enough, the Chef estimates that picking the  $i$ 'th employee in the group will increase the quality of food prepared in his kitchen by exactly  $2^{c_i}$  where  $c_i$  are nonnegative integer. Also note that Chef has assigned different numbers for his employees. Chef doesn't want to form just one friendly group of employees that make the largest increase of food quality. He wants to have several possibilities of such groups. Namely he wants to know  $K$  best friendly groups by the value of increase of food quality. Note that Chef considers only non-empty groups. Help him to find these groups.

## Input

The first line contains a single integer  $T$  denoting the number of test cases (at most **20**). Each test case begins with three integers  $N$ ,  $M$  and  $K$ . Here,  $N$  is the number of employees (between **1** and **10000**),  $M$  is the number of compatible pairs of employees (between **0** and **20000**) and  $K$  is the number of friendly groups that you need to find (between **1** and **50**). The employees are numbered from **0** to  $N-1$ . The next line contains  $N$  integers  $c_0, c_1, \dots, c_{N-1}$  described in the problem statement. These numbers are different non-negative integers not greater than **10<sup>9</sup>**. The next  $M$  lines describe pairs of compatible employees, one per line. The  $i$ 'th such line contains two distinct integers  $u_i, v_i$  between **0** and  $N-1$ .

No pair of employees will be given more than once in the input. That is, for distinct indices  $i$  and  $j$ , we do not have both  $u_i = u_j$  and  $v_i = v_j$ , nor do we have both  $u_i = v_j$  and  $v_i = u_j$ .

## Output

The output for each test case consists of  $K+1$  lines. Each of the first  $K$  lines consists of the employees indices that are used in  $i$ 'th maximum total value grouping where  $i$  is the number of line. These indices should be given in increasing order with a single space between consecutive numbers. Last line of the output of each test case must be blank. It is guaranteed that there exist at least  $K$  different friendly groups. If there is more than one possible output, then any will do.

## Example

**Input:**

2

4 5 11

4 2 3 1

0 1  
0 2  
1 2  
2 3  
3 1  
1 0 1  
1000000000

**Output:**

0 1 2

0 2

0 1

0

1 2 3

1 2

2 3

2

1 3

1

3

0

# Palindrome Palindrome

Problem Code: PALIPALI

A string is called beautiful if it has the form  $AA^rAA^r$  where  $A$  is some non-empty string and  $A^r$  is the reversed  $A$ . You are given a string  $S$  composed only of lowercase letters of the English alphabet. Find the number of its beautiful substrings. Equal substrings in different positions are considered different. For example, the string **aaaaa** has **2** different substrings because we have beautiful substring **aaaa** in two different positions.

## Input

The first line contains a single integer  $T \leq 10$ , the number of test cases.  $T$  test cases follow. The only line of each test case contains a non-empty string composed only of lowercase letters of the English alphabet. The length of the string is not greater than **100000**.

## Output

For each test case, output a single line containing the number of beautiful substrings of the corresponding string.

## Example

### Input:

6

aaaa

aaaaa

abbaabba

abbaabb

abaaba

xyyxxxxyyxx

### Output:

1

2

```
1
0
0
2
```

[Home](#) » [Compete](#) » [March Cook-off Challenge](#) » Exponentiation Commutativity

## Exponentiation Commutativity Problem Code: EXPCOMM

For a given prime number  $p$  find the number of all pairs  $(m, n)$  of positive integers such that  $1 \leq m, n \leq p^*(p-1)$  and  $p$  divides  $n^m - m^n$ . Output the result modulo **1000000007**.

### Input

The first line contains a single positive integer **T  $\leq 100$** , the number of test cases. **T** test cases follow. The only line of each test case contains a prime number  $p$ , where **2  $\leq p \leq 10^{12}$** .

### Output

For each test case, output a single line containing the answer for the corresponding test case.

### Example

#### Input:

```
2
2
3
```

#### Output:

```
2
14
```

## Explanation

In the first test required pairs are (1, 1) and (2, 2).

In the second test case required pairs are (1, 1), (1, 4), (2, 2), (2, 4), (3, 3), (3, 6), (4, 1), (4, 2), (4, 4), (4, 5), (5, 4), (5, 5), (6, 3) and (6, 6).

APRIL11	<a href="#">April Long Contest</a>	01 Apr 2011 15:00:00	11 days	714
---------	------------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [April Long Contest](#) » Number Game Revisited

## Number Game Revisited Problem Code: NUMGAME2

Alice and Bob play the following game. They choose a number N to play with. The rules are as follows :

1. Bob plays first and the two players alternate.
2. In his/her turn, a player can subtract from N any prime number (including 1) less than N. The number thus obtained is the new N.
3. The person who cannot make a move in his/her turn loses the game.

Assuming both play optimally, who wins the game ?

### Input format:

The first line contains the number of test cases T. Each of the next lines contains an integer N.

### Output format:

Output T lines one for each test case, containing "ALICE" if Alice wins the game, or "BOB" if Bob wins the game.

### Example

#### Sample Input:

2

1

2

**Sample Output:**

ALICE

BOB

**Constraints:** $1 \leq T \leq 1000000$  $1 \leq N \leq 1000000000$ 

Note : For the first test case, Bob cannot make any move and hence Alice wins the game. For the second test case, Bob subtracts 1 from N. Now, Alice cannot make a move and loses the game.

[Home](#) » [Compete](#) » [April Long Contest](#) » Rectangles Counting

## Rectangles Counting Problem Code: RECTCNT

Given N separate integer points on the Cartesian plane satisfying: there is no any three of them sharing a same X-coordinate. Your task is to count the number of rectangles (whose edges parallel to the axes) created from any four of given points.

### Input

There are several test cases (ten at most), each formed as follows:

- The first line contains a positive integer N ( $N \leq 10^5$ ).
- N lines follow, each containing a pair of integers (each having an absolute value of  $10^9$  at most) describing coordinates of a given point.

The input is ended with  $N = 0$ .

### Output

For each test case, output on a line an integer which is the respective number of rectangles found.

### Example

#### Input:

6

7 1

3 5

3 1

1 5

1 1

7 5

0

**Output:**

3

[Home](#) » [Compete](#) » [April Long Contest](#) » Gravel

## Gravel Problem Code: SPREAD

Bob has  $n$  heap(s) of gravel (initially there are exactly  $c$  piece(s) in each). He wants to do  $m$  operation(s) with that heaps, each maybe:

- adding pieces of gravel onto the heaps from  $u$  to  $v$ , exactly  $k$  pieces for each,
- or querying "how many pieces of gravel are there in the heap  $p$  now?".

### Request

Help Bob do operations of the second type.

### Input

- The first line contains the integers  $n, m, c$ , respectively.
- $m$  following lines, each forms:
  - **S u v k** to describe an operation of the first type.
  - **Q p** to describe an operation of the second type.

*(Each integer on a same line, or between the characters **S**, **Q** and the integers is separated by at least one space character)*

### Output

For each operation of the second type, output (on a single line) an integer answering to the respective query (follows the respective Input order).

## Example

### Input:

7 5 0

Q 7

S 1 7 1

Q 3

S 1 3 1

Q 3

### Output:

0

1

2

## Limitations

- $0 < n \leq 10^6$
- $0 < m \leq 250,000$
- $0 < u \leq v \leq n$
- $0 \leq c, k \leq 10^9$
- $0 < p \leq n$

[Home](#) » [Compete](#) » [April Long Contest](#) » Central Point

## Central Point Problem Code: CPOINT

Given a set of  $N$  integer points on the Cartesian plane. Your task is to find an integer point satisfying its sum of distances to  $N$  given points (**S**) is minimum.

### Input

There are several test cases (fifteen at most), each formed as follows:

- The first line contains a positive integer  $N$  ( $N \leq 2,000$ ).
  - $N$  lines follow, each containing a pair of integers (each having an absolute value of  $10^9$  at most) describing coordinates of a given point.
- The input is ended with  $N = 0$ .

## Output

For each test case, output on a line a real number (with exactly 6 decimal places) which is the respective minimum sum **S** found.

## Example

**Input:**

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

**Output:**

```
2.828427
9.640986
```

[Home](#) » [Compete](#) » [April Long Contest](#) » Graphs in Euclidean Space

## Graphs in Euclidean Space Problem Code: L2GRAPH

The Chef is a bit tired of graphs. He has spent far too many days calculating shortest paths, finding bipartite matchings, minimum cuts, and optimizing over NP-hard problems. He needs a break. Unfortunately for him, the food services industry doesn't take breaks.

Once again, the Chef has to navigate through an undirected graph to keep his business going.

In this particular problem, the layout of edges in the graph doesn't matter very much. All that really matters is the distances between pairs of nodes. Given this, you have a great idea that will help the Chef not worry about graphs at all during this latest task. Rather than presenting the graph to the Chef, you will present something else.

After some thought, you have settled on the following. You will present the nodes of the graph as distinct points in  $d$ -dimensional Euclidean space. Each node  $v$  will be represented by a  $d$ -tuple of numbers  $p_v = (x_{v,1}, \dots, x_{v,d})$ . The distance between two points is the usual Euclidean distance in  $d$ -dimensional space. That is, the distance between two points  $p_u$  and  $p_v$  representing nodes  $u$  and  $v$  is the square root of  $(x_{u,1} - x_{v,1})^2 + \dots + (x_{u,d} - x_{v,d})^2$ .

The quality of such an assignment of nodes into points in Euclidean space is measured as follows. Let  $a(u,v)$  be the length of the shortest path connecting node  $u$  to node  $v$  in the graph and let  $b(u,v)$  be the distance between points  $p_u$  and  $p_v$  in Euclidean space. Define  $K$  as the minimum value of  $b(u,v)/a(u,v)$  over all pairs of distinct nodes  $(u,v)$ . Similarly, define  $L$  as the maximum value of  $b(u,v)/a(u,v)$  over all pairs of distinct nodes  $(u,v)$ . The quality of the solution is then  $d*L/K$ .

The idea behind this measurement is as follows. We want the distances between nodes in Euclidean space to be close to their distances in the original graph. So,  $K$  measures the greatest "contraction" between nodes when they are mapped into the Euclidean space. In the same way,  $L$  measures the greatest "stretch" between nodes. We put quotes around "contraction" and "stretch" because it could be that  $K > 1$  or  $L < 1$ . Then the ratio  $L/K$  measures, in some way, the worst case distortion when embedding the nodes in Euclidean space. Finally, lower-dimensional Euclidean spaces are simpler so the final score is scaled by  $d$ .

## Input

The first line consists of a single integer  $T \leq 30$  indicating the number of test cases to follow. Each test case begins with an integer  $n$  between 2 and 300. Then  $n$  lines follow, each containing  $n$  integers. This will be such that the  $j$ 'th integer on the  $i$ 'th row describes the distance of the shortest path between nodes  $i$  and  $j$  in the original graph. For  $i = j$ , this distance will be 0 and for  $i \neq j$  this distance will be at least 1 and at most 10,000. Furthermore, the distance between  $i$  and  $j$  is equal to the distance between  $j$  and  $i$  because the graph is undirected. Since these represent shortest path distances, we also have that the distance from  $i$  to  $j$  is at most the distance from  $i$  to  $k$  plus the distance from  $k$  to  $j$  for any three nodes  $i,j,k$ .

## Output

For each test case you are to output  $n+1$  lines. The first line contains a single integer  $d$  which must be between 1 and 20. Then  $n$  lines follow where the  $i$ 'th such line contains  $d$  integers describing the coordinates of the  $i$ 'th point in Euclidean space. Each of these  $d$

integers must have absolute value at most 1,000,000. Also, each of these n points must be distinct.

To emphasize, any output that conforms to this output specification will be considered valid. The score for this output will be calculated as described above.

## Scoring

Each test case will be given a score  $d*L/K$  where d is the dimension provided in the output and K and L will be calculated in the manner described in the problem statement. The score for all test cases in a test file is simply the sum of the scores of each test case.

## Example

**Input:**

2

3

0 1 2

1 0 2

2 2 0

4

0 1 0 1 3 1 4

1 0 0 1 1 1 8

1 3 1 1 0 1 5

1 4 1 8 1 5 0

**Output:**

2

2 0

```

0 2
0 0
1
0
1
2
3

```

## Explanation Of Sample Data

The provided answers are certainly not optimal for either test case, but they are valid solutions according to the output specification and such output would be accepted. The score for this output would be as follows.

In the first test case, we would have  $K = 1$  since the Euclidean distance from the point representing node 2 to the points representing either 0 or 1 is exactly 2 and the Euclidean distance between the points representing nodes 0 and 1 exceeds 1. We also have  $L = \sqrt{8}/1$  since  $\sqrt{8}$  is the distance between the points representing nodes 0 and 1 and 1 is distance between nodes 0 and 1 in the original graph. Thus, the score for this case is  $d*L/K = 2*\sqrt{8} = 5.6568542$ .

In the second test case, the least ratio of Euclidean distance to original graph distance is between nodes 2 and 3. Their Euclidean distance is 1 and their original graph distance is 15 so  $K = 1/15$ . So, after scaling all Euclidean distance values by  $K$ , the worst graph distance to Euclidean distance ratio is between nodes 0 and 3. Their original graph distance was 14 and their Euclidean distance is only 3, so the value  $L$  is  $3/14$ . Since  $d = 1$ , then the score for this test case is simply  $d*K/L = 1*3/14*15 = 3.2142857$ .

## Test Data

There are roughly three types of test data. Some are hand-crafted to cause certain heuristics to perform poorly. Some are randomly generated from a variety of distributions. Some are known "worst-case" examples of graphs which cannot be represented very well in Euclidean space.

[Home](#) » [Compete](#) » [April Long Contest](#) » Generalized Independent Sets

# Generalized Independent Sets Problem Code: GENIND

Given a graph  $G$  on nodes  $V$  with undirected edges  $E$ , an independent set  $X$  is a subset of  $V$  such that no edge in  $E$  has both endpoints in  $X$ . Finding the size of the largest independent set in a graph is currently a very difficult problem.

Consider the following generalization of an independent set. A function  $f$  that maps each node  $v$  to a non-negative value  $f(v)$  is said to be a partially generalized independent set if, for any edge  $e = (u, v)$ , we have  $f(u) + f(v) \leq 1$ . Unfortunately, this doesn't provide a great generalization. Consider a graph  $G$  where every possible edge exists in  $E$ . The size of a maximum independent set is only 1 whereas assigning  $1/2$  to every value  $f(v)$  has the total fractional value of the  $f(v)$  values being  $|V|/2$ .

So, let's add one more constraint to strengthen the notion of a generalized independent set. A cycle in a graph is a sequence of  $k \geq 2$  distinct nodes  $v_1, v_2, \dots, v_k$  where there is an edge between  $v_i$  and  $v_{i+1}$  for any  $1 \leq i < k$  as well as an edge between  $v_1$  and  $v_k$ . Notice that cycles of length 2 (*i.e.* edges) are permitted by this definition.

If  $C$  is a cycle of  $G$  of length  $|C|$ , then no independent set can have more than  $\text{floor}(|C|/2)$  nodes from  $C$ . Given this, define a generalized independent set as a function  $f$  on nodes in  $V$  that satisfies the following conditions. For any cycle  $C = v_1, v_2, \dots, v_{|C|}$ , we have  $f(v_1) + f(v_2) + \dots + f(v_{|C|}) \leq \text{floor}(|C|/2)$ . Finally, for each node  $v$  we must have  $0 \leq f(v) \leq 1$ .

Your task will be to determine if a given function  $f$  on the nodes of a graph is indeed a generalized independent set. If not, you must produce a cycle that is violated.

## Input

The first line of the input contains an integer  $T \leq 30$  indicating the number of test cases. Each test case begins with two integers  $n$  and  $m$ . Here,  $n$  is the number of nodes and  $m$  is the number of edges in the graph. Say the nodes of the graph are denoted by  $v_0, v_1, \dots, v_{n-1}$ . Following this is a line containing  $n$  floating point values. The  $i$ 'th such value is the value of  $f(v_i)$ . Finally,  $m$  lines follow, each containing two distinct integers  $i, j$  between 0 and  $n-1$  that indicate that there is an edge between  $v_i$  and  $v_j$  in the graph. No edge will be given more than once in the input.

Bounds:  $1 \leq n \leq 500$  and  $0 \leq m \leq 2,000$ . The value of each  $f(v_i)$  will be between 0 and 1 (inclusive) and will be presented with at most three decimals of precision.

## Output

The output for each test case consists of a single line. If the input function is a generalized independent set then the line consists simply of the text "Ok". Otherwise, you should output the text "Bad Cycle: " followed by a sequence of distinct integers between 0 and  $n-1$  that forms a cycle  $C$  for which the sum of the  $f(v)$  values for nodes in  $C$  exceeds  $\text{floor}(|C|/2)$ . Following this sequence is the single integer -1. These integers should be separated by a single space.

If there are multiple such cycles, then any will do. Also, the order in which the nodes of such a cycle are output does not matter beyond ensuring that consecutive nodes in the output are connected by an edge and that the first and last nodes are also connected by an edge.

## Example

Input:

3

4 4

0.5 0.5 0.5 0.5

0 1

1 2

2 3

3 0

6 7

0.5 0.5 0.334 0.5 0.333 0.334

0 1

0 2

1 3

2 3

1 4

2 5

4 5

5 5

1.0 0.0 0.667 0.5 0.0

0 1

1 2

2 3

3 4

4 1

**Output:**

Ok

Bad Cycle: 4 5 2 0 1 -1

Bad Cycle: 3 2 -1

COOK09	<a href="#">April Cook-Off Challenge</a>	24 Apr 2011 21:30:00	2 hours 30 minutes	297
--------	--	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [April Cook-Off Challenge](#) » Internet Media Types

## Internet Media Types Problem Code: MIME2

Many internet protocols these days include the option of associating a media type with the content being sent. The type is usually inferred from the file extension. You are to write a program that facilitates the lookup of media types for a number of files.

You will be given a table of media type associations that associate a certain file extension with a certain media type. You will then be given a number of file names, and tasked to determine the correct media type for each file. A file extension is defined as the part of the file name after the final period. If a file name has no periods, then it has no extension and the media type cannot be determined. If the file extension is not present in the table, then the media type cannot be determined. In such cases you will print "unknown" as the media type. If the file extension does appear in the table (case matters), then print the associated media type.

### Input

Input begins with 2 integers N and Q on a line. N is the number of media type associations, and Q is the number of file names. Following this are N lines, each containing a file extension and a media type, separated by a space. Finally, Q lines, each containing the name of a file.

N and Q will be no greater than 100 each. File extensions will consist only of alphanumeric characters, will have length at most 10, and will be distinct. Media types will have length at

most 50, and will contain only alphanumeric characters and punctuation. File names will consist only of alphanumeric characters and periods and have length at most 50.

## Output

For each of the Q file names, print on a line the media type of the file. If there is no matching entry, print "unknown" (quotes for clarity).

## Sample Input

```
5 6

html text/html

htm text/html

png image/png

svg image/svg+xml

txt text/plain

index.html

this.file.has.lots.of.dots.txt

nodotsatall

virus.exe

dont.let.the.png.fool.you

case.matters.TXT
```

## Sample Output

```
text/html

text/plain

unknown

unknown

unknown
```

unknown

[Home](#) » [Compete](#) » [April Cook-Off Challenge](#) » A Prime Conjecture

## A Prime Conjecture Problem Code: PRIMES2

Chef has been exploring prime numbers lately, and has recently made a conjecture resembling one of Goldbach's conjectures. Chef's conjecture is that any odd number greater than 61 can be expressed as the sum of a prime, a square of a prime, and a cube of a prime. He wants you to help verify his conjecture for small numbers.

Note: negative numbers are never considered to be prime, nor are 0 and 1.

### Input

Input will consist of a series of odd numbers greater than 61 and less than  $10^6$ , one per line, terminated by the number 0 on a line by itself. There will be at most 1000 lines.

### Output

For each odd number in the input, print 3 primes  $P_1$ ,  $P_2$ ,  $P_3$  on a line, where  $P_1 + P_2^2 + P_3^3$  is equal to the number from the input. If no such primes exist, print "0 0 0" instead (quotes for clarity). If there are multiple triplets of primes that satisfy the equation, print any such triplet.

### Sample Input

81

85

155

0

### Sample Output

5 7 3

73 2 2

5 5 5

[Home](#) » [Compete](#) » [April Cook-Off Challenge](#) » The Grand Cook Off

# The Grand Cook Off Problem Code: COOKOFF2

Chef is planning a grand cooking tournament for aspiring chefs in the region. The competition progresses with a number of one-on-one elimination rounds until only one chef remains. For each round, two chefs are chosen at random for a cook-off. The loser of the cook-off is eliminated, while the winner moves on. Additionally, each chef begins with a positive number of prize tokens (some chefs begin with more than others, based on their performance in a qualification round). The winner of each cook-off receives all of the loser's prize tokens, and the loser is awarded a prize based on the number of prize tokens he/she had accrued.

The final cook-off always receives the most publicity. Your task is to compute the expected difference between the number of prize tokens of the final two competitors before the final cook-off. You will be given the number of tokens each chef begins with.

## Input

Input begins with an integer  $T$ , the number of test cases (at most 50). Each test case consists of a single line, beginning with an integer  $C$ , the number of chefs in the competition, followed by  $C$  positive integers each describing the number of prize tokens a chef begins with. The total number of prize tokens among all chefs will be between 1 and 100, inclusive.

## Output

For each test case, output the expected difference in prize tokens, rounded to 6 places after the decimal point.

## Sample Input

3

3 1 1 1

3 1 1 2

4 10 10 10 30

## Sample Output

1.000000

1.333333

26.666667

## Frosting Cupcakes Problem Code: MUFFINS2

Now that Chef has streamlined his cupcake baking procedure, he's turning his attention toward frosting the cupcakes. He recently purchased a machine that produces frosting. Each cupcake requires one unit of frosting, and the frosting machine requires  $K^2$  units of energy to produce  $K$  units of frosting in one minute ( $K$  need not be an integer). Cupcakes arrive in batches, once per minute, needing frosting. The cupcakes must be frosted within a minute of when they arrive. The frosting machine also has a reserve unit, so it can produce extra frosting to be used later. The reserve unit has a capacity of  $S$  units of frosting, and is initially empty. The machine may produce more units of frosting than is currently demanded, and the excess will be added to the reserve unit. Or, the machine can produce fewer units of frosting than is currently demanded, drawing the remainder of what is needed from the reserve. At all times the reserve unit must have between 0 and  $S$  units of frosting.

Chef has a schedule ahead of time of exactly how many cupcakes will arrive each minute needing frosting, and can create a frosting production schedule for the machine that will optimize its energy usage. The machine may produce a different amount of frosting each minute. Given the cupcake schedule, calculate the minimum energy needed to frost all the cupcakes divided by the total number of cupcakes.

### Input

Input will begin with an integer  $T$ , the number of test cases. Each test case begins with 2 integers  $N$  and  $S$  on a line, representing the number of minutes the machine will operate, and the capacity of the storage unit, respectively. Following is a line with  $N$  space-separated integers  $C_0 \dots C_{N-1}$ , where  $C_i$  is the number of cupcakes arriving at time  $i$ .

### Output

For each test case, output on a single line the minimum total energy consumption of the machine divided by the total number of cupcakes, rounded to 5 places after the decimal point.

### Sample input

4

5 0

1 2 3 4 5

5 1

0 0 0 0 2

5 2

0 0 0 0 2

5 2

2 0 0 0 0

## Sample output

3.66667

0.62500

0.40000

2.00000

## Constraints

$T < 50$

$1 \leq N \leq 30000$

$0 \leq S \leq 30000$

$0 \leq C_i \leq 20000$

At least one  $C_i$  will be non-zero

[Home](#) » [Compete](#) » [April Cook-Off Challenge](#) » Product of Digits Again

# Product of Digits Again Problem Code: DIGITS2

Chef has called on you many times in the past to solve unusual tasks that seem irrelevant to cooking, and today is no different. Today Chef demands that you find integers whose product of digits is equal to a given integer, but in bases besides 10. Given a string  $S$ , you are to determine the smallest positive integer  $I$  such that there exists a base  $B > 1$  for which the product of the digits of  $I$  gives the integer represented by  $S$ . For example, if  $S = "11"$ , then the smallest  $I$  is 8, because the base 3 representation of  $I$  is 22, and in base 3  $2^2 = 11$ .

Formally, your task is this: find the smallest integer  $I$  such that there exists an integer  $B > 1$  and integers  $A_i$  for which:

- $A_n * B^n + A_{n-1} * B^{n-1} + \dots + A_1 * B + A_0 = I$
- $0 \leq A_i < B$  for all  $i$
- $A_n \neq 0$
- $A_n * A_{n-1} * \dots * A_1 * A_0 = S_m * B^m + S_{m-1} * B^{m-1} + \dots + S_1 * B^1 + S_0$ , where  $S_m$  is the leftmost character of  $S$ , and  $S_0$  is the rightmost character of  $S$ .
- $0 \leq S_i < B$  for all  $i$

## Input

Input begins with an integer  $T$ , the number of test cases (at most 50). Each test case consists of a single line containing a string  $S$  comprised of 0-9 and A-Z characters. Characters A-Z represent the decimal values 10-35, respectively.  $S$  will have between 1 and 4 characters, and the first character will not be 0.

## Output

For each test case, output a single integer on a line, giving the minimum value of  $I$  (printed in decimal). It is guaranteed that  $I$  will fit in a 64-bit integer.

## Sample Input

```
4
9
11
7Q2
PIES
```

## Sample Output

```
9
8
15227
7845414
```

MAY11	<a href="#">May Long Contest 2011</a>	01 May 2011 15:00:00	10 days	608
-------	---------------------------------------	-------------------------	---------	-----

[Home](#) » [Compete](#) » [May Long Contest 2011](#) » Tree Product

## Tree Product Problem Code: TPRODUCT

Given a complete binary tree with the height of  $H$ , we index the nodes respectively top-down and left-right from 1. The  $i$ -th node stores a positive integer  $V_i$ . Define  $P_i$  as follows:  $P_i = V_i$  if the  $i$ -th node is a leaf, otherwise  $P_i = \max(V_i * P_L, V_i * P_R)$ , where  $L$  and  $R$  are the indices of the left and right children of  $i$ , respectively. Your task is to calculate the value of  $P_1$ .

## Input

There are several test cases (fifteen at most), each formed as follows:

- The first line contains a positive integer  $H$  ( $H \leq 15$ ).
- The second line contains  $2^H - 1$  positive integers (each having a value of  $10^9$  at most), the  $i$ -th integer shows the value of  $V_i$ .  
The input is ended with  $H = 0$ .

## Output

For each test case, output on a line an integer which is the respective value of  $P_1$  found, by modulo of 1,000,000,007.

## Example

**Input:**

```
2
1 2 3
3
3 1 5 2 6 4 7
0
```

**Output:**

```
3
105
```

**Explanation:**

The second test case is constructed as follows:

```
3
/ \
/   \
```

```

1      5
/ \   / \
2   6 4   7

```

[Home](#) » [Compete](#) » [May Long Contest 2011](#) » The Great Wall of Byteland

## The Great Wall of Byteland Problem Code: BWALL

In order to protect the southern border of Bytelandian Empire against intrusions by various nomadic groups, the Emperor of Byteland has decided to build a wall along the southern border. The best architect Johny is recruited for the task.

In order to minimise the cost of raw material, Johny is restricted to use only following two kinds of building blocks:

```

X   #
XX

```

The height of the wall is fixed and is 2 units, but the length of the wall varies. As a part of his job Johny needs to find out the number of ways he can construct the wall using above two types of building blocks where the length of the wall is specified. Write a program to help Johny.

### Input Format:

The first line contains the number of test cases T followed by T lines. Each of the next lines contains an integer N representing the length of the wall.

### Output Format:

Print T lines one for each test case, containing an integer that represents the number of ways of constructing the wall, modulo 1000000007.

### Example:

#### Sample Input:

```

3
7
2
13

```

**Sample Output:**

655

5

272767

**Constraints:**

1&lt;=T&lt;=1000

1&lt;=N&lt;=10^9

**Explanation of 2nd Test Case**

N=2

The wall can be constructed in following 5 ways:

##

##

X#

XX

#X

XX

XX

X#

XX

#X

## Bickering Cooks Problem Code: BICKER

The cooks in the Chef's kitchen are bickering quite a bit today. Frankly, the Chef can't stand it and he wants to send some of the cooks to his other restaurant to help keep the peace. Unfortunately, some cooks work well in pairs and splitting this pair up will affect the overall quality of the food prepared in the kitchen.

More formally, for each pair of cooks  $(i,j)$  the Chef is able to quantify how disruptive their bickering is by a non-negative value  $d(i,j)$ . The Chef is also able to quantify the overall quality decrease in the food if the pair  $(i,j)$  is split up by a non-negative value  $q(i,j)$ .

He wants to split up (partition) the cooks into two non-empty groups  $S$  and  $T$ . This means every cook  $i$  is in either  $S$  or  $T$ , but not both. The value  $d(S,T)$  is then the total sum of the values  $d(i,j)$  over pairs  $(i,j)$  with  $i$  in  $S$  and  $j$  in  $T$  or  $i$  in  $T$  and  $j$  in  $S$ . The value  $q(S,T)$  is similarly defined as the total sum of the values  $q(i,j)$  over pairs  $(i,j)$  with  $i$  in  $S$  and  $j$  in  $T$  or  $i$  in  $T$  and  $j$  in  $S$ .

Let  $q_{\text{Tot}}$  denote the total of all  $q(i,j)$  values and let  $d_{\text{Tot}}$  denote the total of all  $d(i,j)$  values. Finally, we say the cost effectiveness of the partition  $S,T$  is then  $(q(S,T)/d(S,T)) * (d_{\text{Tot}}/q_{\text{Tot}})$  which measures the fraction of all  $q(i,j)$  values represented in  $q(S,T)$  divided by the fraction of all  $d(i,j)$  values represented in  $d(S,T)$ . The Chef wants to find such a partition  $S,T$  to minimize the cost effectiveness. The idea is that the Chef wants to separate many disruptive pairs while somehow minimizing the effect it has on the food.

### Input

The first line contains a single integer  $T$  between 1 and 30 indicating the number of test cases. Each test case begins with three integers  $N$ ,  $D$ , and  $Q$ . This means there are  $N$  cooks with precisely  $D$  pairs having  $d(i,j) > 0$  and precisely  $Q$  pairs having  $q(i,j) > 0$ . The cooks are identified with the integers 1 through  $N$ . Then  $D$  lines follow, each containing three integers  $i,j,v$ . Such a line means  $d(i,j) = v$ . Following this are  $Q$  lines, each containing three integers  $i,j,w$ . This means  $q(i,j) = w$ . Bounds:  $2 \leq N \leq 500$  and  $1 \leq D,Q \leq 10,000$ . In each line  $i,j,v$  describing a disruptive pair we have  $1 \leq i < j \leq N$  and  $1 \leq v \leq 10,000$ . Similarly, in each of the last  $Q$  lines  $i,j,w$  we have  $1 \leq i < j \leq N$  and  $1 \leq w \leq 10,000$ . No pair of cooks  $(i,j)$  will appear more than once among the disruptive pairs and neither will they appear more than once in the last  $Q$  lines of the test case. It might be that a pair  $(i,j)$  appears in the list of  $D$  disruptive pairs as well as in the list of  $Q$  pairs of cooks that work well together (a love/hate relationship).

If any pair  $i,j$  does not appear among the  $D$  disruptive pairs, then you are to assume that  $d(i,j) = 0$ . Similarly, if any pair  $i,j$  does not appear among the  $Q$  pairs that work well together, then you are to assume that  $q(i,j) = 0$ .

### Output

The output for each test case consists of a single line. The first integer on this line, say  $k$ , denotes the size of  $S$  in a partition of the  $N$  cooks into two non-empty groups  $S, T$ . Of course, this means  $1 \leq k \leq N-1$ . Following  $k$  should be  $k$  integers between 0 and  $N-1$  in strictly increasing order. These  $k$  integers describe the cooks in group  $S$ .

The only further restriction on the output is that  $d(S, T)$  must be non-zero. Otherwise it is pointless to split the cooks into these groups. Any output that conforms to these specifications is considered valid and will be assigned a score according to the scoring mechanism described below. Of course, lower scores will rank higher among the valid submissions.

## Example

**Input:**

2

3 3 3

1 2 1

1 3 2

2 3 3

1 2 3

1 3 1

2 3 2

5 1 6

1 5 1

1 2 1

2 3 1

3 4 1

4 5 1

1 5 1

2 4 1

**Output:**

1 2

3 1 2 5

**Scoring**

Say the output for a test case describes a set  $S$ . Then  $T$  will automatically be equal to the cooks that are not in  $S$ . The score for the test case is then  $(q(S,T)/d(S,T))*(d_{\text{Tot}}/q_{\text{Tot}})$ . The overall score over all test cases in a single file is the sum of the scores for each individual test case. Finally, there are multiple input files and the final score is the average of the scores over all files.

**Explanation of Sample Data**

In the first test case, the pairs (1,2) and (2,3) are separated so  $q(S,T) = 5$  and  $d(S,T) = 4$ . Since  $d_{\text{Tot}} = q_{\text{Tot}} = 6$ , then the score for the first test case is  $(5/4)*(6/6) = 5/4$ .

In the second test case, the only bickering pair (1,4) is separated so  $d(S,T) = 1$ . The pairs that work well together that are separated are (2,3) and (4,5) so  $q(S,T) = 2$ . In this case,  $d_{\text{Tot}} = 1$  and  $q_{\text{Tot}} = 6$  so, the score for the second test case is exactly  $(2/1)*(1/6) = 1/3$ .

**Test Cases**

There are a few different types of test cases. Some are test cases that are generated randomly according to various distributions. Some are hand-crafted to defeat some simple approaches. Finally, some are test cases that are considered "difficult to solve".

[Home](#) » [Compete](#) » [May Long Contest 2011](#) » Pruning Trees

**Pruning Trees** Problem Code: PRUNING

You have a tree (an acyclic, connected graph) with edge weights, but it is far too big. You are to prune the tree by deleting some edges so that the following property holds. For each connected component  $C$  of the pruned tree, there is some node  $r_C$  such that for every node  $v$  in  $C$ , the total weight of the edges on the path between  $v$  and  $r_C$  is at most some given value  $d$ .

You happen to like heavy trees so you want to delete the minimum possible total edge weight to satisfy this property.

**Input**

The first line of each test case is an integer  $T \leq 40$  indicating the number of test cases to follow. The first line of each test case contains two integers  $n$  and  $d$  that indicate there are  $n$  nodes in the tree and  $d$  is as specified in the problem description. Suppose the nodes in the tree are indexed by integers between 0 and  $n-1$ . Then  $n-1$  lines follow, each containing three integers  $u, v$ , and  $w$ . This indicates that there is an edge of weight  $w$  between nodes  $u$  and  $v$ . The input will be such that the graph is connected with no cycles. No edge will be given more than once and no edge in the input will be a loop (i.e.  $u \neq v$  for each input edge).

Bounds:  $1 \leq n \leq 100$ ,  $0 \leq d \leq 10^9$ , and, for each edge  $u, v, w$  in the input,  $0 \leq u < v \leq n-1$  and  $1 \leq w \leq 10^7$ .

## Output

The output for each test case consists of a single line containing a single integer denoting the maximum total weight of edges that can remain after deleting some edges so that the remaining graph satisfies the property described in the problem description.

## Example

Input:

3

5 10

0 1 10

1 2 9

1 3 10

3 4 9

10 1

0 1 1

0 2 1

1 3 1

1 4 1

2 5 1

2 6 1

2 7 1

6 8 1

6 9 1

7 2

0 1 1

0 2 2

1 3 1

1 4 2

2 5 2

2 6 7

**Output:**

29

7

7

[Home](#) » [Compete](#) » [May Long Contest 2011](#) » Sum Divides Product

## Sum Divides Product Problem Code: PRDIVSUM

For a given positive integer **N** find the number of all pairs **(a, b)** of positive integers such that **1 <= a < b <= N** and the sum **a + b** divides the product **a \* b**.

### Input

The first line contains a single positive integer **T <= 5**, the number of test cases. **T** test cases follow. The only line of each test case contains a positive integer **N <= 10<sup>9</sup>**.

### Output

For each test case, output a single line containing the answer for the corresponding test case.

## Example

**Input:**

2

2

15

**Output:**

0

4

## Explanation

In the second test case required pairs are (3, 6), (4, 12), (6, 12) and (10, 15).

[Home](#) » [Compete](#) » [May Long Contest 2011](#) » Harmonious Decorating

## Harmonious Decorating Problem Code: HARMONY

The Chef has formed a very nice planar graph out of icing on his latest cake. The nodes of the graph are represented by distinct points on the cake and every edge is drawn as a straight line segment between its two endpoints. For each edge  $e = (u, v)$  the line segment connecting  $u$  and  $v$  does not pass through any other point representing some other node. Finally, if two line segments representing different edges  $e, e'$  share a common point on the cake, it must be that this common point is in fact a common endpoint  $v$  of both  $e$  and  $e'$  (i.e. no two edges cross).

Not only is the drawing of the planar graph very nice, the graph itself has some nice properties. For starters, there are no loops in the graph and no two nodes are connected by more than one edge. The graph is also connected and it is impossible to disconnect the graph by removing a single edge.

Drawing the planar graph divides the plane into "faces". More specifically, consider the subset of the plane  $F = \{(x, y) : (x, y) \text{ is not a point and } (x, y) \text{ does not lie on an edge}\}$ . The points in  $F$  are partitioned into faces where two points  $P, Q$  in  $F$  lie on the same face if and

only if it is possible to draw a path from P to Q without lifting the pencil and without touching any edge or point (i.e. the path stays contained in F).

As is well-known in the theory of planar graphs, there is exactly one face that is "unbounded" or "outside" of the drawing. All other faces are called "interior" faces. Note that if a graph G is simply a cycle, then there is also only one interior face. For each cycle C and each face F, we say that F is "contained in" C if F is a subset (as a set of points in the plane) of the only interior face of the planar drawing obtained by erasing all edges and nodes of G except for those that appear in C.

The Chef wants to add an icing blossom on some of the edges in the drawing of the graph in a harmonious manner. He wants to do this so that for each cycle C, the parity of the number of blossoms on edges of C is equal to the parity of the number of faces contained in C. Your task is to determine if this is possible and, if so, describe which edges should receive a blossom.

## Input

The first line contains a single positive integer **T** indicating the number of test cases. Each test case begins with two integers **N** and **M**. Here, **N** indicates the number of nodes in the graph and **M** indicates the number of edges.

The nodes of the graph are numbered from 0 to **N**-1. Then **N** lines follow, each consisting of two integers **x,y** describing the coordinates of the points (if we view the surface of the cake as the Euclidean plane). Then **M** lines follow, each consisting of two integers **u,v** describing the endpoints of one of the **M** edges. Here,  $0 \leq u, v, < N$  are the indices of the nodes that are connected by the edge. Recall that the graph is drawn on the cake by connecting the locations of the points representing the endpoints of an edge by a straight line segment.

Constraints:  $3 \leq N \leq 10,000$  and  $N \leq M \leq 3N - 6$ . The integers describing the coordinates of the points are at most 1,000,000,000 in absolute value. No two nodes will be located at the same point, no edge has equal endpoints, and no edge will be given more than once in the input (this also means that if **u,v** is given, then **v,u** will not be given in a different line). Finally, the graph is connected and cannot be disconnected by deleting a single edge.

## Output

For each test case, you are to output a single line. If it is not possible to add blossoms to some edges of the cake in the harmonious manner, then this line should contain the text "impossible" (without the quotes). Otherwise, the line should consist of **M** 0 or 1 values where the *i*'th such value is a 1 if and only if the *i*'th edge in the input should receive a blossom. No other integers than 0 or 1 may be output on this line.

## Example

Input:

3

3 3

0 0

0 1

1 0

0 1

1 2

2 0

5 6

0 0

1 1

1 -1

-1 -1

-1 1

0 1

1 2

2 3

3 4

4 0

4 1

6 9

-1 -1

0 1

1 -1

-2 -2

2 -2

0 2

0 1

1 2

0 2

3 4

3 5

5 4

0 3

4 2

5 1

**Output:**

1 0 0

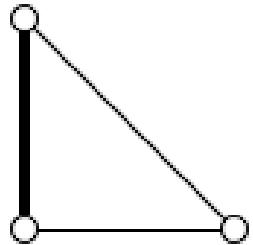
0 1 0 0 0 1

0 0 1 0 1 1 0 0 0

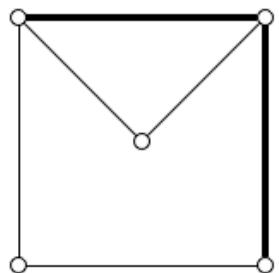
## Illustration Of Sample Data

The following three images illustrate the sample data. The thick edges indicate edges that receive a blossom. A red cycle has been highlighted in the last example. Note, in particular, that there are three faces contained in this cycle and there is one edge lying on this cycle that receives a blossom.

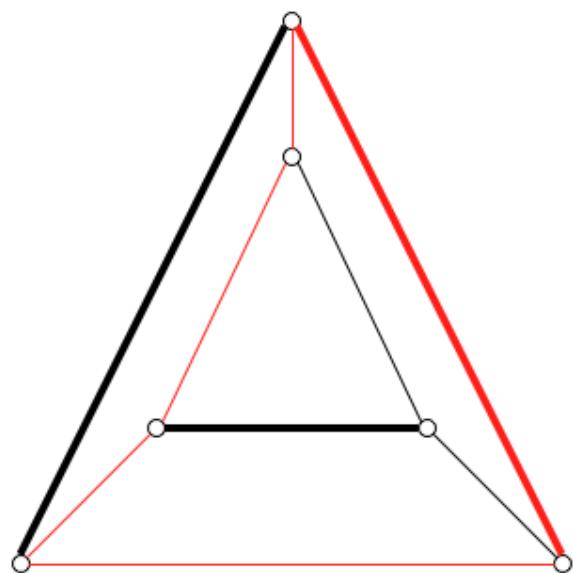
## Test Case 1:



## Test Case 2:



### Test Case 3:



# Popular Rice Recipe Problem Code: TIDRICE

Did you know that there are over 40,000 varieties of Rice in the world ? There are so many dishes that can be prepared with Rice too. A famous chef from Mumbai, Tid Gusto prepared a new dish and named it 'Tid Rice'. He posted the recipe in his newly designed blog for community voting, where a user can plus (+) or minus (-) the recipe. The final score is just the sum of all votes, where (+) and (-) are treated as +1 and -1 respectively. But, being just a chef ( and not a codechef ) he forgot to take care of multiple votes by the same user.

A user might have voted multiple times and Tid is worried that the final score shown is not the correct one. Luckily, he found the user logs, which had all the **N** votes in the order they arrived. Remember that, if a user votes more than once, the user's previous vote is first nullified before the latest vote is counted ( see explanation for more clarity ). Given these records in order ( and being a codechef yourself :) ), calculate the correct final score.

## Input

First line contains **T** ( number of testcases, around 20 ). **T** cases follow. Each test case starts with **N** ( total number of votes,  $1 \leq N \leq 100$  ). Each of the next **N** lines is of the form "userid vote" ( quotes for clarity only ), where userid is a non-empty string of lower-case alphabets ( 'a' - 'z' ) not more than 20 in length and vote is either a + or - . See the sample cases below, for more clarity.

## Output

For each test case, output the correct final score in a new line

## Example

### Input:

3

4

tilak +

tilak +

tilak -

tilak +

3

ratna +

shashi -

ratna -

3

bhavani -

bhavani +

bhavani -

**Output:**

1

-2

-1

**Explanation**

Case 1 : Initially score = 0. Updation of scores in the order of user tilak's votes is as follows,

( + ): +1 is added to the final score. This is the 1st vote by this user, so no previous vote to nullify. score = 1

( + ): 0 should be added ( -1 to nullify previous (+) vote, +1 to count the current (+) vote ). score = 1

( - ) : -2 should be added ( -1 to nullify previous (+) vote, -1 to count the current (-) vote ). score = -1

( + ): +2 should be added ( +1 to nullify previous (-) vote, +1 to count the current (+) vote ). score = 1

[Home](#) » [Compete](#) » [May Cook-off 2011](#) » Distribute idlis Equally

## Distribute idlis Equally Problem Code: EQIDLIS

Did you know that Chwee kueh, a cuisine of Singapore, means water rice cake ? Its a variety of the most popular South Indian savory cake, only that we call it here idli :). The tastiest idlis are made in Chennai, by none other than our famous chef, Dexter Murugan. Being very popular, he is flown from Marina to Miami, to serve idlis in the opening ceremony of icpc world finals ( which is happening right now ! ).

There are **N** students and they are initially served with some idlis. Some of them are angry because they got less idlis than some other. Dexter decides to redistribute the idlis so they all get equal number of idlis finally. He recollects his father's code, "Son, if you ever want to redistribute idlis, follow this method. While there are two persons with unequal number of idlis, repeat the following step. Select two persons A and B, A having the maximum and B having the minimum number of idlis, currently. If there are multiple ways to select A (similarly B), select any one randomly. Let A and B have P and Q number of idlis respectively and  $R = \text{ceil}((P - Q) / 2)$ , Transfer R idlis from A to B."

Given the initial number of idlis served to each student, find the number of times Dexter has to repeat the above step. If he can not distribute idlis equally by following the above method, print -1.

## Notes

- $\text{ceil}(x)$  is the smallest integer that is not less than x.

## Input

First line contains an integer T ( number of test cases, around 20 ). T cases follows. Each case starts with an integer N (  $1 \leq N \leq 3000$  ). Next line contains an array A of N integers separated by spaces, the initial number of idlis served (  $0 \leq A[i] \leq N$  )

## Output

For each case, output the number of times Dexter has to repeat the given step to distribute idlis equally or -1 if its not possible.

## Example

### Input:

3

4

1 2 2 3

2

1 2

7

1 2 3 4 5 6 7

**Output:**

```

1
-1
3

```

**Explanation:**

Case 1 : { 1, 2, 2, 3}. Maximum 3, Minimum 1.  $R = \text{ceil}((3-1)/2) = 1$ . Transfer 1 idli from person having 3 idlis to the person having 1 idli. Each of them has 2 idlis now, so just 1 step is enough.

Case 2 : {1,2}  $R = \text{ceil}((2-1)/2) = 1$ . {1,2}  $\rightarrow$  {2,1}  $\rightarrow$  {1,2} .... they can never get equal idlis :(

Case 3 : Sorted arrays, in the order encountered {1, 2, 3, 4, 5, 6, 7}  $\rightarrow$  {2, 3, 4, 4, 4, 5, 6}  $\rightarrow$  {3, 4, 4, 4, 4, 5}  $\rightarrow$  {4, 4, 4, 4, 4, 4}

**Note**There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted. Time limit on each test set is 3 sec

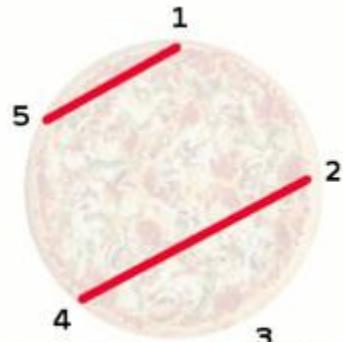
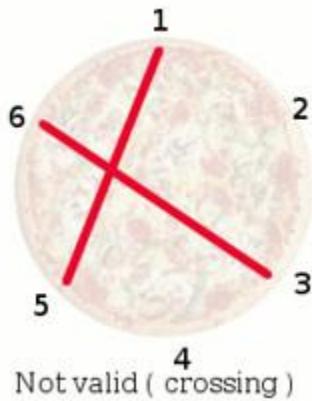
[Home](#) » [Compete](#) » [May Cook-off 2011](#) » Socializing Game around Pizza

## Socializing Game around Pizza Problem Code: BIGPIZA

Did you know that the people of America eat around 100 acres of pizza per day ? Having read this fact on internet, two chefs from the Elephant city, Arjuna and Bhima are set to make pizza popular in India. They organized a social awareness camp, where **N** people ( other than these two ) sit around a large pizza. To make it more interesting, they make some pairs among these N people and the two persons in a pair, feed each other.

Each person should be a part of at most one pair and most importantly, to make feeding easy, any two pairs should not cross each other ( see figure for more clarity ). Arjuna and Bhima decided to play a game on making the pairs. In his turn, a player makes a pair ( selects two persons, as long as its valid ) and this pair start feeding each other. Arjuna and Bhima take turns alternately, by making a pair in each turn, and they play optimally ( see Notes for more clarity ). The one who can not make a pair in his turn, loses. Given N,

find who wins the game, if Arjuna starts first.



## Notes

- 'Optimally' means, if there is a possible move a person can take in his turn that can make him win finally, he will always take that. You can assume both are very intelligent.

## Input

First line contains an integer  $T$  ( number of test cases, around 1000 ). Each of the next  $T$  lines contains an integer  $N$  ( $2 \leq N \leq 10000$  )

## Output

For each test case, output the name of the winner ( either "Arjuna" or "Bhima" ( without quotes ) ) in a new line.

## Example

**Input:**

4

2

4

5

6

**Output:**

Arjuna

Arjuna

Bhima

Arjuna

### Explanation:

Let the people around the table are numbered 1, 2, ..., N in clock-wise order as shown in the image

Case 1 : N = 2. Only two persons and Arjuna makes the only possible pair (1,2)

Case 2 : N = 4. Arjuna can make the pair (1,3). Bhima can not make any more pairs ( without crossing the pair (1,3) )

Case 3 : N = 5. No matter which pair Arjuna makes first, Bhima can always make one more pair, and Arjuna can not make any further

[Home](#) » [Compete](#) » [May Cook-off 2011](#) » Packing the Golden Triangles

## Packing the Golden Triangles Problem Code: PACKBAND

Did you know that the yummy golden triangle was introduced in India as early as 13th century ? By the way, I'm referring to the popular South Asian snack, Samosa. I guess its hard to code while thinking of Samosa, especially if you are very hungry now ; so lets not get in to any recipe or eating game.

You have **N** boxes of Samosas, where each box is a cube. To pack a box, you need to use a rubber band ( pseudo-circular, elastic band ) by placing it around the box ( along 4 faces of the cube ). A (R1,R2)-rubber band has initial radius **R1** and it can stretch at max to radius **R2** without breaking. You can pack a cubical box of side length **L** using a rubber band of circumference  $4 * L$  ( see Notes for clarity). Given **M** rubber bands along with their initial radius and max radius, we need to match ( assign ) some rubber bands to boxes. A box needs at least one rubber band to pack it and of course, each rubber band can be used to pack at most one box. Find the maximum number of boxes we can pack.

### Notes

A pseudo-circular rubber band having a radius R has circumference of  $2 * K * R$  , where K is a constant =  $22 / 7$ . So, a (R1,R2) rubber band can be used to pack a cubical box of side length L, only if  $2 * K * R1 \leq 4 * L \leq 2 * K * R2$

### Input

First line contains an integer T ( number of test cases, around 20 ). T cases follow. Each test case starts with an integer N ( number of boxes,  $1 \leq N \leq 1000$  ). Next line contains

N integers, the side lengths L of the N boxes (  $1 \leq L \leq 100,000,000$  ). Next line contains an integer M ( number of rubber bands,  $1 \leq M \leq 1000$  ). Each of the next M lines contains two integers R1 R2 (  $1 \leq R1 \leq R2 \leq 100,000,000$  ), separated by a space.

## Output

For each test case, output the maximum number of boxes you can pack, in a new line.

## Example

Input:

1

4

10 20 34 55

4

7 14

7 21

14 21

7 35

Output:

2

**Explanation:** Only 1 test case here, and a possible answer can be, using (7,14) rubber band to pack box  $L = 10$ , and using (7,35) rubber band to pack box  $L = 55$ . We cannot pack more than 2 boxes. [Home](#) » [Compete](#) » [May Cook-off 2011](#) » Preferred Cooking Schools

## Preferred Cooking Schools Problem Code: CKSCHOOL

Did you know that the famous chef Ramsay gets paid around 23 million US dollars a year ? Yes, you heard it right, that's over 100 crore Indian Rupees a year. Having read this in News paper, Mr.Nagulu from Hyderabad decides to get his two sons join a Culinary school. There are **N** schools in total, with varying distances **D** from his home and with varying fee **F** they collect. As the number of schools is huge, he would like to prepare a list of preferred schools before he starts his school hunt.

Mr.Nagulu prefers school X over school Y if X is at a distance less than or equal to Y and X collects fees less than or equal to Y. Also, a school cannot be preferred over itself. Consider a set A of schools: a school X is termed 'Special A' if there is no school in A that is preferred over X. 'Max Special A' is the maximal subset of A such that every school in 'Max Special A' is 'Special A'. He wants to consider only some schools, based on how much fee he can pay and at most how far he can send his sons. But, he has a lot of such queries for you, each of them specifying 3 conditions, **maxDistance** ( he can send his sons not farther than **maxDistance** ), **maxFee** ( he can pay at most **maxFee** ) and **minFee** ( he wants to pay at least **minFee** ; you know... Indian father, prestige issues ! ;- ) .

In each of his queries, he wants you to consider only a set S of schools that satisfy the given 3 conditions and all you need to do is just count the maximum number of schools that can be in a 'Max Special S'.

## Notes

If school X has **distanceX** & **feeX** and school Y has **distanceY** & **feeY**, then Mr.Nagulu prefers school X over school Y if and only if ( **distanceX**  $\leq$  **distanceY** AND **feeX**  $\leq$  **feeY** ). You can assume that no two schools have both distance and fee equal. Also, a school can not be preferred over itself.

## Input

First line contains an integer **N** ( total number of schools  $1 \leq N \leq 100,000$  ). Each of the next **N** lines contains **D F** ( **D** is the distance and **F** is the fee of each school,  $1 \leq D, F \leq 1,000,000,000$  and no two schools have both distance and fee equal. ) Next line has an integer **Q** ( number of queries  $1 \leq Q \leq 100,000$  ). Each of the next **Q** lines describe a query, having **maxDistance** **minFee** **maxFee** ( as described in the problem,  $1 \leq \text{maxDistance}, \text{minFee}, \text{maxFee} \leq 1,000,000,000$  and **minFee**  $\leq$  **maxFee** ).

## Output

For each query, print the answer in a new line.

## Example

**Input:**

5

1 1

2 2

4 3

3 4

1 5

5

5 3 4

5 4 6

5 1 4

3 1 4

4 2 6

**Output:**

2

2

1

1

2

**Explanation :**

Listing the schools in 'Max Special S' for each of the queries in the given sample case. Schools are listed in (distance, fee) pairs.

5 3 4 --> (4,3) (3,4) ;

5 4 6 --> (3,4) (1,5) ;

5 1 4 --> (1,1) ; This school is preferred over all other schools ;

3 1 4 --> (1,1) ;

4 2 6 --> (2,2) (1,5) ; (2,2) is preferred over (3,4) and (4,3)

**Warning :** Large input / output. You may have to use efficient input / output methods if you are using languages with heavy input / output overload. Eg: Prefer using scanf/printf to cin/cout for C/C++

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted. Time limit on each test set is 3 sec

JUNE11	<a href="#">June Long Contest 2011</a>	01 Jun 2011 15:00:00	11 days	653
--------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [June Long Contest 2011](#) » Chef Teams

## Chef Teams Problem Code: CTEAMS

The executive chef is trying to bring some competitive spirit into his kitchen. He wants to split the chefs into two teams based on their age - he'll form the young and the old team. To make it fair, he will split them evenly or give the young team one person advantage when there is an odd number of chefs. Ages of all employees are unique. The executive chef also rated all chefs according to their cooking skills. Rating of a team is equal to the sum of ratings of its members. The chefs have developed a habit of coming to work late. The executive chef wants to keep the teams as fair as possible at all times and is therefore forced to change the teams each time one of the chefs comes to work in the morning. He needs your help with this task.

### Input

The first line contains the number of chefs N. The following N lines describe the chefs in order as they come to work. Each chef is described by two integers, his or her age  $A_i$  and rating  $R_i$ .

### Output

Every time a new chef joins the kitchen, output the absolute difference between team ratings.

### Constraints

- $1 \leq N \leq 10^5$
- $1 \leq A_i \leq 10^9$
- $1 \leq R_i \leq 1000$

### Example

Input:

5

2 3

1 7

5 5

3 1

8 15

**Output:**

3

4

5

4

9

[Home](#) » [Compete](#) » [June Long Contest 2011](#) » Maximal Score Path

## Maximal Score Path Problem Code: RG\_01

### Problem Statement

Given a weighted and undirected graph  $G = (V, E)$ , let us define the score of an edge as its weight, and the score of a path as the minimum of the scores of its edges. For each pair of vertices  $(u, v)$ , let us define a best path as a path with the maximal score, that starts at  $u$  and ends at  $v$ . Your task is to find out the score of a best path over all pairs of distinct vertices  $(u, v)$  given the description of the graph  $G$ .

### Input

The first line contains  $V$ , the number of vertices, and  $E$ , the number of edges in the graph. The graph will be weighted, undirected, simple (no self loops and no parallel edges), and connected. Each of the next  $E$  lines contains three non-negative integers  $u$ ,  $v$ , and  $w$ , denoting that there is an edge  $(u, v)$  in the graph with a score of  $w$ .  $u$  and  $v$  are guaranteed to be distinct, and no edge will repeat in the input.

### Output

Output a total of  $V$  lines each containing  $V$  integers. The  $v$ th integer on the  $u$ th line should be 0 if  $u = v$ , or the score of a best path that starts at vertex  $u$  and ends at vertex  $v$ .

### Constraints

$2 \leq V \leq 1000$

$V - 1 \leq E \leq V(V - 1)/2$

$0 \leq u, v \leq V - 1$   
 $0 \leq w \leq 10^8$

## Example

**Input:**

3 3  
0 1 1  
1 2 2  
0 2 3

**Output:**

0 2 3  
2 0 2  
3 2 0

## Warning

Large Input/Output. Use faster Input/Output techniques.

[Home](#) » [Compete](#) » [June Long Contest 2011](#) » Restock

## Restock Problem Code: RESTOCK

The chef has to replenish supplies in the kitchen. He divided the kitchen floor into a rectangular grid of cell with  $N$  rows and  $M$  columns. All coordinates in this problem are 0-indexed and in the form (row, column). The supplies will be delivered to the cell  $(R,C)$  and have to be moved to the storage, which is located in the cell  $(0,0)$ . The chef will organize his employees so that they can pass individual items between them from the point of delivery to the storage. However, he has one additional request. The chef wants to see progress with every pass, which means that the Euclidean distance between the item and the storage has to decrease with every pass.

An employee assigned to the cell  $(Y_1, X_1)$  can pass an item to another employee in cell  $(Y_2, X_2)$  if  $|Y_1 - Y_2| \leq D$  and  $|X_1 - X_2| \leq D$ . Note that one person has to be assigned to cell  $(R, C)$  in any valid assignment. The chef assigned a wage to each cell to compensate for different working conditions. For example, standing next to an oven is not a very popular

spot among the employees. The chef will take the position at the storage himself, therefore the wage for the cell (0,0) will be 0 in all cases. Of course, the chef wants to pay as little as possible. The cost of some assignment of employees to different cells is equal to the sum of wages associated with their cells. The chef can always find some new workforce, so the number of workers is not a restriction. Help him find the cost of the cheapest assignment.

## Input

The first line contains a single integer  $T$ , the number of test cases. The first line of each testcase contains the number of rows  $N$  and the number of columns  $M$ . Second line contains integers  $D$ ,  $R$  and  $C$ . Following  $N$  lines with  $M$  integers describe the wages.  $j$ -th number in  $i$ -th line represents the wage  $w_{i-1,j-1}$  for the cell  $(i-1,j-1)$ .

## Output

Output a line with a single integer, the cost of the cheapest assignment of workers to cells such that they can pass the items from the point of delivery to the storage.

## Constraints

- $1 \leq T \leq 10$
- $1 \leq N, M \leq 500$
- $1 \leq D \leq 500$
- $0 \leq w_{i,j} \leq 10000$
- The sum of  $N \times M$  over all test cases in a single test file will not exceed 250000.

## Example

**Input:**

```
2
1 5
2 0 4
0 1 5 1 4
5 6
2 4 3
0 7 8 5 9 1
1 6 8 4 6 2
5 4 2 5 0 3
```

5 2 0 6 8 8

3 5 3 3 8 4

**Output:**

6

4

[Home](#) » [Compete](#) » [June Long Contest 2011](#) » Mushroom Cave

## Mushroom Cave Problem Code: CAVE

Chef is exploring a large cave looking for exotic mushrooms. <!-- the cave is looking for mushrooms??? --> It is very dark in the cave, so Chef uses torches to help him see. Torches do not last forever, so as Chef explores the cave he must find new torches to light. Specifically, once Chef picks up a torch, he may only travel K steps before it will burn out. If Chef doesn't find another torch within K steps, he will likely be eaten by a grue. Whenever Chef finds a torch, he immediately lights it, and drops any currently held torch to the ground, rendering it unusable. Chef wants to explore as much of the cave as possible before exiting. Chef begins at the northwest corner of the cave and must end at the southeast corner.

You will be given the layout of the cave, including the locations of all torches. Your task is to plot a route through the cave from the northwest corner to the southeast corner that visits as many distinct cells of the cave as possible. Chef must be able to follow your route without running out of torch light. Chef must also still have a burning torch when he finishes the route. Note that optimal routes are not required, and your submission will be scored relative to others' submissions. Any route that leads Chef from the northwest corner to the southeast corner will be accepted.

### Input

Input will begin with an integer T, the number of test cases (between 3 and 10). Each test case begins with 3 integers M, N, K ( $M, N \leq 100$ ,  $2 \leq K \leq 15$ ). M and N are the dimensions of the cave, K is the duration of each torch. M lines follow of N characters each, describing the cave. Rows proceed from north to south, and columns from west to east. A '.' character indicates an empty cell, a '#' character indicates an impassable cell, and a 't' character indicates a torch. The cell in the northwest corner will always be a 't', and the cell in the southeast corner will never be '#'.

### Output

For each test case, output a string of 'N', 'W', 'S', and 'E' characters (corresponding to the cardinal directions **North**, **West**, **South**, and **East**, respectively) giving a route from the northwest corner to southeast. It is guaranteed that at least one such route will exist.

## Scoring

Your score for each test case is the total number of cells visited by your route (visiting the same cell multiple times only counts once toward the total) divided by the total number of passable cells. Your score for each file is the average of your scores on the individual test cases. Your overall score is the average of your scores on the individual test files.

## Sample Input

```
2
4 5 3
ttttt
.##..
.t...
##t..
6 6 4
tt#...
...##t
...tt.
..t.t#
....t.
..t..t
```

## Sample Output

```
SSEESEE
SENSSESSENNWEEENSWSSESWE
```

The score for the first test case is  $8/16=0.5$ . The score for the second test case is  $23/32=0.71875$ . The overall score is thus  $(0.5+0.71875)/2=0.609375$ . Note that the solution to the first test case is unique (in particular, the solution "EEEESSS" is invalid because Chef would run out of torch light during his final step), but in the second test case there are many alternate solutions (including some that visit more than 23 distinct cells).

## Test Case Generation

M and N are chosen randomly and uniformly between 10 and 100, inclusive, and K is chosen randomly and uniformly between 2 and 15. Additionally, a real number D is chosen randomly and uniformly between .05 and .2, inclusive. Each cell is chosen as '#' with probability D, 't' with probability  $(1-D)/K$ , and '.' with probability  $(1-D)*(K-1)/K$ . If no valid path exists from the northwest to southeast corner, the process is restarted with the same values of M, N, K, and D.

[Home](#) » [Compete](#) » [June Long Contest 2011](#) » Attack of the Clones

## Attack of the Clones Problem Code: CLONES

A boolean function is a function of the form  $f: B_n \rightarrow B$ , where  $B = \{0, 1\}$  and  $n$  is a non-negative integer called the arity of the function. Some Boolean functions are projections:  $p_{n^k}(x_1, \dots, x_n) = x_k$ . And given an  $m$ -ary function  $f$ , and  $n$ -ary functions  $g_1, \dots, g_m$ , we can construct another  $n$ -ary function:  $h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_m(x_1, \dots, x_n))$ , called their composition. A set of functions closed under composition and containing all projections is called a clone. One trivial clone is a set of all boolean functions. Some of the special clones are:

- $Z$  is a set of 0-preserving functions:  $f(0, \dots, 0) = 0$ ;
- $P$  is a set of 1-preserving functions:  $f(1, \dots, 1) = 1$ ;
- $D$  is a set of self-dual functions:  $!f(x_1, \dots, x_n) = f(!x_1, \dots, !x_n)$ ;
- $A$  is a set of affine functions: the functions satisfying that if  $f(a_1, \dots, c, \dots, a_n) = f(a_1, \dots, d, \dots, a_n)$  then  $f(b_1, \dots, c, \dots, b_n) = f(b_1, \dots, d, \dots, b_n)$ , where  $c$  and  $d$  are at some position  $i$ . This should hold for every valid  $i$ ,  $a_1, \dots, a_n, b_1, \dots, b_n, c$  and  $d$ .

Now we are interested how many  $n$ -ary functions are there in some combinations of mentioned above sets. For example, for  $n = 2$ , there are exactly 8 functions in  $Z$ , 4 functions in the intersection of  $Z$  and  $P$ , 8 function in the complement of  $A$  and so on.

## Input

The first line of the input file contains  $n$  - the arity of the boolean functions we are looking at. The second line contains the  $q$  - number of queries. Each of the next  $q$  lines will describe a query. The query is a set expression. The expression will contain the following characters: 'Z', 'P', 'D', 'A' denoting the sets, described above; 'v' - which is set union; '^' - which is set intersection; '!' which is complement; '\\' which is set difference; and also '(' and ')' to define operations priority. Operations in brackets have higher priority. Otherwise the '!' operation has the higher priority and 'v', '^' and '\\' are of the same priority. It is guaranteed that the expression will be correct. See samples for some examples of set expressions.

## Constraints

$1 \leq n \leq 100$

$1 \leq q \leq 100$

The length of each expression won't exceed 100 characters.

## Output

For each query in the input print how many  $n$ -ary function are in the set described by the according set expression modulo 1000003.

## Example

**Input:**

2

6

$Z$

$Z^P$

$!A$

$! (AvP) ^ D$

$AvZvP \setminus A$

$!A ^ (Z \setminus (Dv!P))$

**Output:**

8

4

8

0

6

2

# Minesweeper Reversed

Problem Code: MINESREV

You are probably familiar with the classic computer game called Minesweeper. In case you're not, here's a short description. The game is played on a rectangular grid with  $R$  rows and  $C$  columns. Some cells contain mines and the others are empty. Initially, all cells are closed - their content is hidden. A player has to open all empty cells without opening a cell which contains a mine. You can open a cell by clicking on it. Opening an empty cells displays a number of mines in the neighbouring 8 cells. In case there aren't any mines in the neighbourhood, all neighbouring cells are automatically opened. This way you can open large mine-free areas with a single click.

As the name of this problem suggests, you will be dealing with a reversed process of minesweeping. A player starts with a grid where all cells are open and are either empty or contain a mine. The goal of this reversed game is to hide the mines by closing all cells as fast as possible. There are two ways to close a cell. You can either click on it or the cell can automatically close as a result of clicking on some other cell. Clicking on a cell  $C_1$  closes all cells  $C_2$  which could open simultaneously with  $C_1$  in a normal game of Minesweeper (as a result of a single click somewhere on the grid). More formally, let  $S_{x,y}$  be a set of all cells which are opened when you click on the cell  $(x,y)$  in a normal game of Minesweeper. Clicking on a cell  $C_1$  will also close all cells  $C_2$  such that there exists a set  $S_{x,y}$  which contains both  $C_1$  and  $C_2$ . You can safely click on a mine cell to close it. The goal is to close all cells with as few clicks as possible.

## Input

The first line contains a single integer  $T$ , the number of test cases. The first line of each testcase contains number of rows  $R$  and number of columns  $C$ . Following lines describe the playing grid. '.' represents an empty cell and '\*' indicates a mine.

## Output

Output a single line containing the minumum number of clicks necessary to close all cells.

## Constraints

- $1 \leq T \leq 50$
- $1 \leq R, C \leq 50$

## Example

Input:

2

3 8

\*\*...\*..

....\*..

...\*\*\*..

2 1

\*

\*

**Output:**

9

2

COOK11

[June Cook-off](#)

19 Jun 2011  
21:30:00

2 hours 30 minutes

413

[Home](#) » [Compete](#) » [June Cook-off](#) » Correctness of Knight Move

## Correctness of Knight Move Problem Code: KNIGHTMV

Chef develops his own computer program for playing chess. He is at the very beginning. At first he needs to write the module that will receive moves written by the players and analyze it. The module will receive a string and it should report at first whether this string represents the correct pair of cells on the chess board (we call such strings correct) and then report whether it represents the correct move depending on the situation on the chess board. Chef always has troubles with analyzing [knight moves](#). So at first he needs a test program that can say whether a given string is correct and then whether it represents a correct knight move (irregardless of the situation on the chess board). The cell on the chessboard is represented as a string of two characters: first character is a lowercase Latin letter from **a** to **h** and the second character is a digit from **1** to **8**. The string represents the correct pair of cells on the chess board if it composed of 5 characters where first two characters represent the cell where chess figure was, 3rd character is the dash **-** and the last two characters represent the destination cell.

### Input

The first line contains a single integer **T <= 50000**, the number of test cases. **T** test cases follow. The only line of each test case contains a non-empty string composed the characters with ASCII-codes from 32 to 126. The length of the string is not greater than **10**.

### Output

For each test case, output a single line containing the word "**Error**" if the corresponding string does not represent the correct pair of cells on the chess board. Otherwise output "**Yes**" if this pair of cells represents the correct knight move and "**No**" otherwise.

## Example

**Input:**

4

a1-b3

d2-h8

a3 c4

ErrorError

**Output:**

Yes

No

Error

Error

[Home](#) » [Compete](#) » [June Cook-off](#) » Super-plane

## Super-plane Problem Code: SUPERPLN

Dunno was invited by Roly-Poly to his birthday party. But he don't know how to get to Roly-Poly. So Dunno turned for advice to Doono. Doono reminded him that their country of shorties have a network of super-planes airlines, which can move not only in space but also in time. Thanks to a super engine, developed by Doono and constructed by Bendum and Twistum, such aircraft, flying from one city could get to the other even before they departed.

Dunno rejoiced and rushed into the super-airport. Not thinking through his route, Dunno began to fly at super-planes aimlessly, i.e. boarded the next flight out of that city, where he was. Dunno flies until he reaches the city where Roly-Poly lives not later than the start time of the birthday party, or until it reaches a certain city, where no longer be able to fly away. The closest flight is a flight in the same city, for which the departure time is not less than

the arrival time of Dunno in this city, and the difference between these time points is minimal.

Write a program, that determines whether Dunno ever gets to Roly-Poly given the schedule of the super-plane flights. If he gets the party in time then find also the number of flights he uses.

## Input

The first line contains a single positive integer **T**  $\leq 10$ , the number of test cases. **T** test cases follow. The first line of each test case contains the number of available flights **N** (**0**  $\leq N \leq 10000$ ) . Each of the following **N** lines contains 4 space-separated integers **C<sub>1</sub>**, **T<sub>1</sub>**, **C<sub>2</sub>**, **T<sub>2</sub>**, where **C<sub>1</sub>** is the departure city, **T<sub>1</sub>** is departure time, **C<sub>2</sub>** is the destination city, **T<sub>2</sub>** is arrival time. It is guaranteed that there are no flights with the same departure city and time. The last line of each test case contains the city where Dunno lives, the time when he came to the super-airport, the city where Roly-Poly lives and the time when the birthday party starts. All city numbers are positive integers not greater than **10000** and all time points are not greater than **100000** in absolute value.

## Output

For each test case, output a single line containing the word "**Yes**" followed by space and the number of flights that the Dunno uses if he gets the Roly-Poly birthday party in time and "**No**" otherwise.

## Example

Input:

3

3

1 3 4 3

4 4 3 5

3 10 2 -1

1 2 2 0

1

1 2 3 0

1 2 2 2

2

2 0 2 0

2 1 4 0

2 0 4 0

**Output:**

Yes 3

No

No

[Home](#) » [Compete](#) » [June Cook-off](#) » Longest Arithmetic Progressions

## Longest Arithmetic Progressions Problem Code: ARITHPR

You are given positive integers  $L, R, k$  such that  $k \leq R - L$ . Consider all strictly increasing arithmetic progressions with difference not less than  $k$  composed of numbers from the set  $\{L, L+1, \dots, R\}$ . At first you need to find the length of the longest such progression. Easy as pie, right? Now consider all such longest progressions and write them down in lexicographical order. You need to find first two terms of the  $n^{\text{th}}$  such progression. (Note that because of the condition  $k \leq R - L$  the length of the longest progression is at least two.)

**Remark.** We say that sequence  $a = (a[0], a[1], \dots, a[n])$  is lexicographically smaller than  $b = (b[0], b[1], \dots, b[n])$  if there exists some  $i$  such that  $0 \leq i \leq n$ ,  $a[j] = b[j]$  for  $0 \leq j < i$  and  $a[i] < b[i]$ .

### Input

The first line contains a single integer  $T \leq 10000$ , the number of test cases.  $T$  test cases follow. The only line of each test case contains four positive integers  $L, R, k$  and  $n$  where  $R \leq 10^9$ ,  $k \leq R - L$  and  $n \leq 10^{18}$ .

### Output

For each test case, output a single line containing the length of the longest strictly increasing arithmetic progressions with difference not less than  $k$  composed of numbers from the set  $\{L, L+1, \dots, R\}$  followed by the first two terms of the  $n^{\text{th}}$  such progression in lexicographical order. If the total number of such progressions is less than  $n$  then simply output two zeros instead of the first two terms of the progression.

## Example

### Input:

3

2 6 2 1

2 6 2 2

1 3 2 8 12

### Output:

3 2 4

3 0 0

4 5 14

[Home](#) » [Compete](#) » [June Cook-off](#) » Sines Sum Queries

## Sines Sum Queries Problem Code: SINSUMQ

You are given a sequence of integers  $A_0, A_1, \dots, A_{N-1}$ . Initially  $A_i = i$  for all  $i$ . You need to perform some strange queries with it. Each query has the form " $L \ R \ D$ " where  $0 \leq L \leq R < N$  and  $D$  is an integer. If  $D=0$  then you need to find the sum of sines of the numbers  $A_L, A_{L+1}, \dots, A_R$  that is  $\sin A_L + \dots + \sin A_R$ . Otherwise you need to add  $D$  to the numbers  $A_L, A_{L+1}, \dots, A_R$ .

### Input

The first line contains two positive integers  $N$  and  $Q$ . Here,  $N \leq 10^9$  is the length of the initial sequence and  $Q \leq 100000$  is the number of queries you need to perform with it. The next  $Q$  lines describe queries, one per line. The  $i$ 'th such line contains three integers  $L$ ,  $R$  and  $D$ . Here  $0 \leq L \leq R < N$  and  $-10000 \leq D \leq 10000$ .

### Output

The output consists of answers for all queries where  $D=0$ . For each such query you must produce a line with the corresponding sines sum. Answers within an absolute error of  $10^{-6}$  will be accepted.

## Example

**Input:**

```
5 6
0 4 0
0 2 1
3 4 2
0 4 0
2 3 -1
1 4 0
```

**Output:**

```
1.1350859
0.65354865
0.782376860
```

[Home](#) » [Compete](#) » [June Cook-off](#) » D-Power Permutations

## D-Power Permutations Problem Code: DPOWPERM

Let  $N$  be a positive integer and  $S = \{1, 2, 3, \dots, N\}$ . For a given positive integer  $d$  the function  $f : S \rightarrow S$  is called  $d$ -power permutation if there exists a bijection  $g : S \rightarrow S$  such that  $g(g(\dots g(x) \dots)) = f(x)$  for each  $x$  from  $S$ , where  $g$  is repeated exactly  $d$  times.

You are given some bijection  $f : S \rightarrow S$  and a positive integer  $D$ . You need to find the number of those  $d \leq D$  such that  $f$  is  $d$ -power permutation.

**Input**

The first line contains a single positive integer  $T \leq 10$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains two positive integers  $N$  and  $D$ , where  $N \leq 700$  and  $D \leq 10^{18}$ . The second line contains  $N$  space-separated integers.  $i^{\text{th}}$  number in the second line is the value  $f(i)$ . It is guaranteed that  $f$  is bijection from  $S$  onto  $S$ .

**Output**

For each test case, output a single line containing the answer for the corresponding test case.

## Example

**Input:**

2

3 6

1 2 3

5 10

2 1 4 5 3

**Output:**

6

3

## Explanation

In the second case the appropriate values of **d** is 1, 5 and 7.

JULY11	<a href="#">July 2011 Long Contest</a>	01 Jul 2011 15:00:00	10 days	460
--------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » Most Popular Friend

## Most Popular Friend Problem Code: LOKBIL

Anna Hazare is a well known social activist in India.

On 5th April, 2011 he started "Lokpal Bill movement".

Chef is very excited about this movement. He is thinking of contributing to it. He gathers his cook-herd and starts thinking about how our community can contribute to this.

All of them are excited about this too, but no one could come up with any idea. Cooks were slightly disappointed with this and went to consult their friends.

One of the geekiest friend gave them the idea of spreading knowledge through Facebook. But we do not want to spam people's wall. So our cook came up with the idea of dividing Facebook users into small friend groups and then identify the most popular friend in each group and post on his / her wall. They started dividing users into groups of friends and identifying the most popular amongst them.

The notoriety of a friend is defined as the averaged distance from all the other friends in his / her group. This measure considers the friend himself, and the trivial distance of '0' that he / she has with himself / herself.

The most popular friend in a group is the friend whose notoriety is least among all the friends in the group.

Distance between X and Y is defined as follows:

Minimum number of profiles that X needs to visit for reaching Y's profile (Including Y's profile). X can open only those profiles which are in the friend list of the current opened profile. For Example:

- Suppose A is friend of B.
- B has two friends C and D.
- E is a friend of D.

Now, the distance between A and B is 1, A and C is 2, C and E is 3.

So, one of our smart cooks took the responsibility of identifying the most popular friend in each group and others will go to persuade them for posting. This cheeky fellow knows that he can release his burden by giving this task as a long contest problem.

Now, he is asking you to write a program to identify the most popular friend among all the friends in each group. Also, our smart cook wants to know the average distance of everyone from the most popular friend.

## **Input**

Friends in a group are labelled with numbers to hide their Facebook identity. The first line of input contains the number of groups in which users are divided. First line of each group contains the number of friends that belong to that group.  $i^{th}$  line of each group contains the space separated friend list of 'i'. You are assured that each friend's profile can be accessed by the profile of every other friend by following some sequence of profile visits.

## **Output**

Your output contains the most popular friend name label along with the average distance (space separated) from all other friends (including himself / herself) in six digits of precision. There might be multiple most popular friend, in that case output the friend labelled with least number.

## **Note:**

Each person in a group have atleast one friend and he/she cannot be in his/her own friend list.

Number of friends in a group cannot be more than 100.  
There are atmost 100 groups.

## Example

**Input:**

1

6

3

5

1 4

3 5 6

2 4 6

4 5

**Output:**

4 1.166667

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » Large Kitchen

## Large Kitchen Problem Code: KITCHEN

There is a large kitchen in one of the largest Chef's restaurants. Well, to be honest, that's not a real kitchen yet, since there's nothing in it. The first thing to do now is to place cookers, ovens, tables and other stuff. You are to help Chef in this issue.

The kitchen-to-be can be imagined as a rectangular grid with  $N$  rows and  $M$  columns consisting of  $N \times M$  equal cells which are currently empty. As unlikely as it may seem, each piece of the aforementioned stuff takes exactly one cell of the grid. Chef would like to use as many cells as possible for the stuff, but there's just one restriction: there should be no "closed" areas at the kitchen -- areas which can't be reached without moving anything, otherwise it would be too hard for chefs to pick up, for example, a fallen knife. In other words, there should exist **no** sequences of used cells  $X_1, X_2, \dots, X_K$ ,  $K > 2$ , such that for every  $i$  between 1 and  $K-1$ , inclusive, cells  $X_i$  and  $X_{i+1}$  are neighbouring, cells  $X_1$  and  $X_K$  are neighbouring too, and no cell is repeated twice. Two cells are called neighbouring if they share a common side.

Yet there is another strange restriction. The set of used cells must be *connected*, that is, for every pair of used cells there should exist a sequence of used cells  $X_1, X_2, \dots, X_k$  such that  $X_1$  is one of these cells,  $X_k$  is another one, and for every  $i$  between 1 and  $K-1$ , inclusive, cells  $X_i$  and  $X_{i+1}$  are neighbouring.

Your task is to use as many cells as possible under these restrictions. Note that this is a challenge problem: you don't have to find the optimal solution, it's enough to find any of them (but the better is your solution, the more points you receive).

## Input

Input will begin with an integer  $T$ , the number of test cases (no more than 30). Each test case consists of 2 integers  $N$  and  $M$  ( $N, M \leq 100$ ), which denote the dimensions of the kitchen.

## Output

For each test case output exactly  $N$  lines containing exactly  $M$  characters each, describing the final state of the kitchen. A '#' character should represent a cell which can be used for placing something, and a '.' character should represent a cell which should remain free. You may separate the answers for consecutive test cases with empty lines.

## Scoring

Your score for each test case is one hundred times the total number of used cells in your output divided by the total number of cells (in fact, this number indicates the percentage of used cells in your solution). Your score for each file is the average of your scores on the individual test cases. Your overall score is the average of your scores on the individual test files.

## Example

Input:

2

4 4

5 8

Output:

```

.###

##.#

#.##

##.#

#.#.#.#

####.####

.#.#.#.#

##.##.. .

.###.###
```

The score for the first test case is  $100*12/16 = 75$ . The score for the second test case is  $100*26/40 = 65$ . The overall score is thus  $(75+65)/2 = 70$ . Note that the solution to the first test case is optimal (there is no correct output with a larger number of used cells), but in the second test case there exist several solutions with more than 26 used cells.

## Test Case Generation

Every official input file contains exactly 30 test cases. In every test case M and N are chosen randomly and uniformly between 10 and 100, inclusive.

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » A-E Hash Function

## A-E Hash Function Problem Code: AEHASH

Chef Ash and Chef Elsh invented a new hash function! Their hash function will map a binary string consisting of characters 'A' and 'E' into an integer called the hash value of the string.

The pseudocode of the hash function is as below. hash(S) is the hash value of a binary string S.  $|S|$  denotes the length of S.

```

function hash(S) :

    result = number of characters 'A' in S
```

```

if |S| > 1:

    (S1, S2) = split(S)

    result = result + max(hash(S1), hash(S2))

end if

return result

end function

```

The function split in the above pseudocode takes a binary string S as the parameter and returns a pair of binary strings (S1, S2) such that:

- $|S1| \leq |S2|$ .
  - The difference of  $|S1|$  and  $|S2|$  is at most 1.
  - The concatenation of S1 and S2 (in that order) is S.
- For example, split("AAEAE") returns ("AA", "AEE"), whereas split("AEAEAE") returns ("AEA", "EAE").

You doubt that this hash function have good distribution of different hash values. So, you wonder how many different binary strings consisting of A 'A' characters and E 'E' characters that have hash value of V.

## Input

The first line contains a single integer T, the number of test cases. T test cases follow. Each testcase consists of a single line consisting of three integers A, E, and V.

## Output

For each test case, output a single line consisting the number of different binary strings satisfying the rule, modulo 1000000007.

## Constraints

- $1 \leq T \leq 1000$
- $0 \leq A \leq 50$
- $0 \leq E \leq 50$
- $0 \leq V \leq 1000$

## Example

Input:

0 0 0

1 0 1

3 2 6

4 2 8

**Output:**

1

1

3

4

## Explanation

For the last test case, the solutions are:

- AAEAAE
- AEAAAAE
- AAEAEA
- AEAAEA

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » Wireless Network

## Wireless Network Problem Code: NETWORK

A wireless network has been built consisting of many wireless access points for computers. Each access point can support a certain number of computers, and the network will perform better when computers are close to the access points supporting them. Specifically, the greatest distance between any computer and its access point determines the strength of the network. A network has recently been designed that will automatically reconfigure itself every time a computer is added so that the greatest distance will be as small as possible. Your task is to compute, as each computer is added to the network, this distance.

Consider the following example:

There are 2 access points, at (5,5) and (6,9), each of which can support 2 computers. The first computer to be added to the wireless network is at (3,5). This computer connects to the access point at (5,5), which is only a distance of 2 away. The next computer to be added is at (5,2). This computer also connects to the access point at (5,5), and the greatest distance is now 3. A third computer is added at (4,2). This computer cannot

connect to the (5,5) access point because it has already reached its limit. But connecting to (6,9) is not a good choice either because its so far away. Instead, the network reassigns the (3,5) computer to the (6,9) access point so that the new computer can connect to the (5,5) access point. The greatest distance is now from the (3,5) computer to the (6,9) access point, which is 5. (Note: in the picture below, the top-left corner is (0,0)).



## Input

Input begins with an integer  $T$ , the number of test cases to follow. Each test case begins with a line containing 2 integers,  $S$  (the number of access points) and  $C$  (the number of computers).  $S+C$  lines follow. The first  $S$  lines contain 3 integers each, giving the coordinates of an access point, followed by the number of connections it can support. The next  $C$  lines contain 2 integers each, giving the coordinates of a computer. There is a blank line before each test case.

## Output

For each test case, output C lines, where the  $i^{\text{th}}$  line contains the greatest distance in the network after the first  $i$  computers have been added to the network, rounded to 3 decimal places. Output a blank line after each test case.

**Sample Input**

3

2 3

5 5 2

6 9 2

3 5

5 2

4 2

2 4

3 6 2

10 9 2

2 5

3 8

4 3

6 4

2 1

10 5 1

6 8 1

3 3

**Sample Output**

2.000

3.000

5.000

1.414

2.000

7.071

7.071

5.831

## Constraints

$T$  is between 1 and 15, inclusive.

$S$  is between 1 and 30, inclusive.

Each access point supports between 1 and 30 connections, inclusive.

$C$  is between 1 and the total number of supported connections, inclusive.

All coordinates comprise integers between 0 and 10000, inclusive.

All coordinates are distinct.

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » Billboards

## Billboards Problem Code: BB

There is a long long highway leading to Chef's new restaurant. But the problem is that almost no one stops in it, hence it's even unprofitable to hold it now. The only idea Chef has is that the restaurant has not been very well advertised, so it has no clients just because nobody knows about it! In order to change the situation Chef has decided to use some billboards along the highway for advertising.

There are exactly  $N$  billboards along the highway. Chef may place advertisements at any subset of billboards. To prevent casual drivers from forgetting about how good the restaurant is, there should be at least  $K$  restaurant's advertisement among every  $M$  consecutive billboards. But still each billboard's time costs some money, and Chef would like to pay the least amount possible (after all, he has already spent a lot of money for the restaurant at no profit). Now you are to help Chef count the number of different ways to place the minimal number of advertisements still meeting the condition above.

## Input

The first line of the input contains a single integer  $T$  -- the number of test cases (no more than 25). Each of the next  $T$  lines describes one test case and contains three integers  $N$ ,  $M$  and  $K$  ( $1 \leq K \leq M \leq 50$ ,  $M \leq N \leq 10^9$ ).

## Output

For each test case output a single line containing the requested number of ways modulo 1 000 000 007.

## Example

### Input:

3

3 2 1

4 2 1

6 3 2

### Output:

1

3

6

### Explanation:

In the second test case there are 3 ways to place the advertisements: using the 1st and the 3rd, the 2nd and the 3rd, or the 2nd and the 4th billboards.

[Home](#) » [Compete](#) » [July 2011 Long Contest](#) » Trial of Doom

## Trial of Doom Problem Code: YALOP

Johnny has reached the final trial on the Path of Doom. This is the hardest one. He entered a large room. The floor of the room is divided into square cells of the same size. There are  $n$  rows and  $m$  columns of cells. Each cell can be either red or blue. We can consider that the walls of the room are parallel south-north direction and west-east direction and that

Johnny has entered the room in the north-western corner. So now he is standing in the north-western cell of the room. The exit is in the south-eastern corner. Johnny can make steps in each of the eight directions moving to adjacent cells. He has to reach the exit. But when he goes out of the room all the cells in the room should be blue, otherwise he fails the trial (If Johnny goes out he can't step back again). The colors of the cells change after each step Johnny makes. When he steps on the next cell this cell and the cells to its north, south, east and west change color: from blue to red and from red to blue. Johnny can't ever step on two cells at once or he fails the trial. Also just jumping on the cell he is standing on now will have no effect on the color of cells. Currently some cells in the room may be red. Johnny made a look over the room and he wonders if it's even possible to pass the trial at all or the trial master is playing a trick on him. Help him find this out.

## Input

The first line of input is the number of test cases. Then each of the test cases follows. The test case starts with two numbers  $n$  and  $m$  - the sizes of the room. The next line contains number  $k$  - the amount of red cells in the room. Next  $k$  lines consist of two numbers each  $x$ ,  $y$  - the row number and column number of the following red cell. The rows are numbered from 1 to  $n$  from north to south and columns are numbered from 1 to  $m$  from west to east. So Johnny starts at cell (1, 1) and have to reach cell ( $n$ ,  $m$ ). The coordinates of all of the red cells will be different. The rest of the cells are blue.

## Constraints

$1 \leq t \leq 50$   
 $1 \leq n, m \leq 10^9$   
 $\min(n, m) < 40$   
 $0 \leq k \leq \min(m \cdot n, 10000)$   
 $1 \leq x \leq n$   
 $1 \leq y \leq m$

## Output

For each test case print "YES" (quotes for clarity) if Johnny can reach the south-eastern cell and go out of the room with all cells being blue and "NO" (quotes for clarity) otherwise.

## Example

Input:

3

2 2

0

2 2

4

1 1

1 2

2 1

2 2

4 4

1

1 1

**Output:**

YES

YES

NO

**Explanation**

In the first test case Johnny can pass the trial making the following steps: South-east, North, West, East, West, South-east, out. In the second test case: East, West, South, East, out. It's impossible to pass the trial in the third test case.

COOK12

[July Cook-off  
2011](#)24 Jul 2011  
21:30:00

2 hours 30 minutes

341

[Home](#) » [Compete](#) » [July Cook-off 2011](#) » Garden Squares

**Garden Squares** Problem Code: **GARDENSQ**

Chef has just finished the construction of his new garden. He has sown the garden with patches of the most beautiful carpet grass he could find. He has filled it with patches of different color and now he wants to evaluate how elegant his garden is.

Chef's garden looks like a rectangular grid of cells with  $N$  rows and  $M$  columns. So there are  $N \times M$  cells in total. In each cell Chef planted grass of some color.

The elegance of the garden is defined by the number of squares, composed of at least four garden cells, with edges parallel to the sides of the garden, that have four corner cells of the same color.

Given the description of Chef's garden, calculate how many such squares exist.

#### Input format

The first line contains the number  $T$ , the number of test cases. In the following lines,  $T$  test cases follow (without any newlines between them.) The first line of each test case contains  $N$  and  $M$ , separated by a single space. Each of the next  $N$  lines contains  $M$  characters without any spaces between them, and without any leading or trailing spaces. Each character describes the color of the corresponding cell in the garden and belongs to the set of lowercase and uppercase letters of the English alphabet. One letter in lowercase and uppercase describes different colors.

#### Output format

For each test case, print the number of squares that conform to the definition in the problem statement.

#### Constraints

$$\begin{aligned}1 \leq T \leq 50 \\1 \leq N, M \leq 50\end{aligned}$$

#### Sample input

3

2 2

aa

aA

3 3

aba

bab

aba

4 4

aabb

aabb

bbaa

bbaa

Sample output

0

1

4

Explanation

In the first case the only available square does not conform to the definition in the problem statement because 'a' and 'A' describes different colors.

In the second case, you can select the 4 a's at the corners of the garden.

In the third case, you can only make four squares, from the four 2x2 segments that are of the same color.

[Home](#) » [Compete](#) » [July Cook-off 2011](#) » Misinterpretation

## Misinterpretation Problem Code: MISINTER

Chef's brother likes to put words in Chef's mouth. Chef hates it about him of course. He has decided to become extremely careful about what he speaks. Fortunately, he knows how his brother transforms the words, that Chef uses. He has decided to use only those words which remain the same even after his brother's transformation!

If Chef speaks an N letter word, his brother moves all the letters which are at even positions (assuming, numbering of positions starts from 1), to the beginning of the word; and the rest of the letters follow as they appeared in the word. Eg. abdef becomes beadf; cdcd becomes ddcc.

Chef wants to know how many words can he use, provided that each word is composed of exactly N lowercase letters of the English alphabet. They use an ancient language in Byteland, which allows all possible words within the above definition!

Input format

The first line contains the number  $T$ , the number of test cases. In the following lines,  $T$  test cases follow. Every test case is a single line, that contains a single positive integer,  $N$ , which is the length of the words that Chef wants to use.

#### Output format

For each test case, print the number of words of length  $N$  that Chef can use; that is, number of words, which remain the same after brother's transformation. Since the result can be quite large, output the result modulo 1000000007.

#### Constraints

$1 \leq T \leq 100$   
 $1 \leq N \leq 100000$

#### Sample input

```
3
1
14
45
```

#### Sample output

```
26
456976
827063120
```

[Home](#) » [Compete](#) » [July Cook-off 2011](#) » Pleasing Chief

## Pleasing Chief Problem Code: CHIEFETT

Chef's girlfriend, Chief, is displeased with him. This is not a good sign for their relationship, and as always, he has turned to you for help. He looks forward to take his girlfriend to shopping; offering her to buy ' $K$ ' gifts. The shop that he is taking her to has ' $N$ ' unique items, that is, there is single quantity of each item (its a very special shop, you see!).

Chef has with him,  $K$  discount coupons that he wishes to use. The shop's policy only allows Chef to use one discount coupon on any one purchased item. That is why Chef

is letting Chief buy  $K$  gifts; he intends to use each coupon with him.

Chef of course cannot stop Chief from buying whatever she likes. Chief would select any  $K$  gifts, from among the  $N$  items that the shop has. She makes selection of any  $K$  out of  $N$  items, uniformly. Suppose,

there are 5 items and Chef lets Chief buy 3, then she selects any of the 3 out of 5 items with the probability  $1/10$ .

After Chief selects the items she wishes to buy, Chef will have to pay the bill. Smart as he is, he would apply the  $K$  discount coupons on the  $K$  items (one on each) such that the discount he receives is as large as possible.

Chef wants to prepare himself for this day. He knows beforehand the price of each item in the shop. Can you tell him what expected discount can he expect? You have to tell him the expected discount in the amount of money he can expect to save.

## Input format

The first line contains the number  $T$ , the number of test cases. In the following lines,  $T$  test cases follow (without any newlines between them.) Each case consists of only 3 lines. The first line of each test case contains  $N$  and  $K$ , separated by a single space. The second line contains  $N$  positive integers, the prices of the  $N$  items respectively, separated by a single space. The third line contains  $K$  positive integers (between 1 and 100, inclusive) separated by a single space. They represent the percentage of discount offered by the coupons, that Chef has, respectively.

## Output format

For each test case, print the expected amount of money Chef should expect to save. Output the result rounded to 3 digits after the decimal.

## Constraints

$1 \leq T \leq 50$   
 $1 \leq N \leq 1000$   
 $1 \leq K \leq N$   
 $1 \leq \text{prices} \leq 10000$

## Sample input

2

3 2

100 200 300

10 20

4 3

100 200 300 400

10 20 30

## Sample output

66.667

175.000

## Explanation

In the second case, Chef can make the following selections:

- (100, 200, 300), with probability 1/4. Chef can apply the coupons to at best save 140.
- (100, 200, 400), with probability 1/4. Chef can apply the coupons to at best save 170.
- (100, 300, 400), with probability 1/4. Chef can apply the coupons to at best save 190.
- (200, 300, 400), with probability 1/4. Chef can apply the coupons to at best save 200.

Thus Chef can save an expected  $(140 + 170 + 190 + 200) / 4 = 175.000$

[Home](#) » [Compete](#) » [July Cook-off 2011](#) » Mean Mean Medians

## Mean Mean Medians Problem Code: MEANMEDI

"Medians can't be very mean!", retorted Chef's brother. One would wish Chef's ego wouldn't come in the way, but Chef has taken up the challenge to prove otherwise. He asks for your help. Given N numbers, select K out of them, such that, the absolute difference between the mean and the median of the selected numbers, is as low as possible.

Mean of K numbers is defined as the sum of the numbers divided by K.

Median of K numbers is defined as the number that appears at the order index,  $\text{floor}((K+1)/2)$ ; that is, the  $I^{\text{th}}$  element in the sorted order of the K numbers (where numbering starts from 1), where  $I = \text{floor}((K+1)/2)$ . Note that, if K is even, the median would be the smaller value among the two values that lie in the center.

#### Input format

The first line contains the number T, the number of test cases. In the following lines, T test cases follow (without any newlines between them.) Each case consists of only 2 lines. The first line of each test case contains N and K, separated by a single space. The second line contains N positive integers, separated by a single space.

#### Output format

For each test case, print the minimum absolute difference between the mean and the median that you can get, by selecting any K numbers, from the N numbers. Output the result rounded to 3 digits of precision after the decimal.

#### Constraints

$1 \leq T \leq 20$   
 $1 \leq N \leq 60$   
 $1 \leq K \leq N$   
 $1 \leq \text{numbers} \leq 1200$

#### Sample input

```
2
8 2
4 9 1 3 5 9 4 10
5 4
10 7 4 5 9
```

#### Sample output

```
0.000
0.500
```

## Explanation

In the first case, you can select [4, 4].

In the second case, you can select [10, 7, 4, 9]. The mean would be 7.500, where as median would be 7.

[Home](#) » [Compete](#) » [July Cook-off 2011](#) » Chefs Game

## Chefs Game Problem Code: CHEF\_GAM

Chef loves that his cooks involve themselves in group activities. This month, Chef has involved his cooks in a rather very curious game. It involves all his cooks, standing in a circle. Each cook is given one integer to keep, which may be positive or negative (or zero). The sum of all the integers, given to the cooks, is positive.

Suppose there are  $N$  ( $N \geq 3$ ) cooks. Cooks are labeled from 1 to  $N$ .  $k$ 's neighbors are  $(k-1)$  and  $(k+1)$ , as they are standing in a circle. Note that if  $k = 1$ ,  $k-1$  wraps around to  $N$ ; and if  $k = N$ ,  $k+1$  wraps around to 1. Let  $A[i]$  represent the integer that is kept by the cook  $i$ . The game proceeds in turns. In each turn, any one cook may opt to play. Now, suppose the cook  $k$  wants to play in this turn, the following changes happen, in the given order:

- $A[k-1] = A[k-1] + A[k]$
- $A[k+1] = A[k+1] + A[k]$
- $A[k] = -A[k]$

The game would end as soon as all the numbers kept by the cooks (each  $A[i]$ ) is non-negative.

Cooks are very impatient people! They want to end the game as quickly as possible! Help them determine a sequence in which they must play, such that the number of turns in which the game finishes, is as low as possible. For the purpose of this problem, you are to find the minimum number of turns in which the game can end!

## Input format

The first line contains the number  $T$ , the number of test cases. In the following lines,  $T$  test cases follow (without any newlines between them.) Each test case consists of exactly two lines.

The first line contains the only positive integer  $N$ , the number of cooks playing the game. The second line contains  $N$  space separated integers, the initial numbers kept by the cooks.

## Output format

For each test case, print the minimum number of turns in which the game can end.

## Constraints

$1 \leq T \leq 10$

$3 \leq N \leq 50000$

$-200000 \leq (\text{initial numbers kept by the cooks}) \leq 200000$

## Sample input

2

3

1 -1 2

3

4 -2 -1

## Sample output

1

6

## Explanation

In the first case, cook 2 can take the first turn which converts the sequence into  $(0, 1, 1)$  and end the game.

In the second case, the following sequence of moves are optimal.

- 0.  $(4, -2, -1)$  -- initial
- 1.  $(3, -3, 1)$  -- after cook 3 moves
- 2.  $(0, 3, -2)$  -- after cook 2 moves

- 3.  $(-2, 1, 2)$  -- after cook 3 moves
- 4.  $(2, -1, 0)$  -- after cook 1 moves
- 5.  $(1, 1, -1)$  -- after cook 2 moves
- 6.  $(0, 0, 1)$  -- after cook 3 moves

AUG11

[August Long  
Contest 2011](#)01 Aug 2011  
15:00:00

11 days

629

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » Infinite Grid Game

## Infinite Grid Game Problem Code: IGAME

After a long period of relaxation Alice and Bob decided to play a game. This time of course not a number game. The rules of the game are as follows:

There is a vehicle situated at the point  $(m, n)$  of a rectangular grid. Only one corner of the rectangular grid is defined, i.e. the left-top point  $(0, 0)$ , which is also known as the origin. The grid extends infinitely towards east and infinitely towards south. Alice and Bob are both sitting in a vehicle, initially situated at the point  $(m, n)$ . The game they are playing ends as soon as one of them reaches  $(p, q)$ .

Now, Alice and Bob have to drive the vehicle in their respective turn. In their own turn they can move the vehicle, from  $(x, y)$  to  $(x', y)$  or  $(x, y')$ ; where  $p \leq x' < x$  and  $q \leq y' < y$ . They can also move the vehicle to the point  $(x-a, y-a)$ , where  $0 < a \leq \min(x-p, y-q)$ . Also,  $0 \leq p < m$  and  $0 \leq q < n$ . The winner is the one who drives the vehicle to  $(p, q)$ .

Cunning Alice uses a biased coin for tossing purpose and always plays first. It is assumed that both Alice and Bob play optimally in their respective turns.

### Input

The first line contains a single integer  $T$  denoting the number of test cases.  $T$  test cases follow. Each test case consists of a single line consisting of four space separated integers  $m, n, p, q$  respectively.

### Output

For each test case print a string - either "Alice" or "Bob" (without the quotes), denoting the winner.

### Constraints

$1 \leq T \leq 1000$   
 $1 \leq m, n \leq 1000$   
 $0 \leq p < m$   
 $0 \leq q < n$

### Sample Input

2

1 1 0 0

2 4 1 2

## Sample Output

Alice

Bob

## Explanation

In the second test case, initially the vehicle is at co-ordinate (2, 4). Alice now has four possible moves. Alice can move the vehicle to position (1, 4), (1, 3), (2, 3) and (2, 2). For all moves that Alice can do, Bob can move the vehicle to the position (1, 2), winning the game.

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » Complex Spanning Tree

## Complex Spanning Tree Problem Code: COMPLEXT

Our Chef was in intellectual mode today. He came across the maximum spanning tree problem in a grid; which he was able to solve in a jiffy for integral values. But, the never-say-die spirit of our chef pushed him to something more complex.

"What could be more complex than replacing the integer edges by complex edges", he thought. Help our chef to prove his intellectualism, as much as, he has proven his chef-skills.

You are given a grid of  $n \times n$  vertices named as  $a_{11}, a_{12}, \dots, a_{1n}, a_{21}, a_{22}, \dots, a_{2n}, a_{31}, a_{32}, \dots, a_{nn}$ . There are bidirectional edges between adjacent vertices. So a vertex on the top row (say  $a_{14}$ ) will have an edge to its left element ( $a_{13}$ ), to its right element ( $a_{15}$ ) and to its bottom element ( $a_{24}$ ). Any vertex cannot have more than 4 neighbors. Vertices on the edges will have 3 neighbors. Vertices on the corners of the grid will have 2 neighbors. Every edge is given as a complex number " $x y$ " (without the quotes). Now, you are required to find a spanning tree in this grid, such that, **the modulus of the complex sum of the edges in the spanning tree, is maximized.**

The complex sum is defined for  $(x_1, y_1)$  and  $(x_2, y_2)$  as  $(x_1+x_2, y_1+y_2)$ . The modulus of a complex number  $(x, y)$  is  $\sqrt{x^2 + y^2}$ . A spanning tree is defined as a set of edges in the grid such that every vertex has exactly one unique path to any other vertex.

## Input

The first line of input contains the number "n" (without the quotes); followed by  $2n$  lines of  $2n-2$  integers each. Each line will consist of  $2n-2$  integers denoting  $n-1$  complex numbers. Each consecutive pair of integers represents a complex number denoting the edge between neighboring vertices. For example, in the first row, 1st and 2nd integers form the complex number denoting the edge between  $a_{11}$  and  $a_{12}$ . The 3rd and 4th numbers form the complex number denoting the edge between  $a_{12}$  and  $a_{13}$  and likewise the  $(2n-3)$ th and  $(2n-2)$ th integers from the complex number denoting the edge between  $a_{1(n-1)}$  and  $a_{1n}$ . First  $n$  lines of input will denote the horizontal edges (like edges between  $a_{ij}$  and  $a_{i(j+1)}$ ). Next  $n$  lines of input will denote the vertical edges (like edges between  $a_{ij}$  and  $a_{(i+1)j}$ ). The 1st and 2nd integer in the  $(n+1)$ th line will form the complex number denoting the edge between  $a_{11}$  and  $a_{21}$  and likewise the  $(2n-3)$ th and the  $(2n-2)$ th integer in the same line will form the complex number denoting the edge between  $a_{(n-1)1}$  and  $a_{n1}$ .

## Output

Output  $(n^*n - 1)$  lines denoting  $(n^*n - 1)$  edges in the spanning tree you selected, one on each line - in any order. An edge should be represented as the pair of vertices it joins. So to represent the edge joining  $a_{ij}$  and  $a_{uv}$  you should output "i j u v" (each integer separated by a single space, without the quotes). An invalid output, such as joining non-neighboring vertices or selecting an edge more than once, will be reported as Wrong Answer. Outputting too few edges will also be reported as Wrong Answer. Only the first  $(n^*n - 1)$  lines of the output will be used so take care of not printing blank lines! Out of range values of  $i, j, u$  and  $v$  will also be reported as Wrong Answer. If the edges are not a valid spanning tree (by the definition above), the judge will report Wrong Answer.

## Constraints

$n = 20$

$-100 = \text{weight-}x, \text{ weight-}y = 100$

## Scoring

The Score for any file / input, would be the modulus of the complex sum of the weights of the edges that you select. The overall score for all files / inputs would be the average of the scores for each test file / input. Your objective is to **maximize** your score!

## Sample Input

*For the sake of space, small value of n is chosen.*

2 -1 -1 2

1 3 3 -1

0 0 -3 0

3 -2 -2 -1

2 0 4 -3

0 1 1 2

## Sample Output

1 1 2 1

1 2 2 2

1 2 1 3

2 1 2 2

2 2 2 3

3 1 3 2

3 2 2 2

2 3 3 3

## Explanation

There are 12 edges to be considered. They are given as follows:

- $a_{11} - a_{12} = (2, -1)$
- $a_{12} - a_{13} = (-1, 2)$
- $a_{21} - a_{22} = (1, 3)$
- $a_{22} - a_{23} = (3, -1)$
- $a_{31} - a_{32} = (0, 0)$
- $a_{32} - a_{33} = (-3, 0)$
- $a_{11} - a_{21} = (3, -2)$
- $a_{21} - a_{31} = (-2, -1)$
- $a_{12} - a_{22} = (2, 0)$

- $a_{22} - a_{32} = (4, -3)$
- $a_{13} - a_{23} = (0, 1)$
- $a_{23} - a_{33} = (1, 2)$

This is also the order in which the complex numbers are given in the input. The score of the given output is  $\sqrt{(3+2-1+1+3+0+4+1)^2 + (-2+0+2+3-1+0-3+2)^2} = 13.038405$ . There may of course be better possible score!

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » [Coloring in Hypercube](#)

## Coloring in Hypercube Problem Code: HPARITY

In an  $N$ -dimensional grid the co-ordinates of a cell are denoted as  $X_1, X_2, \dots, X_N$ . Any cell with negative co-ordinate are colored white. The origin cell (the one with all zero co-ordinates) is colored as black. The color of a cell in  $(X_1, X_2, \dots, X_N)$  depends on the  $N$  cells with co-ordinates  $(X_1-1, X_2, \dots, X_N), (X_1, X_2-1, \dots, X_N), \dots, (X_1, X_2, \dots, X_N-1)$ . The cell is colored white if and only if the number of black colored cells among these  $N$  co-ordinates are even, otherwise the cell is colored black.

You are given the starting and ending co-ordinate of sub-hypercube. You need to compute how many hyper cells in this sub hypercube are colored black.

### Input

First line of the input contains  $T$  the number of test cases. Each test case starts with a line containing  $N$  the dimension of the hypercube. The second line contains  $N$  integers denoting the co-ordinate of the starting cell of the hypercube. The third line contains  $N$  integers denoting the co-ordinate of the ending cell of the hypercube.

### Output

For each test case, output the number of black colored cells in the given hypercube. Since the result can be too big so output the result modulo 1000000009.

### Example

#### Input

3

3

4 0 4

7 9 8

2

0 3

10 9

4

0 3 0 2

6 8 1 5

**Output:**

9

32

22

**Constraints**

$T < 51$

$0 < N < 9$

All the co-ordinates will be non negative integers with at most 15 digits.

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » Block Drop

## Block Drop Problem Code: BLOCKDRO

There is a square pool divided into  $N \times M$  cells. In some cells of the pool there stone islands. Each island consists of some number of stones. Let's call this number the height of the island. You can jump from the island with coordinates  $(x, y)$  to any island with coordinates  $(x+1, y)$ ,  $(x+2, y)$ ,  $(x-1, y)$ ,  $(x-2, y)$ ,  $(x, y+1)$ ,  $(x, y+2)$ ,  $(x, y-1)$ ,  $(x, y-2)$ . When you jump off the island its height goes down by one. If the height of any island becomes 0 it goes under water and you can't jump on it any more. You start on island with coordinates  $(sx, sy)$ . The goal is to make all the islands (except the final one) go under water and finish on the island with coordinates  $(fx, fy)$  which should have height equal to 1 when you finish on it. You task is to count the number of different ways to achieve the goal.

### Input

The first line of input file contains number  $t$  - the number of test cases (no more than 5 for each file). Then the description of each test case follows. The first line of each test case contains numbers  $N$  and  $M$ . The next line contains two coordinates  $sx$  and  $sy$  of the start

cell. After that there are two coordinates  $fx$  and  $fy$  of the finishing cell. Then  $N$  lines follow each consisting of  $M$  integers denoting the heights of the islands in each cell of the pool. The height 0 means that there is no island in the cell. Note also that each test case in the official tests will be generated by the following procedure. The dimensions of the pool will be chosen randomly and uniformly:  $N$  and  $M$  will be from 3 to 8 inclusive. Then the final cell will be chosen randomly:  $fx$  will be from 1 to  $N$  and  $fy$  will be from 1 to  $M$ . The height of island in the cell  $(fx, fy)$  will be set to 1. Then the number of jumps will be chosen randomly from 15 to 25. Then this many random valid jumps will be performed adding one stone to the cell the jumps lands on. The cell on which we land after performing all the jumps will be proclaimed as the initial cell:  $(sx, sy)$ .

## Output

For each test case print the total number of different ways to solve the task.

## Example

Input:

```
1
3 3
3 1
3 1
1 0 0
3 1 1
3 1 1
```

Output:

```
152
```

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » Shortest Circuit Evaluation

## Shortest Circuit Evaluation Problem Code: SHORTCIR

Short circuit evaluation of Boolean expressions denotes the semantic in which the second argument of some Boolean operator is not evaluated if the value of the first argument is enough to have the result. This technique is used in many programming languages to optimize the evaluation of Boolean expressions. Specifically for "A and B", if A is false we

know that the whole expression is false and we don't need to evaluate B. For "A or B", if A is true we know the result to be true. Now having that those Boolean operations are commutative we may actually evaluate B first and not evaluate A in case B gives us the result. Moving the idea further if we have "A1 or A2 or...or An" we can evaluate the variables in any order and as soon as we have one of them as true we know that the whole expression is true. We can do similarly for and operation. Now let's consider some complex Boolean expression. We will fix the order in which we will evaluate the variables of the expression. Then we evaluate those variables in that order and we won't evaluate the variables that give us no new information about the value of the whole expression in the process. For example, assume we have "A and B or C" and we fix the order of evaluation B, A, C. First we evaluate B, if it's false we don't have to evaluate A and only evaluate C. However if B is true we will need to evaluate A. If A is true we know the expression is true and won't evaluate C, otherwise we evaluate C to have the value of the expression. Now your task is having some complex Boolean expression containing and, or, not operations and for each variable having the probability that this variable is true, you need to find the order of evaluation for which the expected number of evaluations in the process described above will be minimal.

## Input

The first line of input file contains number t - the amount of test cases. Then t test cases follow. The first line of each test case will contain the Boolean expression. The expression will be valid and will consist of and, or, not operations, brackets and variable names. All the variable names in one expression will be distinct. Then for each variable present in the expression there will be a line in the input in the format s p, where s - the name of the variable and p - is the probability that the variable will be true. The names of the variables will consist of small Latin letters only.

## Constraints

$1 \leq t \leq 50$

$0 < p < 1$

The length of the expression won't exceed 30000 characters.

There will be no more than 1000 variable in the expression.

The length of the variable names won't exceed 5 characters.

***Also the expression will be in the form of either conjunctive or disjunctive normal form.***

## Output

For each test case output the expected number of evaluations for the optimal order of evaluation of variables for short circuit evaluation process described above. Output the answer with 6 digits after the dot.

## Example

Input:

3

(a and b) or c

a 0.3

b 0.4

c 0.5

(a or b) and (not d or c)

a 0.5

b 0.3

c 0.8

d 0.25

ab or bc or cd

ab 0.3

bc 0.1

cd 0.2

#### Output:

1.650000

2.280000

2.260000

## Explanation

In the first test case the best order is c, a, b.

[Home](#) » [Compete](#) » [August Long Contest 2011](#) » Something About Divisors

# Something About Divisors Problem Code: DIVISORS

For a given positive integers **B** and **X** find the number of positive integers **N** such that number **N\*X** has at least one divisor **D** such that **N < D <= B**.

## Input

The first line contains a single positive integer **T <= 40**, the number of test cases. **T** test cases follow. The only line of each test case contains two positive integers **B <= 10<sup>12</sup>** and **X <= 60**.

## Output

For each test case, output a single line containing the answer for the corresponding test case.

## Example

### Input:

3

5 1

10 3

100 6

### Output:

0

5

63

## Explanation

In the second test case required numbers are 1, 2, 3, 4, 6.

## Open the Dragon Scroll Problem Code: DRAGNXOR

Did you ever hear about 'Dragon Food' ? Its used to refer to the chocolates bought for your loved ones :). Po offers dragon food to master Shifu, who is a famous cook in the valley of food. In return, Shifu hands over the dragon scroll to Po, which is said to hold the ingredients of the secret recipe. To open the dragon scroll, one has to solve the following puzzle.

1. Consider a N-bit integer A. We call an integer A' as shuffle-A, if A' can be obtained by shuffling the bits of A in its binary representation. For eg. if N = 5 and A = 6 =  $(00110)_2$ , A' can be any 5-bit integer having exactly two 1s in it i.e., any of  $(00011)_2$ ,  $(00101)_2$ ,  $(00110)_2$ ,  $(01010)_2$ , ...,  $(11000)_2$ .
2. Given two N-bit integers A and B, find the maximum possible value of  $(A' \text{ xor } B')$  where A' is a shuffle-A, B' is a shuffle-B and xor is the bit-wise xor operator.

Given N, A and B, please help Po in opening the dragon scroll.

**Notes**

1. xor operator takes two bit strings of equal length and performs the logical XOR operation on each pair of corresponding bits. The result in each position is 1 if only the first bit is 1 OR only the second bit is 1, but will be 0 if both are 1 or both are 0. For eg: 5  $(0101)_2$  xor 3  $(0011)_2$  = 6  $(0110)_2$ . In most languages it is represented using ^ symbol.  $5 \wedge 3 = 6$ .
2. If the integer actually needs less than N bits to represent in binary, append sufficient number of leading 0 bits. For eg. as shown in the problem statement for N = 5, A = 6 =  $(00110)_2$

### Input

First line contains an integer T ( number of test cases, around 100 ). T cases follow, each having N A B in a single line, separated by a space. (  $1 \leq N \leq 30$ ,  $0 \leq A, B \leq 2^N$  )

### Output

For each case, output the maximum possible value of (shuffle-A xor shuffle-B) in a separate line.

### Example

**Input:**

3

3 5 4

5 0 1

4 3 7

**Output:**

7

16

14

**Explanation:**

Case 1: 5 and 4 as 3-bit binary strings are  $(101)_2$  and  $(100)_2$  respectively. After shuffling, xor can be maximum for  $(110)_2 \wedge (001)_2 = (111)_2 = 7$

Case 2: Maximum Possible result can be for  $(00000)_2 \wedge (10000)_2 = (10000)_2 = 16$

Case 3: Maximum Possible result can be for  $(0011)_2 \wedge (1101)_2 = (1110)_2 = 14$

[Home](#) » [Compete](#) » [August Cook-off 2011](#) » Vote for the Noodle Soup

## Vote for the Noodle Soup Problem Code: NDLVOTE

Did you ever hear about 'crossing the bridge noodle' ? Let me tell you that it's not some kind of bridge made of noodles. It's a dish, a kind of rice noodle soup. Mr.Ping makes the best noodle soup and his son Po is eagerly waiting for the user reviews in his father's blog. Users can vote with a (+) or a (-) and accordingly +1 or -1 is added to the total score respectively. Note that if a user votes multiple times, only his/her latest vote is counted towards the total score.

Po opens the blog to see initial score of 0. To see the updated score without refreshing the page, he has to keep voting himself. After each of Po's clicks on (+) or (-), he can see the current total score, of course that considers Po's vote too. He is wondering how many users other than him could have possibly voted. Given the sequence of clicks made by Po and the total score displayed just after each of his clicks, can you tell him the minimum number of users that could have possibly voted at least once, other than Po.

### Input

There are multiple test cases ( at most 21 ). Each case starts with an integer  $N$  ( $1 \leq N \leq 1000$  ), the number of Po's clicks. Sequence of  $N$  clicks follows, one on each line of the form "vote score" (without quotes, separated by a space), where *vote* is either a 'P' or a 'M', representing Plus(+) and Minus(-) respectively, the one clicked by Po and *score* is the score displayed after Po's click ( $-1,000,000,000 \leq \text{score} \leq 1,000,000,000$  ). The last case has  $N = 0$  and should not be processed. Each case is followed by an empty line.

## Output

For each test case, output the minimum number of users that could have possibly voted at least once.

## Example

### Input:

2

P 1

P 2

2

P 2

M -2

0

### Output:

1

1

## Explanation:

Case 1 :

P 1 , Po voted (+) and score = 1 is possibly Po's vote itself.

P 2 , Po again voted (+), but as only latest vote of a user is counted, Po contributed only +1 to the score, so possibly one more user voted with a (+). Remember that we have to find the number of users other than Po, so answer is 1

Case 2 :

P 2 , Po voted (+) and score = 2, possibly another user A also voted (+)

M -2 , Po voted (-) and score = -2. Possibly the same user A also voted (-)  
So there can possibly be just one other user A

[Home](#) » [Compete](#) » [August Cook-off 2011](#) » Rotate the String

## Rotate the String Problem Code: ROTSTRNG

Did you know that a cluster of bananas can also be called a 'hand' of bananas ? No points for guessing what the individual bananas are called. ya.. 'fingers' :). The monkey from the valley of food is all set to fight for the title 'Dragon Chef' after having a hand of bananas for breakfast. As you know that the judge, master Oogway is always late for the show, the monkey and Po decided to play a game. They took a string of characters and started rotating it.

A k-rotation on a string takes the trailing k characters of the string and attaches it to the beginning of the string in the same order. For eg. 3-rotation on the string "noodles" results in the string "lesnood". On the given string **S**, the monkey performs a **M**-rotation and gives it to Po. Po performs a **P**-rotation on it and gives back to the monkey. The monkey again performs a **M**-rotation on it and so on... Though Shifu is getting angry watching the monkey play with Po, he is wondering after how many minimum (non-zero) number of rotations we can get back to the original string **S**. This can happen after a rotation performed by the monkey or Po. Please find it for him.

### Input

First line contains **T** ( number of test cases, around 10 ). **T** cases follow. Each case consists of two lines. First line has the string **S** of length **n** (  $1 \leq n \leq 500,000$  ), having ('a'-'z' , 'A'-'Z'). Second line contains two integers **M P** separated by a space (  $1 \leq M, P \leq n$  ). Comparisons are case sensitive i.e., 'a' is not equal to 'A'.

### Output

For each test case, output the answer in a new line. If its impossible to get back to the original string by performing the rotations as mentioned above, print -1

### Example

**Input:**

2

AbcDef

1 2

abcabc

1 1

**Output:**

4

3

**Explanation**

Case 1 :

Monkey : fAbcDe , Po : DefAbc, Monkey : cDefAb , Po : AbcDef

Case 2 :

Monkey : cabcab , Po : bcabca, Monkey : abcabc

*Warning : Large input / output. You may have to use efficient input / output methods if you are using languages with heavy input / output overload. Eg: Prefer using scanf/printf to cin/cout for C/C++*

[Home](#) » [Compete](#) » [August Cook-off 2011](#) » Collect the Chocolate Chips

## Collect the Chocolate Chips Problem Code: TKCHOCS

Did you know that chocolate chip cookie was invented by accident ? They even declared May 15 as National Chocolate Chip day :). Po and Mantis love choco chips and they want to collect them on the board. The board is in the shape of a right-angled triangle having **N** rows, with  $i^{th}$  row ( 1-based indices, as shown below for  $N=4$  ). Cell  $(i,j)$  refers to the cell in  $j^{th}$  column in the  $i^{th}$  row.

Po starts from the top-most cell  $(1,1)$  and from a cell  $(i,j)$  Po can move to any of the cells  $(i+1,j-1)$ ,  $(i+1,j)$  or  $(i+1,j+1)$  in one step, if it exists. Mantis starts from the right-most cell  $(N,N)$  and from a cell  $(i,j)$  Mantis can move to any of the cells  $(i-1,j-1)$ ,  $(i,j-1)$  or  $(i+1,j-1)$  in one step, if it exists. When Po or Mantis are in a cell, they can collect all the choco chips in it. They both want to reach the bottom-left cell  $(N,1)$  after collecting as many choco chips as possible. Find the maximum number of choco chips they can collect i.e., maximize the sum of chips collected by each of them.

	1	2	3	4
1	<i>Po</i> (1,1)			
2	(2,1)	(2,2)		
3	(3,1)	(3,2)	(3,3)	
4	(4,1) <i>Exit</i>	(4,2)	(4,3)	(4,4) <i>Mantis</i>

## Input

First line contains an integer T (number of test cases, around 10). T cases follow. Each case starts with a line having an integer N (  $2 \leq N \leq 500$  ). N lines follow with ith line having i non-negative integers from range [0 - 1,000,000], number of chips in each cell as explained above.

## Output

For each case, output the maximum number of choco chips Po and Mantis can collect, in a separate line.

## Example

**Input:**

2

2

1

1 1

3

1

2 5

1 3 1

**Output:**

3

11

**Explanation:**

Case 2 : Po can visit (1,1) -> (2,2) -> (3,1) and Mantis can visit (3,3) -> (3,2) -> (3,1). Note that if both step in to a same cell, we must count the choco chips in it only once in the final answer.

The cell (3,1) is reached by both, but the 1 choco chip in it must be added only once ( of course : ) .

[Home](#) » [Compete](#) » [August Cook-off 2011](#) » Angry Chef - Crispy Chips

## Angry Chef - Crispy Chips Problem Code: KCHIPS

Did you know that potato was the first food that was ever grown in space ? Today is a big day for chef Crum, as he was called to make potato dishes for the people of the Valley of Food. He got angry knowing someone said that his potatoes were thick and so he started making them too thin and crisp ! (and thus invented potato chips :)). There are **N** persons sitting in a row ( numbered 0 to N-1 ) and you are given an array **V**, where  $V[i]$  is the village number, where the  $i^{\text{th}}$  person is from.

In each of the **R** rounds, Po serves potato chips to a group of people sitting continuously (a sub-array). Shifu is worried that, if in a round, more than **K** people from a village are served potato chips, others may protest and that leads to disruption of outer peace. To estimate the damage of a round, he wants to know how many distinct villages are there such that more than **K** people from each of them are served in that round.

**Input**

First line contains two integers **N K** ( **N** is the number of persons,  $1 \leq N \leq 100,000$  and **K** is the limit in each round  $0 \leq K \leq N$  ). Next line contains **N** numbers, the array **V** (as described above.  $0 \leq V[i] \leq 100,000,000$  ). Third line contains **R** (number of rounds,  $1 \leq R \leq 100,000$  ). **R** rounds follow, each in a new line, having integers **i j** ( $0 \leq i \leq j < N$  ). Po serves all the persons [ **i, i+1, ..., j** ] in that round.

**Output**

For each of the **R** rounds, print the answer in a new line.

## Example

### Input:

8 2

3 1 2 2 1 1 2 1

3

0 5

2 4

1 7

### Output:

1

0

2

### Explanation:

$V[0..5] = \{ 3, 1, 2, 2, 1, 1 \}$  : Only Village 1 occurs more than  $K (= 2)$  times.

$V[2..4] = \{ 2, 2, 1 \}$  : None of the villages occur more than 2 times.

$V[1..7] = \{ 1, 2, 2, 1, 1, 2, 1 \}$  : Villages 1 and 2 occur more than 2 times.

Warning : Large input / output. You may have to use efficient input / output methods if you are using languages with heavy input / output overload. Eg: Prefer using `scanf/printf` to `cin/cout` for C/C++

Note : There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted. Time limit on each test set is 2 sec

SEPT11

[September Long  
Contest](#)

01 Sep 2011  
15:00:00

10 days

710

[Home](#) » [Compete](#) » [September Long Contest](#) » [BIT Magazine](#)

# BIT Magazine Problem Code: TRMAG

Taru likes reading. Every month he gets a copy of the magazine "BIT". The magazine contains information about the latest advancements in technology. Taru reads the book at night and writes the page number to which he has read on a piece of paper so that he can continue from there the next day. But sometimes the page number is not printed or is so dull that it is unreadable. To make matters worse Taru's brother who is really naughty tears of some of the pages of the Magazine and throws them in the dustbin. He remembers the number of leaves he had torn but he does not remember which page numbers got removed. When Taru finds this out he is furious and wants to beat him up. His brother apologizes, and says he won't ever do this again. But Taru did not want to be easy on him and he says "I will leave you only if you help me find the answer to this. I will tell you how many pages (Printed sides) were there in the Magazine plus the pages on which the page numbers were not printed. You already know the number of leaves you tore (T). Can you tell me the expected sum of the page numbers left in the Magazine?" Taru's brother replied "huh!! This is a coding problem". Please help Taru's brother.

Note: The magazine is like a standard book with all odd page numbers in front and the successive even page number on its back. If the book contains 6 pages, Page number 1 and Page number 2 are front and back respectively. Tearing a leaf removes both the front and back page numbers.

## Input

The first line contains the number of test cases  $t$ .  $3t$  lines follow. The first line of each test case contains the number of pages (printed sides) in the book. The second line's first integer is  $F$ ,  $F$  integers follow which tell us the numbers of the page numbers not printed. The third line contains a single integer telling us the number of leaves Taru's brother tore.

## Output

Output one real number correct up to 4 decimal digits which is equal to the expected sum of the page numbers left in the book.

## Constraints

Number of printed Sides $\leq 2000$ . All other values abide by the number of printed sides.

## Example

**Input:**

2

10

2 1 2

2

10

1 8

0

**Output:**

31.2000

47.0000

[Home](#) » [Compete](#) » [September Long Contest](#) » Younger Brother

## Younger Brother Problem Code: CHEFBRO

Chef's younger brother is in town. He's a big football fan and has a very important match to watch tonight. But the Chef wants to watch the season finale of MasterChef which will be aired at the same time. Now they don't want to fight over it like they used to when they were little kids. They want to decide it in a fair way. So they agree to play a game to make a decision. Their favourite childhood game!

The game consists of C boards. Each board  $i$  is a grid of dimension  $n_i \times m_i$ .

**Rules of the game:**

- A coin is placed at  $(1,1)$  on every board initially.
- Each one takes a turn alternatively.
- In one turn, a player can choose any **one** board and move a coin from a cell  $(i,j)$  to one of the following cells:  
 $(i+1,j)$  OR  $(i+2,j)$  OR  $(i,j+1)$  OR  $(i,j+2)$  OR  $(i+1,j+1)$  OR  $(i+2,j+2)$ .
- A coin cannot be moved out of the board at any point during the game.
- A coin cannot be moved once it reaches the cell  $(n,m)$  where  $n$  and  $m$  are the dimensions of the board of that coin.
- A player **MUST** make one valid move.
- The player who makes the last move gets to watch TV.

Both of them are passionate about their interests and want to watch their respective shows. So they will obviously make optimal moves in every turn. The Chef, being the elder brother, takes the first turn.

Your task is to predict which show they will be watching tonight.

**Input:**

The first line of input contains a single integer  $T$ , the number of test cases.  $T$  tests follow. Each test case starts with a single line containing  $C$ , the number of boards in the game. Then follow  $C$  lines: each containing 2 integers  $n_i$  and  $m_i$ , the dimensions of the  $i$ th board.

## Output:

Given the number and dimensions of boards, for each test case, output in a single line: "MasterChef" if the Chef wins or "Football" if his brother wins.

## Constraints:

$1 \leq T \leq 10000$

$1 \leq C \leq 20$

$2 \leq n_i, m_i \leq 1000$

## Example:

### Input:

1

1

2 2

### Output:

MasterChef

### Explanation:

The Chef can move the coin on the board from  $(1,1) \rightarrow (2,2)$ . This coin cannot be moved any further. And so, the Chef wins. Notice that if the Chef moves it to any other valid position, i.e. either to  $(1,2)$  or  $(2,1)$  he will lose!

[Home](#) » [Compete](#) » [September Long Contest](#) » Haunted Maze

## Haunted Maze Problem Code: HAUNTED

Chef is trapped in a haunted maze. There are ghosts that patrol the maze by following fixed paths. Chef is trying to escape the maze, but must avoid the ghosts or he will become frightened and faint. Additionally, Chef needs to escape within a certain amount of time, or he will faint from hunger. Your task is to determine if Chef can escape the maze without fainting, and if so the minimum time it will take.

You will be given the layout of the maze, including Chef's starting position, the position of the exit, and the patrol paths of the ghosts. There is a timer that keeps track of how long Chef has been in the maze. The timer starts at 0. Each second, the following happens (in order):

- If the timer is equal to the maximum time Chef is allowed to spend in the maze, he will faint from hunger.
- Chef may move to one of the adjacent cells of the maze (north, east, south, or west of his current position), provided that cell is neither a wall nor occupied by a ghost. Chef also has the option of standing still.
- All of the ghosts move to the next cell in their respective patrol paths. If any ghost moves into the cell now occupied by Chef, Chef will faint.
- The timer is incremented by 1.
- If Chef is now standing at the location of the exit, Chef successfully escapes the maze.

Ghost patrol paths will be given as a list of waypoints. Each waypoint will be in-line either horizontally or vertically with the next waypoint in the list, and the final waypoint will similarly be in-line with the first waypoint. Ghosts will move from waypoint to waypoint, one square at a time, using the shortest possible path. Once a ghost reaches its final waypoint, it will travel back to the first waypoint and repeat the cycle. For example, if a patrol path were given as the waypoints:

(2,3) (1,3) (1,0) (0,0) (0,1) (2,1)

Then the ghost's complete path would be:

(2,3) (1,3) (1,2) (1,1) (1,0) (0,0) (0,1) (1,1) (2,1) (2,2)

After which the sequence would repeat.

## Input:

Input will begin with  $T$ , the number of test cases. Each test case begins with 4 integers:  $M$ ,  $N$ ,  $C$ , and  $K$ , where  $M$  is the height of the maze,  $N$  is the width of the maze,  $C$  is the number of ghosts, and  $K$  is the number of seconds before Chef will faint from hunger.  $C$  lines follow, each describing the patrol path of a ghost. Each patrol path begins with an integer  $L$ , the number of waypoints in the path, followed by  $L$  pairs of integers giving the  $(x,y)$  coordinates of a waypoint on the path, where  $(0,0)$  is the top-left corner and  $(N-1, M-1)$  is the bottom-right corner. Finally,  $M$  lines of  $N$  characters each describe the maze itself. A '.' character indicates a passable square, and a '#' character indicates an impassable wall (although the ghosts may pass through walls). A '@' character indicates the starting position of Chef (it is guaranteed that no ghost will begin here), and a "character indicates the exit of the maze (there will be exactly one '@' character and one "character indicates the exit of the maze (there will be exactly one '@' character and one " character in each test case).

## Output:

For each test case, if Chef can escape without fainting, print the minimum number of seconds it will take him to escape. Otherwise, print -1.

## Constraints:

$T \leq 10$

$1 \leq M, N \leq 30$

$0 \leq C \leq 30$

0 ≤ K ≤ 100000  
2 ≤ L ≤ 10

### Sample Input:

4

10 10 0 200

```
##. ....
@#.#####
.#...#..
.###...#.
....###.#
#.#. ....#
....#.###.
.####.#.#
.#....#...
.....
3 10 1 18
2 1 1 7 1
####.#####
@.....@#.#####..#...#.###....#....###.#.#. ....#....#.###..#####
.#.#..#....#.....3 10 1 182 1 1 7 1###.#####@.....
#####.###
5 5 2 13
2 2 4 2 1
3 4 2 3 2 2 2
##@##
##.##
..##
#####.
####.
2 2 1 1000
2 1 1 1 0
@..#####2 2 1 10002 1 1 1 0@
```

..

### Sample Output:

22

18

-1

-1

[Home](#) » [Compete](#) » [September Long Contest](#) » Short

## Short Problem Code: SHORT

Given  $n$  and  $k$ , find the number of pairs of integers  $(a, b)$  such that  $n < a < k$ ,  $n < b < k$  and  $ab - n$  is divisible by  $(a-n)(b-n)$ .

### Input

The first line contains the number of test cases  $t$  ( $1 \leq t \leq 5$ ). Then  $t$  test cases follow, each test case consists of a line containing two integers  $n$  and  $k$  ( $0 \leq n \leq 100000$ ,  $n < k \leq 10^{18}$ ).

### Output

For each test case output one line containing the required number.

### Example

#### Input:

2

1 5

2 5

#### Output:

2

3

#### Explanation:

In the first test case, the only sought pairs are  $(2,2)$  and  $(3,3)$ . In the second one, they are  $(3,3)$ ,  $(3,4)$  and  $(4,3)$ .

[Home](#) » [Compete](#) » [September Long Contest](#) » Counting Hexagons

# Counting Hexagons Problem Code: CNTHEX

Dexter wants to make a hexagon by using six sticks for sides from a collection of sticks of lengths **1, 2, 3, ... N**. Dexter will only make **valid** hexagons whose area is strictly positive.

He considers a hexagon "good" if the biggest stick has **length at least L** and lengths of all the other sticks **are not more than X**. A "good" hexagon does not have sticks of the **same length more than K times**. How many ways he can make a "good" hexagon?

For example, when  $N=8$ ,  $L=7$ ,  $X=5$ ,  $K=3$ :  $\{1,2,2,5,5,7\}$  is a good hexagon but  $\{1,2,2,2,2,7\}$ ,  $\{1,2,3,4,5,6\}$ ,  $\{1,2,3,4,6,7\}$  are not .

Two hexagons are considered different if their side length sets are different. For example  $\{1,2,3,4,5,6\}$ ,  $\{1,1,1,2,2,3\}$  and  $\{1,1,2,2,2,3\}$  are all different hexagons. But  $\{1,2,3,4,5,6\}$  and  $\{2,4,6,1,3,5\}$  are not different.

## Input

Input contains four integers **N** ( $2 \leq N \leq 10^9$ ), **L** ( $2 \leq L \leq N$  and  $N-L \leq 100$ ), **X** ( $1 \leq X < L$ ) and **K** ( $1 \leq K \leq 5$ ).

## Output

Output the number of different ways to make a "good" hexagon **% 1000000007**.

## Sample Input

10 8 6 2

## Sample Output

374

[Home](#) » [Compete](#) » [September Long Contest](#) » Table Tilt

# Table Tilt Problem Code: TILT

Chef recently discovered a game called "Table Tilt". The game consists of a rectangular table which has several holes in it, and several balls placed on the table. The goal is to tilt the table so that the balls will roll into the holes. The edges of the table are blocked so that a ball cannot fall off the edge of a table. By careful tilting of the table, Chef is able to precisely control the movement of the balls. Chef controls the tilting so that each second, each ball will simultaneously attempt to roll 1 unit in the same direction parallel to an edge of the table. A ball will fail to move if doing so would cause it to fall off the table, or if doing so would result in 2 balls in the same location after the move.

There are different types of balls, and they have different effects when they roll into a hole. The ball types, along with their shorthand notations, and effects, are as follows:

- Neutral (=): no effect
- Positive (+): score increases by 5
- Negative (-): score decreases by 5
- Inverter (%): all positive balls become negative, and vice versa

Additionally, the score decreases by 1 every second until all of the balls have rolled into holes, at which point the game ends. Note that multiple balls may roll into holes simultaneously. In this case, inverter balls are processed last. That is, if a negative ball and an inverter ball simultaneously roll into holes, the score is decreased by 5 before the remaining negative balls become positive.

Additionally, inverter balls are processed one at a time, so if 2 inverter balls simultaneously roll into holes there is no net effect.

Since Chef has mastered his tilting skill, he now needs strategy to improve his score. He has tasked you with finding a sequence of tilts that gives a good score. Chef is in a good mood and is not demanding an optimal sequence of tilts, but merely requires you find one that yields a positive score (of course, the higher the score, the more appreciative Chef will be).

## Input

Input begins with an integer  $T$ , the number of test cases. Each test case begins with 2 integers  $H$  and  $W$  on a line, giving the height and width, respectively, of the table.  $H$  lines follow of  $W$  characters each, giving the initial layout of the table. A '\*' character indicates a hole. All other characters will be '=', '+', '-' or '%', indicating neutral, positive, negative, and inverter balls, respectively.

## Output

For each test case, output a single line with a sequence of 'U', 'L', 'D', and 'R' characters, corresponding to tilts up, left, down, and right, respectively. This sequence of tilts must be one that completes the game with a positive score.

## Scoring

Your total score is the sum of the scores for the individual test cases. Note that if the score for any individual test case is non-positive, the submission will be judged "Wrong Answer".

## Sample Input

2

3 3

%+-

=\*+

-=+

3 4

+==-

- -%\*

+%-\*

## Sample Output

LDRDLRULR

RDLRRRDR

The score for the first test case after each tilt is: 4, 8, 7, 6, 10, 9, 8, 2, 6.

The score for the second test case after each tilt is: -6, -2, -3, 1, 5, 14, 13, 12.

The total score is therefore 18. Note that better scores may be possible, but the only requirement is that the score be positive.

## Test Case Generation

For each official input file, T is 5. For each test case, H and W are chosen randomly and uniformly between 8 and 20. Each cell of the table is chosen with the following probabilities:

- Hole: 2%
- Inverter ball: 2%
- Neutral ball: 32%
- Positive ball: 32%
- Negative ball: 32%

Furthermore, it is guaranteed there will be at least one hole and one inverter ball.

COOK14

[September Cook-off](#)

18 Sep 2011  
21:30:00

2 hours 31 minutes

566

[Home](#) » [Compete](#) » [September Cook-off](#) » Hotel Bytelandia

## Hotel Bytelandia Problem Code: HOTEL

A holiday weekend is coming up, and Hotel Bytelandia needs to find out if it has enough rooms to accommodate all potential guests. A number of guests have made reservations. Each reservation consists of an arrival time, and a departure time. The hotel management

has hired you to calculate the maximum number of guests that will be at the hotel simultaneously. Note that if one guest arrives at the same time another leaves, they are never considered to be at the hotel simultaneously (see the second example).

## Input

Input will begin with an integer  $T$ , the number of test cases. Each test case begins with an integer  $N$ , the number of guests. Two lines follow, each with exactly  $N$  positive integers. The  $i$ -th integer of the first line is the arrival time of the  $i$ -th guest, and the  $i$ -th integer of the second line is the departure time of the  $i$ -th guest (which will be strictly greater than the arrival time).

## Output

For each test case, print the maximum number of guests that are simultaneously at the hotel.

## Sample Input

```
3
3
1 2 3
4 5 6
5
1 2 3 4 5
2 3 4 5 6
7
13 6 5 8 2 10 12
19 18 6 9 9 11 15
```

## Sample Output

```
3
1
3
```

## Constraints

- $T \leq 100$
- $N \leq 100$
- All arrival/departure times will be between 1 and 1000, inclusive

[Home](#) » [Compete](#) » [September Cook-off](#) » Digit Rotation

## Digit Rotation Problem Code: DIGROT

For any positive integer, we define a digit rotation as either moving the first digit to the end of the number (left digit rotation), or the last digit to the front of the number (right digit rotation). For example, the number 12345 could be left digit rotated to 23451, or right digit rotated to 51234. If there are any leading zeros after digit rotation, they must be removed. So 10203 could be left digit rotated to 2031, then left digit rotated again to 312. Given an integer  $N$ , determine the largest integer that can result from performing a series of one or more digit rotations on  $N$ .

### Input

Input will begin with an integer  $T$  (at most 1000), the number of test cases. Each test case consists of a positive integer  $N < 100000000$  ( $10^8$ ) on a line by itself.

### Output

For each test case, print the largest integer that can result from performing one or more digit rotations on  $N$ .

### Sample Input

6

12345

54321

10901

211011

7

90

### Sample Output

51234

54321

11090

211011

7

9

[Home](#) » [Compete](#) » [September Cook-off](#) » Last Digit Sum

## Last Digit Sum Problem Code: LASTDIG

For a non-negative integer  $N$ , define  $S(N)$  as the sum of the odd digits of  $N$  plus twice the sum of the even digits of  $N$ . For example,  $S(5)=5$ ,  $S(456)=2*4+5+2*6=25$ , and  $S(314159)=3+1+2*4+1+5+9=27$ . Define  $D(N)$  as the last digit of  $S(N)$ . So  $D(5)=5$ ,  $D(456)=5$ , and  $D(314159)=7$ . Given 2 non-negative integers  $A$  and  $B$ , compute the sum of  $D(N)$  over all  $N$  between  $A$  and  $B$ , inclusive.

### Input

Input will begin with an integer  $T$ , the number of test cases.  $T$  lines follow, each containing 2 integers  $A$  and  $B$ .

### Output

For each test case, output a single integer indicating the corresponding sum.

### Sample Input

3

1 8

28 138

314159 314159

### Sample Output

36

495

7

## Constraints

- $T \leq 1000$
- $0 \leq A \leq B \leq 400,000,000$

[Home](#) » [Compete](#) » [September Cook-off](#) » Target Practice

# Target Practice Problem Code: TARRACT

Chef has recently taken up marksmanship, but he's not very good at it. Chef has set up a number of targets in a line. When Chef shoots at a target, he may hit the target, but he may instead hit the target to the left, or the target to the right (or if there is no such target, miss completely). Chef knows the probability of his shot going straight, left, or right, and wants to hit each target at least once using as few shots as possible.

Help Chef determine the expected number of shots required, provided an optimal strategy is followed. Note that Chef may only shoot at targets, but may shoot at a target that's already been hit.

## Input

Input will begin with an integer  $T$ , the number of test cases. Each test case will consist of 4 integers:  $N$ , the number of targets,  $L$ , the probability of the shot veering to the left,  $S$ , the probability of the shot going straight, and  $R$ , the probability of the shot veering to the right.  $L$ ,  $S$ , and  $R$  are given as percents.

## Output

For each test case, output the expected number of shots to hit all targets, rounded to 6 decimal places.

## Sample Input

4

3 25 50 25

3 10 10 80

5 25 50 25

8 30 35 35

## Sample Output

4.666667

10.361111

7.541667

13.107725

In the first example, Chef shoots at the center target until any 2 targets have been hit, then he shoots at the remaining target.

## Constraints

- $T \leq 30$
- $1 \leq N \leq 18$
- $0 < L, S, R$
- $L + S + R = 100$

[Home](#) » [Compete](#) » [September Cook-off](#) » Yet Another Sequence

# Yet Another Sequence

Problem Code: YASEQ

You are given the first  $N$  terms of a sequence. The remaining terms of the sequence are defined as follows:

$A[i] = \text{number of } j \text{ such that } 0 \leq j < i \text{ and } A[j] \geq i - j$

The sequence is indexed starting at 0. The first  $N$  terms of the sequence will each be equal to  $N-1$ ,  $N$ , or  $N+1$ .

In addition to the first terms of the sequence, you will be given a number of queries. For each query you are to calculate the value of the sequence at that index.

## Input

Input will begin with an integer  $T$ , the number of test cases. Each test case begins with 2 integers  $N$  and  $Q$  on a line, followed by a line with  $N$  integers (each equal to  $N-1$ ,  $N$ , or  $N+1$ ), then a line with  $Q$  non-negative integers.

## Output

For each test case, output  $Q$  integers on a line, one per query, indicating the respective element of the sequence.

## Sample Input

3

3 4

3 3 3

100 1000 1000000 1

4 11

3 4 4 5

0 1 2 3 4 5 6 7 8 9 10

5 4

5 6 5 6 5

6 14 22 29

## Sample Output

3 3 3 3

3 4 4 5 3 4 4 4 4 4 4

5 5 5 5

## Constraints

- $T \leq 100$
- $0 < N \leq 100000$
- $0 < Q \leq 100000$
- Each query will be between 0 and  $10^{15}$ , inclusive
- The sum of  $N+Q$  over all test cases will not exceed 500000

OCT11	<a href="#">October Long Contest 2011</a>	01 Oct 2011 15:00:00	10 days	666
-------	---	-------------------------	---------	-----

[Home](#) » [Compete](#) » [October Long Contest 2011](#) » Dish Distribution

# Dish Distribution Problem Code: DISHDIS

Chef feels that being the Head Chef is a very difficult job. More than cooking, managing the kitchen and distributing the work between various cooks proves to be a tiring task. There are many dishes to be prepared. And he has a team of talented cooks to do this for him. But the Chef always has a problem distributing tasks among them. For a cook  $i$ , the Chef wants to give him *at least*  $x_i$  dishes to prepare in a day. But different cooks have different capacities. Any cook  $i$  can cook *at most*  $y_i$  number of dishes in a single day. Chef is very aware of every cook's capability and will not give anyone more work than his maximum capacity. Now, he wants you to help him out here.

Your task will be simple: Tell him the number of ways in which he can distribute the given number of dishes among his team.

## Input:

First line contains  $T$ , number of tests. Each test begins with a single line containing 2 space separated integers:  $n$  and  $m$ , the number of dishes to be prepared and the number of cooks in Chef's team.

Then follow  $m$  lines. The  $i$ th line contains 2 space separator integers which are the corresponding  $x_i$  and  $y_i$ .

## Output:

For every test case, output a single integer: the number of ways of distributing the work mod 1000000007.

## Constraints:

$1 \leq T \leq 50$   
 $1 \leq n, m \leq 100$   
 $0 \leq x_i, y_i \leq 100$

## Example:

### Input

```
1
3 2
0 3
1 3
```

### Output

```
3
```

### Explanation:

The chef can distribute the dishes in the following ways- 0 and 3 ; 1 and 2 ; 2 and 1 .

# The Great Plain Problem Code: LAND

You have been hired to create a map of The Great Plain. The first thing to do is to find out how the land's height changes, and that's exactly what you are to do.

The plain can be imagined as a rectangular grid with  $N$  rows and  $M$  columns consisting of  $N \times M$  equal cells. The height of each cell is an integer number between 1 and 50, inclusive.

You have already explored some of the cells, and now you know the height of these cells. Unfortunately, the budget is quite tight, so you have no opportunity to explore the remaining cells. With no other choice, you decided to fill the information about the unknown cells with fake data. But you wouldn't like to be quickly caught, so you want the resulting grid to resemble a 'plain' as much as possible.

Formally speaking, you are to fill the empty cells of a grid with integers between 1 and 50, inclusive, so that the neighboring cells don't differ too much (see the 'Scoring' part of the problem statement for the exact explanation on how the scoring works). Note that this is a challenge problem: you don't have to find the optimal solution, it's enough to find any of them (but the better is your solution, the more points you receive).

## Input

Input will begin with an integer  $T$ , the number of test cases (no more than 10). Each test case consists of 2 integers  $N$  and  $M$  ( $N, M \leq 100$ ), which denote the dimensions of the grid, and then  $N$  lines containing  $M$  integers each, representing the heights of the cells of the grid, which are between 1 and 50, inclusive. The value of 0 means that the corresponding cell is actually empty.

## Output

For each test case output exactly  $N$  lines containing exactly  $M$  integers (between 1 and 50, inclusive) each, representing the resulting grid. Note that you aren't allowed to change the values in the cells with non-zero values of the given grid. You may separate the answers for consecutive test cases with empty lines.

## Scoring

Let  $S$  be the sum calculated as follows: for each pair of side-by-side neighboring cells the value of  $2^K$  is added to  $S$ , where  $K$  is the absolute difference of heights of these two cells. Then your score for the test case is equal to  $\log_2 S$ . Your score for each file is the average of your scores on the individual test cases. Your overall score is the average of your scores on the individual test files.

Your goal is to minimize this score.

## Example

**Input:**

2

4 4

0 0 0 0

0 3 0 0

0 0 0 7

9 0 0 0

5 8

0 0 0 0 0 0 0

0 4 0 0 0 0 4 0

0 0 0 4 0 0 0 4

0 0 0 0 0 4 0 0

0 4 0 4 0 0 0 0

**Output:**

3 3 3 4

3 3 4 5

4 5 6 7

9 7 6 6

4 4 4 4 4 4 4 4

4 4 4 4 4 4 4 4

4 4 4 4 4 4 4 4

4 4 4 4 4 4 4 4

4 4 4 4 4 4 4 4

In the first test case  $S = 81$ , so the score is equal to approximately 6.33985. In the second test case  $S = 67$ , so the score is equal to approximately 6.06609. The overall score is thus the average of these values, or approximately 6.20297.

## Test Case Generation

Every official input file contains exactly 10 test cases. In every test case  $M$  and  $N$  are chosen randomly and uniformly between 10 and 100, inclusive, and the values of all cells are set to zero. Then, a random integer  $K$  between 1 and 10, inclusive, is chosen. Then the following operation is executed exactly  $N \cdot M \cdot K / 100$  (rounded down) times: a random cell with zero value is chosen among cells which don't have any side-by-side neighboring cells with non-zero value, and the chosen cell is filled with a random integer between 1 and 50, inclusive.

[Home](#) » [Compete](#) » [October Long Contest 2011](#) » [Lucky Palin](#)

## Lucky Palin Problem Code: LUCKPAL

Chef Palin, as his name suggests, is always very interested in palindromic strings. Recently, he made a pretty interesting discovery on palindromes and that made him feel really Lucky. He came across something known as Lucky Palindromes. He defines a string as being a lucky palindrome if it is a palindrome containing the string "lucky" as a substring. As always, now he wants to turn every string he comes across into a lucky palindrome. Being a chef, he is a man of patience and creativity, so he knows the operation of replacing any character of the string with any other character very well and he can perform this action infinitely many times. He wants you to write a program that can help him convert a given string to a lucky palindrome using the minimum number of operations and if several such lucky palindromes are possible, then output the lexicographically smallest one.

### Input

The first line contains a single integer  $T \leq 100$  the number of testcases. The following  $T$  lines each contain a string of length  $\leq 1000$  and only containing characters 'a'-'z'.

### Output

For each line of testcase, your program should output on a single line, the required lucky palindrome along with the minimum number of operations, both separated by a single space. If there is no lucky palindrome possible, then just output "unlucky" in a single line.

### Example:

#### Input

```
3
laubcdkey
luckycodechef
aaaaaaaa
```

### Output

```
luckykul 8
luckyccykul 6
unlucky
```

[Home](#) » [Compete](#) » [October Long Contest 2011](#) » Repeated String

## Repeated String Problem Code: REPSTR

Given a string S (containing at most  $10^5$  lowercase English letters). You are requested to find out from continuous substrings a string having length from L to H, which appears the most times; if there are more than one answer, find the most length.

### Input

There are several test cases (fifteen at most), each formed as follows:

- The first line contains two positive integers L, H.
- The second line contains the string S.

The input is ended with  $L = H = 0$ .

### Output

For each test case, output on a line two integers which are the number of times appearing and the length of the found string, respectively.

### Example

#### Input:

```
3 5
```

```
aabcbcbca
```

```
3 5
```

```
baaaababababbababbab
```

```
1 4
```

```
abcd
```

0 0

**Output:**

2 4

6 3

1 4

## Explanation

Case #1: **bcbc** occurs twice - at position 3 and position 5 (occurrences may overlap).

Case #2: **bab** occurs 6 times.

Case #3: **abcd** occurs 1 time.

[Home](#) » [Compete](#) » [October Long Contest 2011](#) » The Baking Business

## The Baking Business Problem Code: BAKE

The chef has turned into an entrepreneur running a bakery business. He is selling products like Biscuit, Bread etc which are packaged in various sizes. He is selling the products in the various cities of different provinces which are further sub-categorized into various regions. He maintains the data about the sex and age of all his customers along with the time of purchase. For running his business successfully, he requires a business intelligence tool to get analysis reports of sales aggregates based on various parameters so that he can run his business efficiently.

### Input

There are 10 different products which will be packaged in maximum 3 different sizes.

There are 10 different provinces and each province can have upto 20 cities which can be further divided in at maximum 5 different regions.

The first line of the input contains the total number S ( $\leq 100000$ ) of input lines. Each input line either begins with character 'I' (which stands for sale input)or 'Q' (which stands for query). The format of sale input is

**I product\_id[size\_id] province[.city\_id[.region\_id]] M/F age units\_sold**

This line contains details of individual sale vis-vis product details, location detail, customer sex and age (from 1 to 90, including the limit) separated by spaces in the following format. The product details e.g 6.2 which means the product with code 6 and size id 2. Next will be location like 8.18.4 which means the 8th province, in the 4th region of the city with code 18.

Then the sex of the customer either 'M' or 'F' and age from 1 to 90, including the limits. Note that all the codes/id's will begin from 0. Note that parts in the square brackets in the given format are optional because some specific information about the sale is missing due to practical reasons. The units\_sold will be less than or equal to 100.

Similarly, the format for a query line is

**Q product\_id[.size\_id] province[.city\_id[.region\_id]] M/F start\_age[-end\_age]**

This queries the total number of product units sold which falls under that specification. In case the optional parts are missing, then it means that the query is asking for the total units sold falling under all the sub-specifications under that. For the age parameter, if the end\_age is missing then the query is only for the unit sold to customer of age equal to start\_age, otherwise for all the customers falling in the range. A special product\_id is -1 which means all the products and similarly -1 for province means all the provinces. This id will never be qualified with sub-specifications.

## Output

For each query line specification output the total units sold which falls under that on a separate line. The answer should be based on the input given before the line and not on the input which will appear in the subsequent lines.

## Example

Input:

6

I 1.1 2.1 F 70 22

I 1 4.1 F 37 73

Q 1.1 2 F 10-90

I 2 4.2 M 63 24

Q 1 -1 F 30-70

Q 2 4.1 F 37

Output:

22

95

0

### Sample I/O Explanation

For the first query at the 4th line of input, only the first input is relevant as the the second input has location specification as 4.1 whereas in the query it is 2. Location specification 2 includes all the cities in the province 2 such as 2.1 whic is the case with the first input. Hence the solution is 22.

For the query specification on the second last line, it only includes the first 2 inputs as the third input mismatches with the query w.r.t the location and sex specification.

For the third query, it does not match with any input, hence the answer is 0.

[Home](#) » [Compete](#) » [October Long Contest 2011](#) » Sine Partition Function

## Sine Partition Function Problem Code: PARSIN

Chef Ciel is participating an arithmetic contest now. Why? Because of the top prize for the contest, a limited edition toaster oven.

She must calculate the values  $f(M, N, X)$  of function named *sine partition function* to be the first place. The sine partition function  $f(M, N, X)$  is defined by

$$\sum_{\substack{k_1 + k_2 + \dots + k_M = N \\ k_1, \dots, k_M \text{ are nonnegative integers}}} \sin(k_1 X) \sin(k_2 X) \dots \sin(k_M X)$$

Examples are following:

$$f(1, 3, X) = \sin(3X)$$

$$f(2, 3, X) = \sin(0) \sin(3X) + \sin(X) \sin(2X) + \sin(2X) \sin(X) + \sin(3X) \sin(0) = 2 \sin(X) \sin(2X)$$

$$f(3, 1, X) = \sin(0) \sin(0) \sin(X) + \sin(0) \sin(X) \sin(0) + \sin(X) \sin(0) \sin(0) = 0$$

Ciel is a great chef, however she is not good at arithmetic. For given  $M$ ,  $N$  and  $X$ , your work is calculating the value of  $f(M, N, X)$ .

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case has 2 integers  $M, N$  and a real number  $X$ .  $X$  has at most two digits after the decimal point.

## Output

For each test case, print the value of  $f(M, N, X)$ . This value must have an absolute or relative error no more than  $10^{-1}$ . You can safely assume the magnitude of the answer is at most  $10^{300}$ .

## Constraints

$1 \leq T \leq 10$   
 $1 \leq M \leq 30$   
 $1 \leq N \leq 1000000000$  ( $10^9$ )  
 $0 \leq X \leq 6.28 < 2\pi$

## Sample Input

5

1 3 1.57

2 3 0

3 1 2.12

1 1 5

1 3 0 0.3

## Sample Output

-0.99999714638

0

0

-0.959

0.412

COOK15

[October Cook-off](#)

23 Oct 2011  
21:30:00

2 hours 45 minutes

307

[Home](#) » [Compete](#) » [October Cook-off](#) » Chef Travel Routes

# Chef Travel Routes Problem Code: TRAVELER

Chef likes to travel very much. He plans some travel routes and wants to know their lengths. He hired you to make these calculations. But be careful, some of the routes are incorrect. There may be some misspelling in city names or there will be no road between some two consecutive cities in the route. Also note that Chef hates to visit the same city twice during his travel. Even the last city should differ from the first. Two consecutive cities in the route should also be different. So you need to check these conditions for the given routes too.

You will be given the list of all cities and all roads between them with their lengths. All roads are one-way. Also you will be given the list of all travel routes that Chef plans. For each route you should check whether it is correct and find its length in this case.

## Input

The first line contains positive integer **N**, the number of cities. The second line contains space separated list of **N** strings, city names. All city names are distinct.

The third line contains non-negative integer **M**, the number of available roads. Each of the next **M** lines describes one road and contains names **C<sub>1</sub>** and **C<sub>2</sub>** of two cities followed by the positive integer **D**, the length of the one-way road that connects **C<sub>1</sub>** with **C<sub>2</sub>**. It is guaranteed that **C<sub>1</sub>** and **C<sub>2</sub>** will be correct names of two different cities from the list of **N** cities given in the second line of the input file. For each pair of different cities there is at most one road in each direction and each road will be described exactly once in the input file.

Next line contains positive integer **T**, the number of travel routes planned by the Chef. Each of the next **T** lines contains positive integer **K** followed by **K** strings, names of cities of the current route. Cities are given in order in which Chef will visit them during his travel.

All strings in the input file composed only of lowercase, uppercase letters of the English alphabet and hyphens. Each string is non-empty and has length at most 20. If some line of the input file contains more than one element than consecutive elements of this line are separated by exactly one space. Each line of the input file has no leading or trailing spaces.

## Output

For each travel route from the input file output a single line containing word **ERROR** if the route is incorrect and its length otherwise.

## Constraints

**1 <= N <= 50**  
**0 <= M <= N \* (N - 1)**  
**1 <= D <= 20000**

**1 <= T <= 50**  
**1 <= K <= 50**  
**1 <= length of each string <= 20**

## Example

**Input:**

5

Donetsk Kiev New-York Miami Hollywood

9

Donetsk Kiev 560

Kiev New-York 7507

New-York Miami 1764

Miami Hollywood 28

Hollywood Miami 30

Miami New-York 1764

Kiev Donetsk 550

Hollywood New-York 1736

New-York Hollywood 1738

13

5 Donetsk Kiev New-York Miami Hollywood

5 Hollywood Miami New-York Kiev Donetsk

3 Donetsk Kiev Donetsk

2 Kyiv New-York

3 New-York Hollywood Miami

2 New-York Miami

3 Hollywood New-York Miami

4 Donetsk Kiev Miami Hollywood

2 Donetsk Hollywood

1 Donetsk

2 Mumbai Deli

6 Donetsk Kiev New-York Miami Hollywood New-York

2 Miami Miami

**Output:**

9859

ERROR

ERROR

ERROR

1768

1764

3500

ERROR

ERROR

0

ERROR

ERROR

ERROR

**Explanation**

The 2<sup>nd</sup> route is incorrect since there is no road from **New-York** to **Kiev**. Note however that inverse road from **Kiev** to **New-York** exists.

The 3<sup>rd</sup> route is incorrect since the first city coincides with the last one.

The 4<sup>th</sup> route is incorrect since there is no city with name **Kyiv** (Probably Chef means **Kiev** but he misspells this word).

The 8<sup>th</sup> route is incorrect since there is no road from **Miami** to **Kiev**.

The 9<sup>th</sup> route is incorrect since there is no road from **Donetsk** to **Hollywood**.

The 10<sup>th</sup> route is correct. Note that a route composed of exactly one city is always correct provided that city name is written correctly.

The 11<sup>th</sup> route is incorrect since there is no cities with names **Mumbai** and **Deli**. (Probably Chef is not so good in geography :))

The 12<sup>th</sup> route is incorrect since city **New-York** is visited twice.

Finally the 13<sup>th</sup> route is incorrect since we have equal consecutive cities.

[Home](#) » [Compete](#) » [October Cook-off](#) » Generalized Arithmetic Progression Free Sequence

## Generalized Arithmetic Progression Free Sequence

Problem Code: GENARSEQ

For given two positive integers **a** and **b** let's construct the sequence  $x[1], x[2], \dots$  by the following rules. For **k=1** we put  $x[1]=1$ . Now let **k >= 2**. Then  $x[k]$  is the least positive integer such that the following two conditions hold

- $x[k] > x[k - 1]$ ;
- for all  $i, j < k$  we have  $x[k]$  is not equal to  $a * x[i] - b * x[j]$ .

You need to find the first **n** terms of this sequence.

**Remark.** Let **a=2** and **b=1**. Then we see that sequence  $x[1], x[2], \dots$  does not contain any arithmetic progression of length three as its subsequence. This justifies the problem title.

### Input

The first line contains a single integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains three integers **a**, **b** and **n**.

### Output

For each test case, output a single line containing the numbers  $x[1], x[2], \dots, x[n]$  generated by the rules given in the problem statement and numbers **a**, **b**. Separate any two consecutive numbers in a line by a single space.

### Constraints

$1 \leq T \leq 50$   
 $1 \leq a, b \leq 50$   
 $1 \leq n \leq 1000$

## Example

**Input:**

3

2 1 10

1 1 5

4 2 4

**Output:**

1 2 4 5 10 11 13 14 28 29

1 2 3 4 5

1 3 4 5

[Home](#) » [Compete](#) » [October Cook-off](#) » First non-Palindrome

## First non-Palindrome Problem Code: NONPALIN

For the given string  $S$  of length  $N$  you need to find for each  $L$  from 1 to  $N$  the first non-palindrome substring of  $S$  that has length  $L$ . That is, for each  $L$  from 1 to  $N$  you need to find the smallest positive integer  $K \leq N - L + 1$  such that the string  $S[K, K + L - 1]$  is not a palindrome. Denote this number by  $K(L)$ . Here  $S[i, j]$  stands for the substring of  $S$  composed of its characters in positions  $i, i + 1, \dots, j$ . Characters of  $S$  are numbered from 1 to  $N$ . If for some  $L$  there is no such  $K$  set  $K(L) = 0$ . After you find all numbers  $K(1), K(2), \dots, K(N)$  output the following sum

$$100007^{N-1} * K(1) + 100007^{N-2} * K(2) + \dots + 100007 * K(N-1) + K(N)$$

modulo  $2^{64}$ .

**Remark.** The string is called a palindrome if it coincides with its reverse. So **abacaba** and **abba** are palindromes but **codechef** and **abbc** are not.

## Input

The first line contains a single integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains a non-empty string composed only of lowercase letters of the English alphabet.

## Output

For each test case, output a single line containing the corresponding sum constructed by numbers **K(1), K(2), ..., K(N)** as mentioned in the problem statement modulo **2<sup>64</sup>**.

## Constraints

**1 <= T <= 20**

**1 <= length of S <= 100000**

## Example

**Input:**

4

abacaba

abba

codechef

aaaa

**Output:**

12789123637940213592

10001500056

18134494634143926857

0

## Explanation

Actual values of **K(1)**, **K(2)**, ..., **K(N)** in the tests are:

abacaba: {0, 1, 2, 1, 1, 1, 0}

abba: {0, 1, 1, 0}

codechef: {0, 1, 1, 1, 1, 1, 1, 1}

aaaa: {0, 0, 0, 0}

[Home](#) » [Compete](#) » [October Cook-off](#) » Permute Digits

## Permute Digits Problem Code: PERMDIG

You are given three positive integers **A**, **B** and **C** written in base **X**  $\leq 10$  without leading zeros. In how many ways you can permute the digits of **A** and permute the digits of **B** such that their sum will be equal to **C** (in base **X** of course)? Leading zeros after permutation are allowed. So for example for **A=101** written in some base all possible ways to permute its digits are: **011, 101, 110**.

**Remark.** For positive integer **A** and base **X**  $\geq 2$  the digits of **A** in base **X** are uniquely determined by the equality  $A = A_k * X^k + A_{k-1} * X^{k-1} + \dots + A_1 * X + A_0$  and inequalities **0**  $\leq A_0, A_1, \dots, A_k < X and **A_k > 0**. Then **A** is written as **0...0A<sub>k</sub>A<sub>k-1</sub>...A<sub>1</sub>A<sub>0</sub>** in base **X**. Here an arbitrary non-negative number of leading zeros is allowed. If there are no leading zeros we say that **A** is written in canonical form.$

### Input

The first line contains a single integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains four space separated positive integers **X, A, B** and **C**, where **A, B, C** is written in base **X** without leading zeros (that is in canonical form).

### Output

For each test case, output a single line containing the number of possible permutations of digits of **A** and **B** such that their sum is equal to **C**.

### Constraints

**1**  $\leq T \leq 10$

**2**  $\leq X \leq 10$

**1**  $\leq \text{len}(A), \text{len}(B), \text{len}(C) \leq 80/X$

where **len(N)** stands for the number of digits of number **N** when it is written in base **X** in canonical form.

### Example

**Input:**

5

2 10 10 11

3 2 2 11

10 101 12 23

10 10 100 1000

10 43716 70251864 71130699

**Output:**

2

1

1

0

4

**Explanation**

In the first case the appropriate ways are  $01+10=11$  and  $10+01=11$ .  
 In the third case the only appropriate way is  $011+12=23$ .

[Home](#) » [Compete](#) » [October Cook-off](#) » Lame Queen Game

**Lame Queen Game** Problem Code: LAMQUGAM

Code and Chef play a game on the set of all points with positive integer coordinates of the usual coordinate plane. Initially the lame queen stands in the point with coordinates  $(x_0, y_0)$  or shortly: in the point  $(x_0, y_0)$ . Players alternate moves. **Code moves first.** During each move the player can move the queen from the point  $(x, y)$  to the point  $(x - a, y)$ ,  $(x, y - a)$  or  $(x - a, y - a)$  for some positive integer  $a$ , provided that the new point has positive coordinates. But since the queen is lame she possibly can't make all diagonal moves. Namely she can move from the point  $(x, y)$  to the point  $(x - a, y - a)$  only if  $a$  is divisible by  $d$ , where  $d$  is some positive integer that is called the queen lameness. So if  $d = 2$  the queen can move diagonally from the point  $(x, y)$  only to the points  $(x - 2, y - 2)$ ,  $(x - 4, y - 4)$ ,  $(x - 6, y - 6)$  and so on. However if  $d = 1$  then the queen becomes the usual chess

queen. **The first player who can't make a valid move loses the game.** Both players play optimally in their respective moves.

After playing this game for some time Chef realizes that he loses almost always. Now he wants to know what is the probability that he wins if initial point  $(x_0, y_0)$  is chosen randomly from the rectangle  $x_1 \leq x_0 \leq x_2, y_1 \leq y_0 \leq y_2$ . But I should warn that he wants to know this for a large number of rectangles.

## Input

The first line contains two space separated positive integers  $d$  and  $T$ , the queen lameness and the number of rectangles for which Chef wants to know the answer. Each of the next  $T$  lines contains four space separated positive integers  $x_1, y_1, x_2, y_2$  the coordinates of rectangle corners.

## Output

For each rectangle from the input file, output a single line containing the probability in the form of an irreducible fraction that Chef wins if the initial cell is chosen randomly from this rectangle.

## Constraints

$1 \leq d \leq 20$   
 $1 \leq T \leq 100000$   
 $1 \leq x_1 \leq x_2 \leq 200000$   
 $1 \leq y_1 \leq y_2 \leq 200000$

## Example

Input:

1 4

1 1 4 4

2 4 5 8

3 7 10 10

5 8 5 8

Output:

3/16

1/10

1/32

1/1

## Explanation

For the first rectangle Chef wins in three cases out of 16. Namely when the starting point is **(1,1)**, **(2,3)** or **(3,2)**.

The fourth rectangle is lucky. It contains only one point and this one is winning for Chef.

NOV11	<a href="#">November Long Contest 2011</a>	01 Nov 2011 15:00:00	10 days	448
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » Chefs new Pet

## Chefs new Pet Problem Code: HAREJUMP

The chef got a new rabbit and he is going to train him so that he can perform for him whenever he needs entertainment. The chef teaches  $k$  types of jumps to the rabbit. Each jump has definite length  $L$  units. The rabbit does not have any brains he will use any type of jump he feels like at any point of time. He is placed on a very long mat and starts at 0 unit. The chef wants to know in how many ways he can perform his jumps and cover exactly  $N$  units of distance.

### Input

The first line contains the number of the test cases  $T$  ( $<=100$ ). Each test case consists of 2 lines. The first line defines the distance  $N$  ( $1 <= N <= 10^{18}$ ) which tells us the number of units the chef wants the rabbit to cover. The next line contains  $k$  the number of jumps which are taught to the rabbit,  $k$  ( $k <= 15$ ) integers follow in the same line each denoting distinct distance  $L$  ( $L <= 15$ ) units which can be jumped by the rabbit.

### Output

Print  $T$  integers each denoting the number of ways the rabbit can perform jumps of  $N$  units of distance. Print the remainder obtained on dividing the answer by 1000000007 if the answer is greater than or equal to 1000000007.

### Example

**Input:**

```

10
1 1
13
3 1 2 8
15
5 1 2 3 4 5

```

**Output:**

```

1
415
13624

```

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » Stepping Average

## Stepping Average Problem Code: STEPAVG

Consider the following weird way of finding the average of **N** numbers. Take any two of the numbers and replace them with their average; repeat this operation exactly **N-1** times. The only remaining number is called the *stepping average* of the initial set. Note that a set may obviously have different stepping averages depending on our choice for every operation.

Your task is, given **N** integers, find a way of performing the operations such that the resulting stepping average is as close to the given integer **K** as possible. Note that this is a challenge problem: you don't have to find the best possible solution, but the better is your solution the more points you get.

### Input

The first line of the input contains an integer **T** -- the number of test cases (no more than 10). Each test case consists of two lines -- the first of them contains two positive integers **N** and **K** ( $N \leq 1000$ ,  $K \leq 10^9$ ), the second contains **N** space-separated positive integers **A<sub>1</sub>..A<sub>N</sub>** ( $A_i \leq 10^9$ ).

## Output

The output for each test case should contain exactly  $N-1$  pairs of integers.  $i$ -th pair (1-based)  $x$   $y$  means that numbers  $A_x$  and  $A_y$  are chosen at the corresponding operation, their average is written to  $A_{N+i}$ , and  $A_x$  and  $A_y$  can't be used any more. See example explanation for more clarity.

## Scoring

Your score for each test case is just the absolute difference between  $K$  and your remaining number. Your final score is the average of all test case scores. Your goal is to minimize the final score.

Note that in order for your submission not to be judged as 'Wrong Answer' the following conditions should be satisfied:

- your output should contain exactly  $N-1$  pairs of positive integers separated by at least one space for each test case and nothing else;
- your output should contain integers strictly less than  $N+i$  in the  $i$ -th pair (1-based) for each test case;
- all integers in your output for each test case should be different.

## Example

### Input:

```
1
5 5
9 1 6 3 4
```

### Output:

```
5 2
4 6
3 1
7 8
```

### Explanation:

We have 5 integers here: 9 1 6 3 4. The first operation is 5 2, so the 6th number becomes  $(4+1)/2 = 2.5$ , and the 2nd and the 5th numbers are removed, so we have 9 x 6 3 x 2.5 now (here x means a removed number). Then, 4 6 means that the 7th number becomes  $(3+2.5)/2 = 2.75$ , and the 4th and the 6th numbers are removed, resulting in 9 x 6 x x 2.75. Then, after 3 1 we get x x x x x 2.75 7.5, and after 7 8 we get x x x x x x x 5.125. The only number left after the operations is 5.125, so your score for the test case is 0.125.

## Test Case Generation

Every official input file contains exactly 10 test cases. In every test case **N** is equal to 1000, **K** is chosen randomly and uniformly between 1 and  $10^9$ , and all **A<sub>i</sub>** are chosen randomly and uniformly between 1 and  $10^9$  as well.

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » New Restaurant

# New Restaurant Problem Code: NEWREST

Chef Dengklek will open a new restaurant in the city. The restaurant will be open for **N** days. He can cook **M** different types of dish. He would like to cook a single dish every day, such that for the entire **N** days, he only cook at most **K** distinct types of dishes.

In how many ways can he do that?

## Input

The first line contains a single integer **T**, the number of test cases. **T** test cases follow. Each test case consists of a single line consisting of three integers **N**, **M**, **K**.

## Output

For each test case, output a single line consisting the number of different ways he can cook for the entire **N** days, modulo 1000000007.

## Constraints

- $1 \leq T \leq 100$
- $1 \leq N \leq 1000$
- $1 \leq M \leq 1000000$
- $1 \leq K \leq 1000$

## Example

### Input:

4

1 1 1

2 2 2

4 3 2

5 7 3

**Output:**

1

4

45

5887

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » The Postal Service

## The Postal Service Problem Code: POSTAL

The Chef is on vacation in the beautiful country of Chefville. The island is a narrow strip of land running from north to south. A single road from north to south connects all the villages in the country. There, he observed, the postal services works in a very curious way.

There is a postman for every village. In the morning, say postman A, starts traveling to the North, disbursing all the mails on the way. When he meets another postman coming from the opposite direction on the way, say B, A hands over all the mails to B which are to be delivered to addresses that lie in the North and vice versa for B. Then both turn back and start traveling in the opposite directions. So, now A travels to the South and when he meets another postman on the way, the same thing happens again. This keep on till the duty hours of the day finishes and they don't loose any time in the swapping and turning back. All postmen travel at the same speed of 1 m/s always and they arbitrarily choose their direction to start with in the morning.

Now, at any given time of the day, the Chef wants to know the position of any given postman and the number of times he has met another postman till that time (including this moment of time).

### Input

The road can be considered to stretch infinitely in both the directions for this problem. The first line contains the total number of test cases ( $\leq 10$ ). Each of the case begins with  $N$  ( $0 < N \leq 500$ ) on a separate line, the total number of villages in Chefville. The next line contain  $N$  distinct integers separated by space which are the positions (between 0 and 1000000000, in meters including those, 0 being the position of a northern point on the strip) in increasing order of the  $N$  villages on the road. The next line the initial directions of the corresponding postmen of the villages as  $N$  integers which are either 0 or 1. 0 means North and 1 South. Then comes the total number of queries  $Q$  ( $\leq 1000$ ) for this test case

on separate line. The next  $Q$  lines contain two integers, first the postman number  $I$  ( $0 \leq I < N$ ) and then the time  $T$  ( $\leq 1000000000$  in seconds).

## Output

For every query of a test case, you have to output the position of the postman number  $I$  at time  $T$  and the number times he has met another postman till then, on the same line separated by a space.

## Example

Input:

```
1
5
1 24 28 34 94
0 1 0 0 0
2
2 11
0 9
```

Output:

```
23 2
-8 0
```

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » [Lucky Days](#)

## Lucky Days Problem Code: LUCKYDAY

Ciel Ciel defined a sequence  $S$  as follows:

$S[1] = A$   
 $S[2] = B$   
 $S[i] = (X * S[i-1] + Y * S[i-2] + Z) \bmod P$ , for  $i \geq 3$

Ciel considers  $C$  is a lucky number, and the  $i$ -th day is a lucky day if and only if  $S[i] = C$ . Ciel's restaurant may have special events in a lucky day.

By the way, your work is calculating the numbers of lucky days in intervals. That is, for each Q intervals  $[L[i], R[i]]$ , you should calculate the number of the integers  $k$  which satisfy  $L[i] \leq k \leq R[i]$  and  $S[k] = C$ .

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line for each test case has 8 integers  $A, B, X, Y, Z, P, C$  and  $Q$ . The next  $Q$  lines have 2 integers  $L[i]$  and  $R[i]$ .

## Output

For each interval, print the number of lucky days in the interval.

## Constraints

$1 \leq T \leq 2$   
 $2 \leq P \leq 10007$   
 $P$  is a prime.  
 $0 \leq A, B, X, Y, Z, C < P$   
 $1 \leq Q \leq 20000 (2 \cdot 10^4)$   
 $1 \leq L[i] \leq R[i] \leq 1000000000000000000 (10^{18})$

## Sample Input

```

2
1 1 1 0 2 0 6
1 1
2 2
3 3
4 4
5 5
6 6
1 2 4 5 3 1 7 4 3
5 8
5 58

```

58 5858

## Sample Output

0

0

1

0

0

1

0

4

362

## Output details

In the first case:

$S[1] = A = 1$

$S[2] = B = 1$

$S[3] = (S[2] + S[1]) \bmod 2 = (1 + 1) \bmod 2 = 0$

$S[4] = (S[3] + S[2]) \bmod 2 = (0 + 1) \bmod 2 = 1$

$S[5] = (S[4] + S[3]) \bmod 2 = (1 + 0) \bmod 2 = 1$

$S[6] = (S[5] + S[4]) \bmod 2 = (1 + 1) \bmod 2 = 0$

$S[7] = (S[6] + S[5]) \bmod 2 = (0 + 1) \bmod 2 = 1$

[Home](#) » [Compete](#) » [November Long Contest 2011](#) » Colored Domino Tilings and Cuts

# Colored Domino Tilings and Cuts

[Problem](#)

**Code:** DOMNOCUT

Consider rectangular grid that composed of **N** rows and **M** columns. **Colored domino tiling** of the grid is some way to put lowercase English letter in each cell of the grid provided that each cell has exactly one neighbor with the same letter as in this cell. Two cells of the grid are neighbors if they share a common side. Each letter represents some color. **Cut** of colored domino tiling is vertical or horizontal line that divides the grid in two colored domino tilings. In other words this line should not pass between two adjacent cells with the same color.

For given **N** and **M** you should find the colored domino tiling that has the minimal possible

number of cuts. If there are several solutions you should find among them the tiling that uses the minimal possible number of colors in it. If your tiling requires **K** colors then use first **K** lowercase letters of English alphabet as colors. If there are still several solutions then find any such tiling.

## Input

The first line contains a positive integer **T**, the number of test cases. In the following lines, **T** test cases follow. Every test case is a single line that contains two space separated positive integers, **N** and **M**, the sizes of the grid.

## Output

For each test case output in the first line the word "**IMPOSSIBLE**" without quotes if there are no tilings of the grid with **N** rows and **M** columns. Otherwise output in the first line two space separated integers, the minimal number of cuts and minimal number of colors in required tiling and then in the next **N** lines output the tiling itself. Each of these **N** lines must be composed of exactly **M** lowercase English letters.

## Constraints

$1 \leq T \leq 3000$   
 $1 \leq N, M \leq 500$   
sum of  $N \times M$  in the file  $\leq 2,000,000$

## Sample input

5

1 1

2 2

2 3

4 3

5 5

## Sample output

IMPOSSIBLE

1 2

ab

ab

1 3

aab

ccb

1 3

aab

ccb

baa

bcc

IMPOSSIBLE

COOK16

[November Cook-Off](#)

20 Nov 2011

21:30:00

2 hours 30 minutes

305

[Home](#) » [Compete](#) » [November Cook-Off](#) » Subanagrams

## Subanagrams Problem Code: SUBANAGR

Let's start from some definitions.

Strings **A** and **B** are called *anagrams* if it's possible to rearrange the letters of string **A** using all the original letters exactly once and achieve string **B**; in other words **A** and **B** are permutations of each other. For example, *remote* and *meteor* are anagrams, *race* and *race* are anagrams as well, while *seat* and *tease* aren't anagrams as *tease* contains an extra 'e'.

String **A** is called a *subsequence* of string **B** if **A** can be obtained from **B** by removing some (possibly none) characters. For example, *cat* is a subsequence of *scratch*, *rage* is a subsequence of *rage*, and *tame* is not a subsequence of *steam*.

String **A** is *lexicographically smaller* than string **B** of the same length if at the first position where **A** and **B** differ **A** contains a letter which is earlier in the alphabet than the corresponding letter in **B**.

Recently, Ann received a set of strings consisting of small Latin letters a..z. 'What can I do with them?' -- she asked herself. 'What if I try to find the longest string which is a subsequence of every string from the set?'. Ann spent a lot of time trying to solve the

problem... but all her attempts happened to be unsuccessful. She then decided to allow the sought string to be an anagram of some subsequence of every string from the set. This problem seemed to be easier to Ann, but she was too tired to solve it, so Ann asked for your help.

So your task is, given a set of strings, to find the longest non-empty string which satisfies Ann. Moreover, if there are many such strings, choose the lexicographically smallest one.

## Input

The first line of the input file contains one integer **N** -- the number of strings in the set ( $1 \leq N \leq 100$ ). Each of the next **N** lines contains a non-empty string consisting only of small Latin letters *a..z* representing a string from the set. None of the strings contain more than 100 letters.

## Output

Output the longest non-empty string satisfying Ann. If there are several such strings, output the lexicographically smallest one. If there are no such strings, output '**no such string**' (quotes for clarity).

## Example

### Input:

3

hope

elephant

path

### Output:

hp

### Input:

2

wall

step

**Output:**

no such string

**Explanation:**

In the first test case the longest string appears to be two characters long. String 'hp' satisfies the requirements as it's an anagram of 'hp' which is a subsequence of 'hope' and an anagram of 'ph' which is a subsequence of both 'elephant' and 'path'. Note that string 'ph' also satisfies the requirements, but 'hp' is lexicographically smaller.

In the second test case there is no such string.

[Home](#) » [Compete](#) » [November Cook-Off](#) » Moving Coins

## Moving Coins Problem Code: MVCOIN

There is a line with 1000 cells numbered from 1 to 1000 from left to right and **N** coins placed on it. Coin **i** is placed at cell **X<sub>i</sub>**, and no two coins are placed at the same cell.

Bob would like to move the coins to the **N** leftmost cells of the line. To do this, he is allowed to take a coin from any cell **T** and move it to cell **T-j**, where **j** is an integer between 1 and **K**, inclusive. This action is possible only if:

- cell **T-j** actually exists and doesn't contain a coin;
- each of the cells **T-j+1**, ..., **T-1** contains a coin.

One coin movement takes exactly one second. Find the smallest time in which Bob can achieve his goal.

### Input

The first line of the input file contains one integer **T** -- the number of test cases (no more than 10). Then **T** test cases follow, and every test case is described by two lines: the first of them contains two integers **N** and **K** ( $1 \leq N, K \leq 1000$ ), the second of them contains **N** integers **X<sub>1</sub>**, ..., **X<sub>N</sub>** in strictly increasing order ( $1 \leq X_i \leq 1000$ ).

### Output

For each test case output one line containing the requested minimal time for Bob to put all the coins to the left side of the line.

## Example

### Input:

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

### Output:

```
5
0
```

### Explanation:

In the first example Bob can move the coin from cell 7 consequently to cells 6, 5, 3 and 1, then move the coin from cell 4 to cell 3. In the second example there is nothing to move.

[Home](#) » [Compete](#) » [November Cook-Off](#) » Colorful Chain

## Colorful Chain Problem Code: COLCHAIN

A chain is a series of connected links. David has been presented a chain for his birthday consisting of **N** links and would like to color each of the links into one of **M** colors. He would like the coloring to be pretty uniform, so that no color is used too often or too seldom.

Finally, David came with the following restriction: he wants to color the links in such a way that every color is used at least once in every **M+1** consecutive links.

Two colorings are considered different if there exists at least one **i** between 1 and **N** such that link **i** is colored differently in these colorings. (The links are numbered, and the chain can not be rotated.) How many ways are there for David to color the chain?

### Input

The first line of the input file contains one integer **T** -- the number of test cases (no more than 10). Each of the next **T** lines contains two integers **N** and **M** -- the length of the chain presented to David and the number of colors ( $1 \leq M < N \leq 10^5$ ).

## Output

For each test case output the number of ways for David to color the chain modulo 1 000 000 007 ( $10^9 + 7$ ).

## Example

Input:

3

2 1

4 2

6 3

Output:

1

10

132

[Home](#) » [Compete](#) » [November Cook-Off](#) » Graph on a Plane

## Graph on a Plane Problem Code: GRAPLANE

Suppose you are given a graph as a set of pairs of numbers representing edges. What is the first thing you usually do with such a graph? Draw it, of course!

Today Caroline has got an undirected graph with **N** vertices numbered from 1 to **N** and **M** edges without loops or multiple edges between two vertices, and she would like to draw it on a Cartesian plane. She wants the edges to be straight line segments, so the only problem is to assign a point on the plane to each vertex. Caroline doesn't want to place any vertices on the coordinate axes -- so each vertex should be assigned to one of four quadrants; moreover, if we count the number of vertices in each quadrant and write down these four numbers, no two of these numbers should differ by more than 1 (so Caroline wants to distribute the vertices among quadrants as equally as possible). She would also like no two vertices to coincide.

Out of all drawings satisfying the given requirements, Caroline would like to choose a drawing with the minimal number of intersections between graph edges and coordinate axes. Note that if an edge intersects the origin (0;0), it's counted as one intersection, and if an edge intersects both axes at different points, it's counted as two intersections. Also note that if two edges intersect either of the axes in the same point, an intersection is counted for both edges.

Caroline doesn't care about the thing that the edges may intersect, overlap, contain vertices or even each other inside. She considers all the graphs satisfying all the requirements equally good.

It's easy to guess now what you are to do in this problem.

## Input

The first line of the input file contains an integer  $T$  -- the number of test cases (no more than 3). Each test case is described by a line containing two integers  $N$  and  $M$  ( $2 \leq N \leq 18$ ;  $N$  is even;  $1 \leq M \leq N*(N-1)/2$ ) followed by  $M$  lines, each containing two integers between 1 and  $N$ , inclusive, and describing an edge of the graph. There will be a blank line preceding each test case. It's guaranteed that no edge connects a vertex to itself, and also no two edges connect the same pair of vertices (remember that the graph is undirected, so  $A B$  and  $B A$  is essentially the same edge).

## Output

For each test case output exactly  $N+1$  lines. The first of them should contain the smallest possible number of intersection of graph edges with coordinate axes. The following  $N$  lines should contain a pair of integers  $x_i y_i$  corresponding to a point on the plane assigned to the  $i$ -th vertex of the graph ( $1 \leq |x|, |y| \leq 10000$ ). You may separate the answers with empty lines.

## Example

Input:

3

2 1

1 2

4 4

1 2

2 3

3 4

4 1

4 2

1 3

2 4

**Output:**

1

1 2

3 -4

4

5 6

7 -8

-9 10

-11 -12

2

2 3

7 -5

-2 -3

-7 5

[Home](#) » [Compete](#) » [November Cook-Off](#) » Buying Candies

## Buying Candies Problem Code: BUYING

Emily is coming back from her two-week vacation at the other continent. But when Emily arrived at the airport, she realized that she had got no presents for her **K** friends! She still had some time before her flight, so she went for a walk around the airport hoping to find something suitable.

Soon Emily found a candy store and decided to buy some of her favorite candies for her friends. The store offers **N** packs of these candies, where pack **i** contains **A<sub>i</sub>** candies.

Another reason Emily decided to buy candies is that she is fond of collecting empty candy packs from various parts of the world. That's why she decided to buy exactly **M** packs and present the friends with the candies and keep the packs for her collection. Emily would also like the total number of candies to be divisible by **K** so that an equal distribution of candies between friends is possible. Among all possible sets of packs, she would like to buy a set with the smallest possible total number of candies (money is the reason).

Your task is to help Emily, of course.

### Input

The first line of the input file contains an integer **T** -- the number of test cases (no more than 5). **T** test cases follow, and each test case consists of two lines. The first of them contains three integers **N**, **M** and **K** ( $1 \leq M \leq N \leq 50000$ ,  $1 \leq K \leq 20$ ). The second of them contains **N** integers **A<sub>i</sub>** ( $1 \leq A_i \leq 10^9$ ).

### Output

For each test case, output just one line containing the smallest possible total number of bought candies, or **-1** if it's impossible to buy exactly **M** packs so that the total number of candies is divisible by **K**.

### Example

Input:

2

5 3 5

1 2 3 4 5

6 3 4

9 5 1 7 3 7

**Output:**

10

-1

**Explanation:**

In the first test case, it's impossible to buy one candy per friend as three smallest packs contain 6 candies all together. Two candies per friend is possible though -- for example, if you buy packs with 1, 4 and 5 candies.

In the second test case, buying 3 packs implies buying an odd number of candies, which is impossible to distribute equally among 4 friends.

DEC11	<a href="#">December 2011 Long Contest</a>	01 Dec 2011 15:00:00	10 days	564
-------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Robot Movings

## Robot Movings Problem Code: MOVES

Given a square table sized  $N \times N$  ( $3 \leq N \leq 5,000$ ; rows and columns are indexed from 1) with a robot on it. The robot has a mission of moving from cell  $(1, 1)$  to cell  $(N, N)$  using only the directions "right" or "down". You are requested to find the number of different ways for the robot using exactly  $K$  turns (we define a "turn" as a right move followed immediately by a down move, or a down move followed immediately by a right move;  $0 < K < 2N-2$ ).

### Input

There are several test cases (5,000 at most), each consisting of a single line containing two positive integers  $N, K$ .

The input is ended with  $N = K = 0$ .

### Output

For each test case, output on a line an integer which is the result calculated. The number of ways may be very large, so compute the answer modulo 1,000,000,007.

## Example

Input:

4 2

4 3

5 3

0 0

Output:

4

8

18

Explanation for the first sample test case: 4 ways are **RRDDDR**, **RDDDRR**, **DRRRDD**, **DDRRRD** ('R' or 'D' represents a right or down move respectively).

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Birthday Gift

## Birthday Gift Problem Code: CHEFGIFT

It's our Chef's Birthday next week. His friends want to gift him something very nice and useful. They know how much Chef likes to go out on trips. So they decide to gift him a caravan i.e a [travel trailer](#). As is clear from the description, a caravan is towed behind a car.

All the friends have chipped in a few dollars and now they have  $D$  dollars in total to buy a caravan. But there's one more thing they need to worry about. Caravans are sold outside their city. They will have to travel through various toll booths to buy one and bring it back. And they will have to pay some amount from their total  $D$  when they cross these toll booths. Luckily, they will not have to pay any toll for their own car(well, being Chef's friend has a lot of perks). But they will have to pay the toll for the caravan. They don't want to be spending too much on their tolls. Needless to say, they can't skip the toll booths.

There are  $m$  different roads between the Chef's city and the place where the caravans are sold. Each road takes a different route. There are  $n$  toll booths on every road. But an interesting observation is that the toll for a caravan need **not** be same for every road **AND** it also need **not** be same at every toll booth on the same road.

The friends want to gift him the most expensive caravan(which they can afford) and want to spend as little on tolls as they can. So  $m$  of the friends decide to travel on each of the roads individually. And once they buy a caravan, any one of them can start towing it behind his car. Also, a friend can move the caravan to some other friend's car i.e. to some other road. This is done so as to minimize the total toll. But there's another thing to be taken care of here. The caravan is not easy to move. So, there are mechanics available on every road who can do this. And there's a fixed amount they will charge to move the caravan from one road to another. This amount depends on the source and destination road between which the caravan is moved. But, there's a small restriction on the movement of the caravan. If the caravan has crossed the  $i$ th toll booth on a road, it can only be moved to a location before the  $(i+1)$ th booth on the other road. Initially, the caravan can be taken on any road.

Chef's friends will not want to buy an expensive caravan and find out later that they are out of money to pay for the toll. Nor do they want to end up buying a less expensive caravan and realize they could have bought a more expensive one. So they have collected all the details and want you to help them. Your task is to tell them the maximum amount they can spend on buying a caravan assuming they have  $D$  dollars initially and they have sufficient amount left to pay for the caravan toll.

### **Input:**

First line of input contains  $T$ , number of tests.

Each test begins with a line containing 3 space separated integers:  $D$ ,  $n$  and  $m$ : total amount of money, the number of toll booths on every road and the number of roads respectively.

Then follow  $m$  lines. Each line consists of  $n$  integers describing every road in order. The  $i$ th number in a line denotes the toll for the caravan at the  $i$ th toll booth on the respective road. Then follows an  $mxm$  matrix,  $C$ , where the  $j$ th number on the  $i$ th row i.e.  $C[i][j]$  signifies the cost of moving the caravan from the  $i$ th road to the  $j$ th road. Ofcourse,  $C[i][i]=0$ .

### **Output:**

Output a single integer: the maximum amount they will be able to spend on a caravan. If they don't have enough money to pay for the toll, print "-1" instead.

### **Constraints:**

$1 \leq T \leq 30$   
 $0 \leq D \leq 20000$   
 $1 \leq n, m \leq 100$   
 $0 \leq \text{toll} \leq 200$   
 $0 \leq C[i][j] \leq 100$

### **Input:**

3  
40 5 4  
10 9 8 7 6  
8 7 6 9 12

```

16 1 1 1 1
7 6 5 4 3
0 6 1 7 7
8 0 1 9 2
9 9 0 0
12 4 18 0
10 2 2
5 6
6 7
0 0
0 0
12 3 2
3 4 5
5 7 2
0 5
2 0
Output:
20
-1
0

```

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Ciel and Eggs

## Ciel and Eggs Problem Code: STREGGS

Chef Ciel has  $N$  strange eggs. The  $i$ -th egg is broken by tapping exactly  $A_i$  times. Ciel needs to break  $K$  eggs as soon as possible for cooking a rice omelet. However she has been put in an uncomfortable situation. Someone shuffled the eggs! Ciel knows the values  $A_i$ , however she doesn't know which egg is which. She'd like to minimize the worst-case number of taps. What is the minimal number?

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line for each test case has 2 integers  $N$  and  $K$ . Then next line has  $N$  integers  $A_1, A_2, \dots, A_N$ .

### Output

For each test case, print the minimal number of taps for the worst-case.

### Constraints

$1 \leq T \leq 10$   
 $1 \leq K \leq N \leq 500$   
 $1 \leq A_i \leq 1000000 (10^6)$

### Sample Input

3

2 1

5 8

2 1

5 58

3 2

1 2 3

## Sample Output

8

10

5

## Output details

In the first case, if a egg isn't broken after 5 taps, she should continue to tap the same egg.

In the second case, if a egg isn't broken after 5 taps, she should tap another egg 5 times.

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Spinning Wheels

## Spinning Wheels Problem Code: SPIN

Spinning Wheels is a game for one player. It is played on a square grid consisting of **N** rows and **N** columns. Each cell contains a *wheel*, which is colored into one of five colors -- red, orange, yellow, blue or violet (for convenience, we'll number the colors from 1 to 5, correspondingly). Each wheel also has two *needles*, and the needles are placed in one of the following positions (for convenience, we'll number the positions from 1 to 4, correspondingly):



Basically, position 1 means that the needles are directed up and right, position 2 means that they're directed down and right down, position 3 -- down and left, and position 4 -- up and left.

The goal of the game is to make all the wheels colored into the same color through a sequence of *moves*. A move consists of choosing a wheel and rotating it 90 degrees clockwise (for example, if the wheel was in position 1, it would change to position 2). After the rotation of some wheel, the state of other wheels (as well as the state of the wheel rotated first) may change.

More formally. Let's denote the color of the wheel chosen on this move by **C**. We'll say that two wheels are *matching* if they are situated in side-by-side neighboring cells and either needle of both wheels is directed to the other wheel. The process of changing the grid is separated into *iterations*. We will maintain set **S** consisting of the wheels rotated at the last iteration. Before the first iteration, set **S** contains only the starting wheel. The following algorithm is executed then repeatedly:

- If set **S** is empty, stop the process.
- Rotate each wheel in set **S** 90 degrees clockwise.
- Assign color **C** to each wheel in set **S**.
- Form set **Q** consisting of wheels which have at least one matching wheel in set **S**.
- Replace set **S** with set **Q** and clear set **Q**.

It's possible to prove that this process will definitely stop. Note that a wheel may get rotated more than once during the move.

Your task is to win the game. More precisely, for each grid configuration given in input you should either output a winning sequence of moves (no longer than 1000 moves) or output -1 meaning that you'd like to skip this test case (it's essentially the same as outputting a configuration with 1001 moves).

**Note.** This might help (though the rules are **not** exactly the same): <http://www.cesmes.fi/#boxSpin>.

## Input

The first line of the input file contains an integer **T** -- the number of test cases. Then **T** test cases follow. The first line of each test case contains an integer **N**. The following **N** lines contain **N** integers (between 1 and 4, inclusive) each, giving the initial positions of the wheels in the corresponding cells. The following **N** give the information about the starting colors (each color between 1 and 5, inclusive) in the same format.

**Note.** The constraints are listed in the 'Test Case Generation' part of the problem statement.

## Output

For each test case output -1 if you would like to skip this test case. Otherwise, output a line containing just one integer **K** ( $0 \leq K \leq 1000$ ) and then **K** lines each containing a pair of

integers **r** **c**, describing the corresponding move ( $1 \leq r, c \leq N$ ; **r** is the row ID, **c** is the column ID).

## Scoring

If at least one of the constraints in the 'Output' section is broken, you'll get 'Wrong Answer'. If at least one of your sequences of moves doesn't lead to winning in the corresponding test case (not all wheels are of the same color after performing all the moves), you'll get 'Wrong Answer'. Otherwise, for each test case you'll get a score of 1001 if you output **-1**, and a score of **K** if you output a valid sequence of moves. Your score for the input file is the sum of your scores for the particular test cases. Finally, there are 10 input files in this problem, and your overall score is the average of your scores on the input files. Of course, your goal is to minimize your overall score.

**Important! You must solve at least one test case in each file, otherwise you'll get 'Wrong Answer'.**

## Example

**Input:**

2

3

1 1 1

4 2 4

1 4 2

3 3 2

3 2 1

4 5 4

4

2 2 2 3

1 4 4 3

3 1 3 1

3 2 3 1

3 4 2 3

4 2 1 1

4 3 2 3

4 4 2 2

**Output:**

3

3 3

2 1

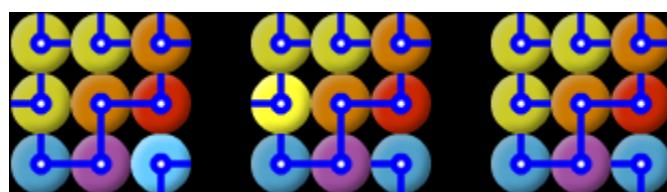
1 3

-1

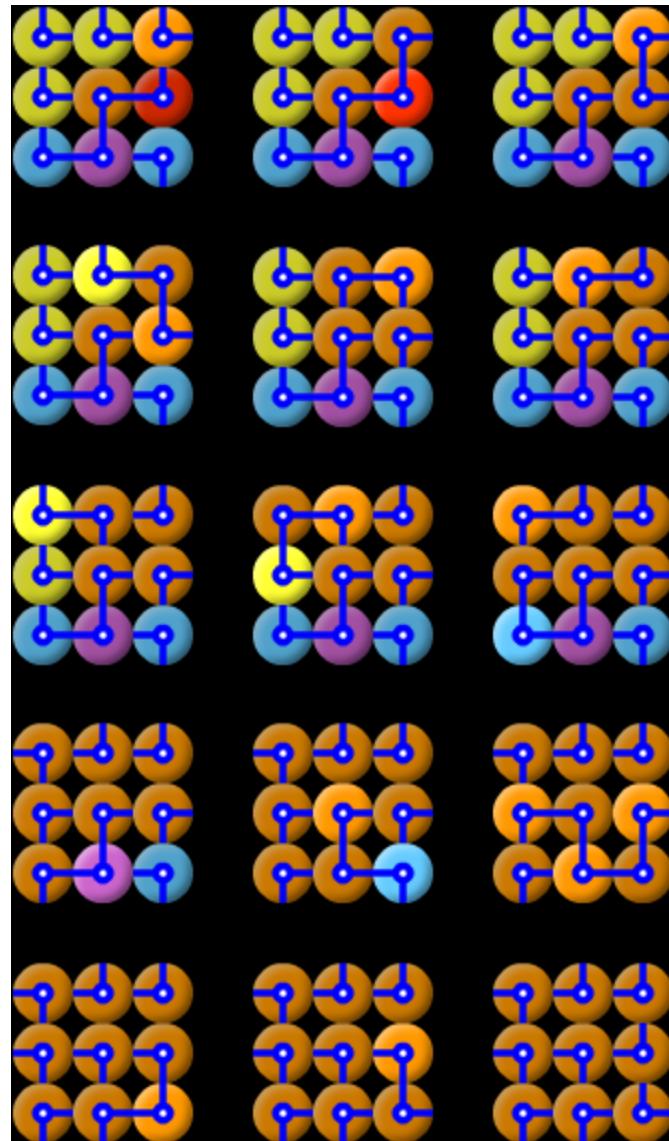
**Explanation:**

Your score for this input file is  $3+1001 = 1004$ .

Let's see what happens in the first test case if we apply the rotations from the output. The first two rotations change the state of only one wheel each, as no wheel matches the chosen one after the rotation:



The third rotation starts a process consisting of 14 iterations, as shown below (if we look at the pictures from left to right, from top to bottom; the rotated wheels at each iteration are highlighted):



Now all wheels are of the same color, so we've achieved our goal.

## Test Case Generation

There are 10 official input files. Each input file contains exactly 19 test cases. The values of **N** in the corresponding test cases are listed in the table below:

Test Case	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9

<b>N</b>	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
	0	1	1	1	2	5	0	2	5	3	0	3	5	4	0	4	5

The initial positions and colors of the wheels are chosen randomly and uniformly in the corresponding ranges.

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Short II

## Short II Problem Code: SHORT2

Given a prime number  $p$ , find the number of pairs of integers  $(a, b)$  such that  $p < a, p < b$  and  $ab$  is divisible by  $(a-p)(b-p)$ .

### Input

The first line contains the number of test cases  $t$  ( $1 \leq t \leq 5$ ). Then  $t$  test cases follow, each test case consists of a line containing one prime integer  $p$  ( $1 < p < 10^{12}$ ).

### Output

For each test case output one line containing the required number. It's guaranteed that this number won't exceed  $2^{63}-1$ .

### Example

#### Input:

```
3
2
23
59
```

#### Output:

```
14
80
116
```

**Explanation:**

In the first test case the sought pairs are (3,3), (3,4), (3,5), (3,8), (4,3), (4,4), (4,6), (5,3), (5,12), (6,4), (6,8), (8,3), (8,6) and (12,5).

[Home](#) » [Compete](#) » [December 2011 Long Contest](#) » Hypertrees

## Hypertrees Problem Code: HYPER

A **hypergraph** is a generalization of a graph, where an edge can connect any number of vertices. A ***k*-uniform hypergraph** is a hypergraph such that all its hyperedges have size *k*. For more information, see [Wikipedia](#).

Let's call a particular hypergraph a **hypertree** if it is connected (that is, you can move from any vertex to any other vertex using only its hyperedges) and removing any of its hyperedges makes the hypergraph disconnected (note that this definition of hypertrees differs from the standard one).

Given just one integer **N**, find the number of 3-uniform hypertrees on **N** vertices. Two 3-uniform hypertrees are considered different if a hyperedge  $(u, v, w)$  exists such that it is present in exactly one of these hypertrees (note that the order of vertices in the hyperedge doesn't matter, and neither does the order of hyperedges in the hypertree).

### Input

The first line of the input contains an integer **T** -- the number of test cases (at most 15). Then **T** lines follow, each contains an integer **N** ( $3 \leq N \leq 17$ ).

**Important! Please include all code used in solving this problem in your solution.**

### Output

For each test case output one line containing the requested number. It's guaranteed that this number won't exceed  $2^{63}-1$ .

### Examples

**Input:**

4

3

4

5

8

**Output:**

1

6

25

93268

**Explanation:**

There is just one 3-uniform hypertree on 3 vertices:  $\{(1,2,3)\}$ . There are six of them on 4 vertices:  $\{(1,2,3), (1,2,4)\}, \{(1,2,3), (1,3,4)\}, \{(1,2,3), (2,3,4)\}, \{(1,2,4), (1,3,4)\}, \{(1,2,4), (2,3,4)\}, \{(1,3,4), (2,3,4)\}$ . Two of the 25 possible hypertrees on 5 vertices are  $\{(1,2,3), (3,4,5)\}$  and  $\{(1,2,3), (1,2,4), (1,2,5)\}$ .

COOK17

[December Cook-Off](#)18 Dec 2011  
21:30:00

2 hours 40 minutes

753

[Home](#) » [Compete](#) » [December Cook-Off](#) » Ciel and A-B Problem

## Ciel and A-B Problem Problem Code: CIELAB

In Ciel's restaurant, a waiter is training. Since the waiter isn't good at arithmetic, sometimes he gives guests wrong change. Ciel gives him a simple problem. What is **A-B** (A minus B) ?

Surprisingly, his answer is wrong. To be more precise, his answer has exactly one wrong digit. Can you imagine this? Can you make the same mistake in this problem?

**Input**

An input contains 2 integers **A** and **B**.

**Output**

Print a wrong answer of **A-B**. Your answer must be a *positive* integer containing the same number of digits as the correct answer, and exactly one digit must differ from the correct

answer. Leading zeros are not allowed. If there are multiple answers satisfying the above conditions, anyone will do.

## Constraints

$1 \leq B < A \leq 10000$

## Sample Input

5858 1234

## Sample Output

1624

## Output details

The correct answer of 5858-1234 is 4624. So, for instance, 2624, 4324, 4623, 4604 and 4629 will be accepted, but 0624, 624, 5858, 4624 and 04624 will be rejected.

## Notes

The problem setter is also not good at arithmetic.

[Home](#) » [Compete](#) » [December Cook-Off](#) » Ciel and New Menu

## Ciel and New Menu Problem Code: CIEL8STR

Chef Ciel likes 8. Ciel's restaurant has many menus whose prices are multiples of 8. Now, Ciel has some digits written on a wooden board, and she'd like to cut the board to display prices in a new menu. In how many ways can Ciel choose consecutive digits from the board which denote integer multiples of 8?

In this problem, an integer must not have leading zeros. For example, 0, 8, 48, 1000 are integer multiples of 8, but 00, 5, 58, 01000 are not.

## Input

An input contains a string **S**, which denotes digits written on the wooden board.

## Output

Print the number of ways in which Ciel can choose consecutive digits which denote integer multiples of 8.

## Constraints

$1 \leq |\mathbf{S}| \leq 1000000 (10^6)$ , where  $|\mathbf{S}|$  means the length of  $\mathbf{S}$ .  
 $\mathbf{S}$  doesn't contain non-digit characters.

## Sample Input

5858

## Sample Output

2

## Output details

Ciel can choose 5, 8, 5, 8, 58, 85, 58, 585, 858 and 5858. Here, only 8 and 8 are multiples of 8.

[Home](#) » [Compete](#) » [December Cook-Off](#) » Ciel and Quiz Game

# Ciel and Quiz Game Problem Code: CIELQUIZ

Chef Ciel is going to organize a quiz game in her restaurant. Ciel has  $\mathbf{N}$  problems, and she will choose  $\mathbf{K}$  problems for the quiz game. Every participant has to answer all  $\mathbf{K}$  problems.

Ciel thinks that if a participant solves all  $\mathbf{K}$  problems correctly, the participant will get bored, and if a participant solves less problems correctly, the participant will feel sad. Therefore, Ciel decides to choose  $\mathbf{K}$  problems maximizing the probability that participants will solve exactly  $\mathbf{K-1}$  problems correctly.

The  $\mathbf{N}$  problems are numbered from 1 to  $\mathbf{N}$ . Ciel assumes, the problem  $i$  will be solved correctly with probability  $P_i\%$  for each  $i$ , and these are statistically independent of each other.

Can you tell what problems are chosen by Ciel?

## Input

The first line contains an integer  $\mathbf{T}$ , the number of test cases. Then  $\mathbf{T}$  test cases follow. The first line for each test case has 2 integers  $\mathbf{N}$  and  $\mathbf{K}$ . The next line has  $\mathbf{N}$  integers  $P_1, P_2, \dots, P_N$ .

## Output

For each test case, print  $\mathbf{K}$  distinct integers denoting the numbers of chosen problems. If there are multiple sets of problems maximizing the probability, then anyone will do.

## Constraints

$1 \leq T \leq 20$   
 $1 \leq K \leq N \leq 36$   
 $0 \leq P_i \leq 100$

### Sample Input

```

4
3 2
70 80 90
9 1
10 20 30 40 50 60 70 80 90
1 1
10
10 5
90 90 90 90 90 90 90 90 90

```

### Sample Output

```

1 2
1
1
5 8 1 4 9

```

### Output details

In the first case, all patterns choosing 2 problems are:

If problems 1 and 2 are chosen, the probability is  $0.7 * (1 - 0.8) + (1 - 0.7) * 0.8 = 0.14 + 0.24 = 0.38$

If problems 1 and 3 are chosen, the probability is  $0.7 * (1 - 0.9) + (1 - 0.7) * 0.9 = 0.07 + 0.27 = 0.34$

If problems 2 and 3 are chosen, the probability is  $0.8 * (1 - 0.9) + (1 - 0.8) * 0.9 = 0.08 + 0.18 = 0.26$

Therefore, Ciel will choose problems 1 and 2.

In the fourth case, there are multiple solutions. Any 5 distinct integers from 1 to 10 will be accepted.

[Home](#) » [Compete](#) » [December Cook-Off](#) » Ciel and Battle Arena

## Ciel and Battle Arena Problem Code: CIELBTL

Chef Ciel bought a new video game for guests to kill time until their orders arrive. In this game, you are a fighter in a battle arena, and you will fight against a fighter called *Shindannin*. In Japan, many people believe that the next day will be a happy day if they beat Shindannin, otherwise the next day will be an unhappy day.

Let's start by definitions of some variables.  $V_A$  and  $V_B$  denote your initial HP (health points) and Shindannin's initial HP respectively.  $S_A$  and  $S_B$  denote your strength and Shindannin's strength respectively.  $M_A$  denotes your initial MP (magical points), which is used when you use a skill. Note that only you can use a skill in the battle.

In each turn, the battle will go on as follows:

- Firstly, you can use a skill as many times as you like as long as your MP is positive. If you use a skill, your HP and Shindannin's HP are decreased by half and your MP is decreased by 1. If HP becomes a non-integer, HP will be rounded up to the nearest integer. To be more precise, new HP will be  $\text{ceil}(\text{old HP} / 2)$ .
- Next, an integer  $s$  is chosen in  $[0, S_A]$  uniformly randomly, and an integer  $t$  is chosen in  $[0, S_B]$  uniformly randomly.
- Then, your HP is decreased by  $t$ , and Shindannin's HP is decreased by  $s$  simultaneously.
- If both fighters have positive HP, next turn will occur.

When a fighter's HP is down to zero or a negative integer, this fighter loses. If both fighters' HP is down to zero or negative integer simultaneously, you will fight against Shindannin again with the same conditions, that is, HP and MP are recovered completely before next battle. There is no limit on the number of rematches. If you fight optimally, what is your winning percentage?

---

### Input

An input contains 5 integers  $V_A$ ,  $V_B$ ,  $S_A$ ,  $S_B$  and  $M_A$ .

---

### Output

Print the maximum winning percentage you can achieve. This value must have an absolute error no more than  $10^{-6}$ .

---

### Constraints

$1 \leq V_A, V_B \leq 100$

$1 \leq S_A, S_B \leq 100$

$0 \leq M_A \leq 5$

---

## Sample Input 1

5 5 3 3 0

---

## Sample Output 1

0.5

---

## Sample Input 2

5 5 3 3 1

---

## Sample Output 2

0.58886609097948

---

## Sample Input 3

5 8 8 5 8 1 3 0

---

## Sample Output 3

0.00011046536069

[Home](#) » [Compete](#) » [December Cook-Off](#) » Ciel and Random Walk

## Ciel and Random Walk Problem Code: CIELWALK

Today, Ciel's restaurant is closed. Ciel is bored, so she decides to take a walk randomly.

This town has  $N$  intersections and  $M$  one way roads. The  $i$ -th road connects from the  $A_i$ -th intersection to the  $B_i$ -th intersection. Ciel is in the 1st intersection now, and Ciel's house is in the  $N$ -th intersection.

If Ciel is in the  $i$ -th intersection, she will take a walk as follows:

- If  $i = N$ , her walk is ended.

- Otherwise, she chooses an integer  $k$  such that  $A_k = i$  uniformly randomly, she walks along the  $k$ -th road.

On the way,  $S$  intersections have beautiful flowers. Let the  $F_1$ -th intersection, the  $F_2$ -th intersection, ..., and the  $F_S$ -th intersection have beautiful flowers. Can you tell the probability that Ciel last sees the flowers in the  $F_i$ -th intersection? That is to say, can you tell the probability that she visits the  $F_i$ -th intersection, then she goes to  $N$ -th intersection without visiting other intersections having beautiful flowers?

---

## Input

The first line contains 3 integers  $N$ ,  $M$  and  $S$ . Then, next line has  $S$  integers  $F_1$ ,  $F_2$ , ...,  $F_S$ . Next  $M$  lines have 2 integers each,  $A_i$  and  $B_i$ .

---

## Output

An output should contain  $S$  lines. In the  $i$ -th line, print the probability that Ciel last sees beautiful flowers at the  $F_i$ -th intersection. These values must have an absolute error no more than  $10^{-6}$ .

---

## Constraints

$1 \leq S < N \leq 30$

$1 \leq F_1 < F_2 < \dots < F_S < N$

$A_i \neq B_i$

If  $i \neq j$  and  $A_i = A_j$ , then  $B_i \neq B_j$

For each  $1 \leq i < N$ , Ciel can go from the  $i$ -th intersection to the  $N$ -th intersection, directly or indirectly.

---

## Sample Input 1

4 6 3

1 2 3

1 2

2 1

1 3

3 1

2 4

3 4

---

### Sample Output 1

0

0.5

0.5

---

### Sample Input 2

4 6 1

2

1 2

2 1

1 3

3 1

2 4

3 4

---

### Sample Output 2

0.66666666666667

---

### Sample Input 3

2 1 1

1

1 2

---

### Sample Output 3

---

## Notes

The sum of output values can be less than 1, because Ciel may go back to her house without seeing beautiful flowers.

AN12	<a href="#">January 2012 Challenge</a>	01 Jan 2012 15:00:00	10 days	500
------	--	-------------------------	---------	-----

[Home](#) » [Compete](#) » [January 2012 Challenge](#) » Houses and Restaurants

## Houses and Restaurants Problem Code: RHOUSE

---

There is a city where people are fond of visiting restaurants. In fact, any building in this town is either a house or a restaurant. There are  $N$  buildings in the city and they are conveniently numerated by integer numbers from 1 to  $N$ . To move from some building to others there are  $M$  two-way roads. Each road connects two buildings. It is possible that there are several roads that connect the same pair of buildings, but there won't be a road that connects a building to itself. Chef wants to make the ways to his restaurants more interesting. To do that he is going to decorate some roads. Each road has its own integer cost of decorating. Some costs may be negative. If the cost is negative Chef will get some reward for decorating this road. Now Chef doesn't have much money so he wants to decorate roads in such a way that from each house, at least one restaurant is reachable using only decorated roads. The total cost of decorated roads should be minimal. So, your task is to help him to calculate the minimal cost of any satisfactory subset. It might be negative if Chef's rewards for decorating some roads are greater than his spent money.

---

## Input

There first line of input contains an integer  $T$  - the number of test cases.

Then  $T$  test cases follow. For each test case there will be integers  $N$  and  $M$  - the number of buildings and the number of roads.

Then a string of  $N$  letter follows.  $i$ -th symbol of this string is "R" if the corresponding building is a restaurant and "H" if it is a house.

Then there are  $M$  lines. Each line consists of three integers -  $X_i$   $Y_i$   $Z_i$ . They describe a road that connects buildings  $X_i$  and  $Y_i$  with the cost of decorating equal to  $Z_i$ .

---

## Output

For each test case output the minimal cost of any satisfactory subset of roads.

---

## Constraints

2<=N<=100000

1<=M<=400000

-20000<=Z<sub>i</sub><=20000

Sum over all N in a single input file will not exceed 200000.

Sum over all M in a single input file will not exceed 400000.

It is guaranteed that you can reach every building from any other building.

There is at least one restaurant in the city.

---

## Example

**Input:**

3

3 5

HHR

1 2 3

1 2 5

1 3 10

3 2 -1

3 1 7

2 2

RR

1 2 1

2 1 2

3 3

HRR

1 2 1

1 3 2

2 3 3

**Output:**2  
0  
1[Home](#) » [Compete](#) » [January 2012 Challenge](#) » [Lucky Sum](#)

## Lucky Sum Problem Code: LUCKY3

---

Chef loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47, 744, 4** are lucky and **5, 17, 467** are not.

Chef also use term "**lucky sum**". Lucky sum is an operation between two integers. Let the first integer is **A**, **A[i]** equals **i-th** digit of **A** (0-base numeration, from right to left) and the second integer is **B**, **B[i]** equals to **i-th** digit of **B**. Then the lucky sum of **A** and **B** is equal to **C**, **C[i] = max(A[i], B[i])**. If **i** is greater than or equal to size of integer, the **i-th** digit is equal to **0**. For example, the lucky sum of **47** and **729** equals **749**, the lucky sum of **74** and **92** and **477** equals **497**.

Chef has an array **W** of integers. Find a number of non-empty subsequences of **W** such that the lucky sum of integers in that subsequences is a lucky number.

Subsequence of **W** is created by erasing some number (probably zero) elements from **W**.

---

### Input

First line contains one number **T**, number of test cases. Each test is formed as follows: first line contains integer **n** - number of integers in **W**, next line contains **n** integers - array **W** for corresponding test.

---

### Output

For each **T** test cases print one integer - result for the corresponding test.

---

### Constraints

$1 \leq T \leq 10$

$1 \leq n \leq 50$

$1 \leq Wi < 10^9$

## Example

**Input:**

2

2

4 7

3

43 87 44

**Output:**

3

2

[Home](#) » [Compete](#) » [January 2012 Challenge](#) » Gap Filler Game

## Gap Filler Game Problem Code: GAPFILL

*Now we are all bored playing, probably that is why they call it board game.*

Chef is facing severe problems at his restaurant. Because his kitchen is small so only small number of cooks can fit in which results in slow service. Customers don't like waiting for their food. While he has already started looking for new place to move in, this might take some time. So in meanwhile he has come up with an innovative idea. He'd put up a board game puzzle on all tables so that customers can play the game while waiting for their orders to arrive. To make this gap filler idea more viral, he'd further give a discount to people who successfully solve the puzzle.

Game that he has chosen consists of  $N * M$  grid with each cell having a light and a switch. Once a switch of a cell is pressed, all lights in the same column or row including the cell toggle once. Note that light in the cell whose switch is pressed is toggled once and not twice. Game starts with some lights on and rest off and the objective of the game is to turn

on all the lights.

He however fears that a lot of people may win discounts. He is a great cook but he is not good with puzzle solving. He seeks your help in determining how easy or tough this game is. Given board dimensions, your task is to find out how many initial configurations of the game can be solved. Two configurations are different if there is at least one cell which is off in one and on in other. As this number might get huge, only print the answer modulo  $10^9 + 7$ .

---

## Input

First line of the input contains a single integer  $T$ , the number of test cases. Each of the next  $T$  lines represents a single test case containing two integers  $N$  &  $M$  separated by a space.  $N$  &  $M$  are the dimensions of the board. You can assume that  $T \leq 50000$  and  $1 \leq N, M \leq 10^{18}$

---

## Output

For each test case, print the number of initial configurations of  $N * M$  board that can be solved modulo  $10^9 + 7$ .

---

## Example

### Input:

```
3
1 1
1 3
3 3
```

### Output:

```
2
2
32
```

### Description :

In first case  $1 \times 1$  board can have 2 configurations - off or on. Both can be solved.

In second case we've a row of 3 lights. There are total of 8 configurations. Out of these only 2 configurations can be solved (one with all lights on and one with all lights off)

[Home](#) » [Compete](#) » [January 2012 Challenge](#) » Ambidextrous Chefs

## Ambidextrous Chefs Problem Code: AMBIDEX

---

A recent spike in demand for competent chefs has caused the emergence of a highly specialized training program, in which chefs are trained to use 2 tools simultaneously (one in each hand). These chefs, called "ambidextrous chefs", can theoretically perform the same work as 2 ordinary chefs. The problem with these ambidextrous chefs is that their training is so specific that they cannot use any tools except for the 2 tools they can use simultaneously.

Head Chef wants to form chef teams of ambidextrous chefs. In order to form a complete team, for every tool present there must be at least one chef capable of operating it. Chef wants to create as many teams as possible, but at the same time hire as few chefs as possible. See the scoring section for details.

---

### Input

Input begins with an integer  $T$ , the number of test cases. Each test case begins with 2 integers  $N$  and  $M$ .  $N$  is the number of different tools, and  $M$  the number of ambidextrous chefs. 2 lines follow with exactly  $M$  integers each, each between 1 and  $N$ . The  $i$ -th integer of each line indicates a tool that the  $i$ -th chef can use. It is guaranteed that the  $i$ -th integer of the first line will be different from the  $i$ -th integer of the second line. It is further guaranteed that each tool will be usable by at least one chef.

---

### Output

For each test case, output  $M$  integers on a line, one per ambidextrous chef. The number should be 0 if the chef should not be hired. Otherwise, the number should be any positive number not exceeding 10000 to indicate the team to which that chef should be assigned. Chefs with the same team number will be assigned to the same team.

---

### Scoring

Your score for each test case is  $(C \cdot N - H)/M$ , where  $C$  is the number of complete chef teams formed,  $N$  is the number of tools,  $H$  is the total number of chefs hired, and  $M$  is the total number of chefs. If this value is non-positive for any test case, the submission will be judged "Wrong Answer" instead. Your score for each test file is the average of the scores for the individual test cases. Your overall score is the average of your scores on the individual test files. Note that optimal solutions are not required, but better solutions will earn more points.

---

### Sample Input

2

5 9

5 1 5 5 1 3 4 4 1

3 2 2 3 5 1 2 2 4

8 12

6 6 8 7 1 1 6 5 3 8 7 6

2 8 1 5 8 2 7 7 1 3 4 4

## Sample Output

2 2 1 3 3 1 3 2 1

2 0 1 1 2 1 0 2 2 1 2 1

The score for the first test case is  $(3*5-9)/9 = 0.666667$ , and the score for the second test case is  $(2*8-10)/12 = 0.5$ . Hence the overall score is 0.583333.

## Test case generation

For each official test file, T is 10. For each test case, N is chosen randomly between 10 and 100, inclusive, and M between 200 and 1000, inclusive. Then  $\text{ceiling}(2*M/(N-1))$  of each type of tool is placed in a sack, and one by one each of the chefs will randomly remove 2 different tools from the sack. If after all chefs have selected tools there is a tool that none of the chefs selected, the process is restarted with the same values of N and M.

[Home](#) » [Compete](#) » [January 2012 Challenge](#) » Card Shuffle

## Card Shuffle Problem Code: CARDSHUF

Chef is organizing an online card game tournament and for this purpose he has to provide a card shuffling software. This software has to simulate the following shuffling process. A stack of N cards is placed face down on the table. Cards in the stack are ordered by value. Topmost card has value 1 and the one on the bottom has value N. To shuffle the cards we repeat the following steps M times:

1. take A cards from the top of the deck.
2. take another B cards from the top of the deck.

3. put the A cards, which you removed in the first step, back on top of the remaining deck.
4. take C cards from the deck
5. put the B cards, which you're still holding from the second move, **card by card** on top of the deck.
6. finally, return the block of C cards on top

Note: taking a block of cards from the top of the deck does not change their order. The entire block is removed in a single move and not card by card. The only exception is the fifth move, where you return cards one by one from the top.

---

## Input

The first line contains integers N and M. The following M lines describe the moves by integers  $A_i$ ,  $B_i$ ,  $C_i$  as described in the previous section.

In the spirit of random card shuffling, all test cases were generated with uniform random distributions to select where to cut the deck of cards.

---

## Output

In a single line output the cards in the deck after performing all the moves. Cards should be listed from top of the deck to bottom and separated by spaces.

---

## Constraints

- $1 \leq N, M \leq 100\,000$
- 

## Example

Input:

10 2

6 2 2

5 3 6

Output:

1 2 8 7 3 9 6 5 4 10

## Misinterpretation 2 Problem Code: MISINT2

---

Chef's brother likes to put words in Chef's mouth. Chef hates it about him of course. He has decided to become extremely careful about what he speaks. Fortunately, he knows how his brother transforms the words, that Chef uses. He has decided to use only those words which remain the same even after his brother's transformation!

If Chef speaks an **N** letter word, his brother moves all the letters which are at even positions (assuming, numbering of positions starts from 1), to the beginning of the word; and the rest of the letters follow as they appeared in the word, e.g. abdef becomes beadf; cdcd becomes ddcc.

Chef wants to know how many words he can use, provided that the length of each word is between **L** and **R** inclusive and each word is composed of lowercase letters of the English alphabet. They use an ancient language in Byteland, which allows all possible words within the above definition!

---

### Input

The first line contains a positive integer **T**, the number of test cases. In the following lines, **T** test cases follow. Every test case is a single line that contains two space separated positive integers, **L** and **R**. **L** is lower bound and **R** is upper bound for the length of the word that Chef wants to use.

---

### Output

For each test case, print the number of words with length between **L** and **R** inclusive that Chef can use; that is, number of words, which remain the same after brother's transformation. Since the result can be quite large, output the result modulo **1000000007**.

---

### Constraints

$1 \leq T \leq 5$   
 $1 \leq L \leq R \leq 10^{10}$   
 $R - L \leq 50000$

---

### Sample input

3

1 5

6 7

45 50

---

## Sample output

1430

18252

871229844

COOK18	<a href="#">January Cook-Off</a>	22 Jan 2012 21:30:00	2 hours 30 minutes	436
--------	----------------------------------	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [January Cook-Off](#) » Collision in Space

## Collision in Space Problem Code: COLLIDE

Did you hear about the Nibiru collision ? It is a supposed disastrous encounter between the earth and a large planetary object. Astronomers reject this idea. But why listen to other people's beliefs and opinions. We are coders above all, so what better way than to verify it by a small code. The earth and N asteroids are in the 2D plane. Each of them is initially located at some integer coordinates at time = 0 and is moving parallel to one of the X or Y axis with constant velocity of 1 unit per second.

Direction of movement is given as 'U' ( Up = towards positive Y ), 'D' ( Down = towards negative Y ), 'R' ( Right = towards positive X ), 'L' ( Left = towards negative X ). Given the initial position and the direction of movement of the earth and each of the N asteroids, find the earliest time at which the earth collides with one of the asteroids. If there can not be any collisions with the earth, print "SAFE" ( without quotes ). You can ignore the collisions between asteroids ( i.e., they continue to move in same direction even after collisions between them ).

---

## Input

First line contains T, number of test cases. T cases follow. In each test case, first line contains XE YE DIRE, where (XE,YE) is the initial position of the Earth, DIRE is the direction in which it moves. Second line contains N, the number of asteroids. N lines follow, each containing XA YA DIRA, the initial position and the direction of movement of each asteroid. No asteroid is initially located at (XE,YE)

---

## Output

For each test case, output the earliest time at which the earth can collide with an asteroid (rounded to 1 position after decimal). If there can not be any collisions with the earth, print "SAFE" (without quotes).

---

## Constraints

$1 \leq T \leq 10$   
 $1 \leq N \leq 2012$   
 $-100 \leq X_E, Y_E, X_A, Y_A \leq 100$   
 $(X_E, Y_E) \neq \text{any of } (X_A, Y_A)$   
DIRE, DIRA is one of 'U', 'R', 'D', 'L'

---

## Example

### Input:

```
3
0 0 R
2
1 -2 U
2 2 D
1 1 U
1
1 0 U
0 0 R
1
3 0 L
```

### Output:

```
2.0
SAFE
```

1.5

**Explanation:**

Case 1 :

Time 0 - Earth (0,0) Asteroids { (1,-2), (2,2) }

Time 1 - Earth (1,0) Asteroids { (1,-1), (2,1) }

Time 2 - Earth (2,0) Asteroids { (1,0), (2,0) }

Case 2 :

The only asteroid is just one unit away below the earth and following us always, but it will never collide :)

Case 3 :

Time 0 - Earth (0,0) Asteroid (3,0)

Time 1 - Earth (1,0) Asteroid (2,0)

Time 1.5 - Earth (1.5,0) Asteroid (1.5,0)

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted.

[Home](#) » [Compete](#) » [January Cook-Off](#) » Makx Sum

## Makx Sum Problem Code: KSUBSUM

---

Did you know that the Hindus believe there are 330 million deities ? Lets discuss about them in long contests. The three main forces of god Brahma, Vishnu and Mahesh, also called Trimurti ( 3 forms ), want to decide when to end the world. Given an array A, which contains the 'Good' ( or 'Evil' ) score of each place on earth, they can easily find the maximum sum of the scores of a (contiguous) sub-array of places. But this is not much useful to them. So, each of them gives a value K and want to know the Kth maximum sum of scores of a contiguous sub-array. In other words, find the value of S[K] (1-based index), where the array S contains the sums of all possible contiguous sub-arrays in non-increasing order.

---

### Input

First line contains T, number of test cases. T cases follow. In each test case, first line contains N K1 K2 K3. Next line contains N space separated integers.

---

### Output

For each test case, output the K1<sup>th</sup>, K2<sup>th</sup> and K3<sup>th</sup> maximum sum in a single line, separated by a single space. See sample cases and explanation for more clarity.

---

## Constraints

$1 \leq T \leq 3$   
 $2 \leq N \leq 10000$   
 $1 \leq K1 < K2 < K3 \leq 2012$   
 $K3 \leq N*(N+1)/2$   
 $-10000 \leq A[i] \leq 10000$

---

## Example

**Input:**

3

3 1 2 3

10 20 30

3 3 4 6

10 20 30

4 2 6 10

20 -15 10 -15

**Output:**

60 50 30

30 30 10

15 -5 -20

**Explanation:**

Case 1 & 2 :  $A = \{ 10, 20, 30 \}$ . All possible sums of sub-arrays in non-increasing order,  $S[1..6] = \{ 60, 50, 30, 30, 20, 10 \}$

**Warning :** "Large input / output. Be careful with certain languages. Eg: Prefer using scanf/printf to cin/cout for C++"

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted.

[Home](#) » [Compete](#) » [January Cook-Off](#) » Zeta-Nim Game

## Zeta-Nim Game Problem Code: TAKEAWAY

---

Did you hear about Zeta Reticuli ? Its a star system deep in the southern sky, and this is where the first Alien film is set. Zetas are the aliens living there and they like playing games with humans. This year they selected Nancy and took her with them to play the famous Zeta-Nim game. Given N piles of stones and a value K, each pile  $i$  has a counter  $C_i$  which is the maximum number of stones that can be removed from it. Initially,  $C_i = K$  for  $0 \leq i < N$ . Each turn involves selecting one of the N piles, say pile  $i$ , and removing  $R$  ( $1 \leq R \leq C_i$ ) stones from it and updating  $C_i = R$ . In other words, the number of stones removed from a pile should not be more than the number of stones removed from the same pile before, starting with removal of at most  $K$  stones.

Zeta and Nancy take turns alternately and the one who can not remove any more stones lose. Given  $N$ ,  $K$  and the number of stones in each of the  $N$  piles, find who wins the game if both play optimally and Nancy starts first.

---

### Input

First line contains  $T$ , number of test cases. Each case contains 2 lines. First line contains  $N$   $K$  and the second line contains the array  $A$  of  $N$  space separated integers.  $A[i]$  is the number of stones in the  $i$ th pile.

---

### Output

For each test case, print the winners name "Nancy" or "Zeta" ( without quotes ).

---

### Constraints

$1 \leq T \leq 200$   
 $1 \leq N \leq 50$   
 $1 \leq A[i] \leq 2012$   
 $1 \leq K \leq 100,000$

---

### Example

**Input:**

1 1

3

2 2

3 3

5 10

5 10 15 20 25

**Output:**

Nancy

Zeta

Nancy

**Explanation:**

Case 1 : Only one pile of 3 stones and  $K = 1$ . At most 1 stone can be removed in each step ( and of course at least 1 stone must be removed, if possible). Nancy removes 1 stone, then Zeta removes 1 stone and finally Nancy removes the last stone and wins !

Case 2 : Two piles of stones with 3 stones in each and  $K = 2$ . Initial pile sizes are  $\{3,3\}$  and after Nancy takes either 1 or 2 stones from one of them, the piles will be one of  $\{2,3\}$  or  $\{1,3\}$ . If its  $\{2,3\}$ , Zeta can remove a stone from the 2nd pile to make it  $\{2,2\}$  and win eventually. If its  $\{1,3\}$ , Zeta can remove two stones from the 2nd pile to make it  $\{1,1\}$  and win in its next turn. So, no matter what step Nancy takes in her first turn, Zeta always has a winning strategy !

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted.

[Home](#) » [Compete](#) » [January Cook-Off](#) » Remys last tour

## Remys last tour Problem Code: FALLDOWN

Do you know what b'ak'tun means ? Its an enormous cycle in the Mayan calendar, roughly around 400 solar years. The 13th b'ak'tun is going to end this December and our chef Remy

is worried about it. What can a chef do if the world is going to end soon ? Enough of making recipes, time to visit the restaurants around the world. He prepares a list of restaurants along with their ratings. This is given as a 2D array A having R rows and C columns.  $A[i][j]$  has the rating of the  $j$ th restaurant in the  $i$ th row. Rows are numbered 1 to R from top to bottom and columns are numbered 1 to C from left to right.

Remy can start his journey from any restaurant in the top row ( row# 1 ) and end at any restaurant in the bottom row ( row# R ). He is afraid that the world may end from top to bottom, and so he will never go up from a restaurant. He can only move left, right or down ( i.e., to cell  $(i, j-1)$  or  $(i, j+1)$  or  $(i+1, j)$  ) in each step. Along the path he takes, he sums up the ratings of all the restaurants visited. Of course, each visited restaurant's rating should be added *exactly once*, even if he visits it again along the path. Find the maximum possible sum of ratings.

( Read the Input section for details on reading / generating A )

---

## Input

First line contains  $T$ , number of test cases.  $T$  cases follow. In each test case, first line contains  $R\ C\ I$ , number of rows, number of columns and the method of taking input, respectively.

If  $I == 1$ ,  $R$  lines follow, each containing  $C$  integers, representing the grid A

If  $I == 2$ , next line contains 4 integers  $X\ P\ Q\ M$ . Generate the grid A using the following method.

Initialize,  $cur = X$

for  $i : 0$  to  $R-1$

    for  $j : 0$  to  $C-1$

    {

$cur = ( cur * P + Q ) \% M;$

$A[i][j] = X - cur$

    }

( \* is used for multiplication and % is the modulo operator. )

*The writer's solution does not depend on the factors  $X, P, Q, M$  used for input generation.*

*They are used just to decrease the size of input files ( Max 8MB ).*

---

## Output

For each test case, output the maximum possible sum of ratings along a path that starts in the top row and ends in the bottom row.

---

## Constraints

$1 \leq T \leq 5$   
 $1 \leq R, C \leq 2012$   
 $-500 \leq A[i][j] \leq 500$   
 $1 \leq P, Q \leq 1,000,000$   
 $1 < X < M \leq 500$

---

## Example

**Input:**

2

3 5 1

1 1 -5 3 1

-1 1 1 -1 3

2 -3 2 -1 1

3 4 2

10 13 13 29

**Output:**

10

3

**Explanation:**

The path with maximum sum of ratings in Case 1 is shown in bold

1	1	-5	3	1
-1	<b>1</b>	1	-1	<b>3</b>
2	-3	<b>2</b>	-1	1

**Warning :** "Large input / output. Be careful with certain languages. Eg: Prefer using scanf/printf to cin/cout for C++"

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted.

[Home](#) » [Compete](#) » [January Cook-Off](#) » Moving between Floors

## Moving between Floors Problem Code: FLOORSMV

Did you know that the world's tallest structure is Burj Khalifa in Dubai ? To survive when the world ends, Mr.Turing thinks that he should climb one of those skyscrapers ( tall buildings ). There are N tall buildings numbered  $b_1, b_2, \dots, b_N$  in that order and  $b_i$  and  $b_{i+1}$  are adjacent to each other for  $1 \leq i < N$ . Each of these have exactly F number of floors,  $f_1, f_2, \dots, f_F$  from bottom to top. Floors  $f_i$  and  $f_{i+1}$  in the same building are adjacent to each other for  $1 \leq i < F$  and it takes one unit of time to move between them. It also takes one unit of time to move between the floor  $f_1$  of a building and  $f_1$  of its adjacent building.

Turing is afraid that some buildings may still collapse and so he lists down the bridges connecting some of them, to know how to escape quickly. There are M bridges, each of which connects two floors of different buildings. Each of these bridges is given as  $bi fi T bj fj$ , meaning it takes T units of time to move along this bridge which connects the floor  $fi$  of building  $bi$  and the floor  $fj$  of building  $bj$ . All ways are bidirectional. Turing wants to know, given  $bq$  and  $fq$ , the sum of the shortest time it takes to reach  $(b,f)$  starting from  $(bq,fq)$ , for all possible  $1 \leq b \leq N$  ( except  $b = bq$  ) and  $1 \leq f \leq F$ . In other words, given  $(bq,fq)$ , find the value of  $\sum D(bq,fq,b,f)$  for all possible  $(b,f)$  except  $b = bq$ , where  $D(bi,fi,bj,fj)$  is the shortest time it takes to reach floor  $fi$  of building  $bi$  from floor  $fj$  of building  $bj$ . Get ready to answer his Q such queries.

---

### Input

First line contains  $N F M$ .  $N$  is the number of buildings ,  $F$  is the number of floors in each building and  $M$  is the number of bridges.  $M$  lines follow, each of the form  $bi fi T bj fj$ , as mentioned in the problem statement. Next line contains  $Q$ , the number of queries and  $Q$  lines follow, each of the form  $bq bf$

---

## Output

For each test case, output the sum of the shortest time it takes from  $(b_i, f_i)$  to reach each floor of every other building.

---

## Constraints

$2 \leq N, M \leq 100$   
 $1 \leq F, T \leq 1,000,000$   
 $1 \leq Q \leq 2012$   
 $1 \leq b_i, b_j, b_q \leq N$  and  $1 \leq f_i, f_j, f_q \leq F$

---

## Example

### Input:

2 4 1

1 3 1 2 3

3

1 1

1 2

1 3

### Output:

10

10

8

**Note :** There are multiple test sets, and the judge shows the sum of the time taken over all test sets of your submission, if Accepted.

FEB12	<a href="#">February 2012 Challenge</a>	01 Feb 2012 15:00:00	10 days	2000
-------	---	-------------------------	---------	------

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Count of Maximum

## Count of Maximum Problem Code: MAXCOUNT

Given an array A of length N, your task is to find the element which repeats in A maximum number of times as well as the corresponding count. In case of ties, choose the smaller element first.

---

### Input

First line of input contains an integer T, denoting the number of test cases. Then follows description of T cases. Each case begins with a single integer N, the length of A. Then follow N space separated integers in next line. Assume that  $1 \leq T \leq 100$ ,  $1 \leq N \leq 100$  and for all  $i$  in  $[1..N]$  :  $1 \leq A[i] \leq 10000$

---

### Output

For each test case, output two space separated integers V & C. V is the value which occurs maximum number of times and C is its count.

---

### Example

**Input:**

```
2
5
1 2 3 2 5
6
1 2 2 1 1 2
```

**Output:**

```
2 2
```

**Description:**

In first case 2 occurs twice whereas all other elements occur only once.

In second case, both 1 and 2 occur 3 times but 1 is smaller than 2.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » [Lucky Long](#)

## Lucky Long Problem Code: LUCKY5

---

Chef loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47, 744, 4** are lucky and **5, 17, 467** are not.

Chef has a positive integer **N**. He can apply any of the following operations as many times as he want in any order:

- Add **1** to the number **N**.
- Take some digit of **N** and replace it by any non-zero digit.
- Add any non-zero leading digit to **N**.

Find the minimum number of operations that is needed for changing **N** to the lucky number.

---

### Input

The first line contains a single positive integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains a positive integer **N** without leading zeros.

---

### Output

For each **T** test cases print one integer, the minimum number of operations that is needed for changing **N** to the lucky number.

---

### Constraints

$1 \leq T \leq 10$

$1 \leq N < 10^{100000}$

---

### Example

**Input:**

3

25

46

99

**Output:**

2

1

2

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Word Couting

## Word Couting Problem Code: WCOUNT

Chef has decided to retire and settle near a peaceful beach. He had always been interested in literature & linguistics. Now when he has leisure time, he plans to read a lot of novels and understand structure of languages. Today he has decided to learn a difficult language called Smeagolese. Smeagolese is an exotic language whose alphabet is lowercase and uppercase roman letters. Also every word on this alphabet is a meaningful word in Smeagolese. Chef, we all know is a fierce learner - he has given himself a tough exercise. He has taken a word and wants to determine all possible anagrams of the word which mean something in Smeagolese. Can you help him ?

### Input

Input begins with a single integer  $T$ , denoting the number of test cases. After that  $T$  lines follow each containing a single string  $S$  - the word chef has chosen. You can assume that  $1 \leq T \leq 500$  and  $1 \leq |S| \leq 500$ . You can also assume that no character repeats more than 10 times in the string.

### Output

Output one line per test case - the number of different words that are anagrams of the word that chef has chosen. As answer can get huge, print it modulo  $10^9 + 7$

---

## Example

**Input:**

4

ab

aa

aA

AAbaz

**Output:**

2

1

2

60

**Description:** In first case "ab" & "ba" are two different words. In third case, note that A & a are different alphabets and hence "Aa" & "aA" are different words.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » [Lucky Count](#)

## Lucky Count Problem Code: LUCKY1

---

Chef loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47, 744, 4** are lucky and **5, 17, 467** are not.

Let  $F_d(x)$  equals to the number of digits  $d$  in decimal representation of the positive integer  $x$ . Chef interests only in functions  $F_4(x)$  and  $F_7(x)$ . For the given positive integer  $N$  he wants to know the total number of different pairs  $(L; R)$  such that  $F_4(L) + F_4(L + 1) + \dots + F_4(R)$  equals to  $F_7(L) + F_7(L + 1) + \dots + F_7(R)$  and  $1 \leq L \leq R \leq N$ .

---

## Input

The first line contains a single positive integer  $T$ , the number of test cases.  $T$  test cases follow. The only line of each test case contains a positive integer  $N$ .

---

## Output

For each test case, output a single line containing the answer for the corresponding test case.

---

## Constraints

$1 \leq T \leq 100000$

$1 \leq N \leq 100000$

---

## Example

**Input:**

3

3

10

100

**Output:**

6

31

1266

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Box and Ball System

## Box and Ball System Problem Code: BBSYSTEM

---

Chef Ciel has forgotten the combination to the safe. It's a serious incident, because the safe contains this month's waitstaff salaries.

To open the safe,  $N$  boxes and  $N$  balls are used. The safe has  $N$  boxes that numbered from 1 to  $N$  uniquely. Each box can contain only one ball. Now, the box  $i$  contains one ball that numbered  $i$ , and the safe is locked.

The only things which Ciel remember for unlocking the safe are the followings:

1. She must put every ball into some box.
2. Let the box  $i$  contains the ball  $A_i$ . When the safe is opened the number of divisors of  $i$  equals to the number of divisors of  $A_i$  for all  $i$  from 1 to  $N$ .

How many combinations which satisfy above conditions should she check? The number of combinations can be very large, so you should print this number modulo 500009 ( $5 \cdot 10^5 + 9$ ).

---

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The only line of each test case contains an integer  $N$ .

---

## Output

For each test case, print the number of combinations modulo 500009 ( $5 \cdot 10^5 + 9$ ).

---

## Constraints

$1 \leq T \leq 100000 (10^5)$   
 $3 \leq N \leq 2000000000 (2 \cdot 10^9)$

---

## Sample Input

3  
3  
5  
100

---

## Sample Output

1  
5  
43264

---

## Output details

In the first case, the valid combination is

Box: 123

Ball: 132

since the number of divisors of 2 is equal to the number of divisors of 3.

In the second case, the valid combinations are

Box: 12345 12345 12345 12345 12345

Ball: 12543 13245 13542 15243 15342

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Restore the Recipe

## Restore the Recipe Problem Code: RSRECIPE

Chef has had a recipe. He had written it as a sequence of **N** integer numbers **A[1], A[2], ..., A[N]**. To be on the safe side, he decided to write out **M** triples of numbers, where **i**-th triple is composed of three numbers **X<sub>i</sub>, Y<sub>i</sub>** and **Z<sub>i</sub>**. It means that the sum of numbers in the recipe from **X<sub>i</sub>**-th to **Y<sub>i</sub>**-th is equal to **Z<sub>i</sub>**. In other words, **A[X<sub>i</sub>] + A[X<sub>i</sub> + 1] + ... + A[Y<sub>i</sub>] = Z<sub>i</sub>**.

Unfortunately, the recipe has been recently lost. So Chef needs to restore his recipe using these **M** triples. Your task is to help him.

### Input

The first line contains two space separated integer numbers **N** and **M**. Then **M** lines follow. Each line contains three space separated integer numbers **X<sub>i</sub>, Y<sub>i</sub>** and **Z<sub>i</sub>**. It means that the sum of numbers in the recipe from **X<sub>i</sub>**-th to **Y<sub>i</sub>**-th is equal to **Z<sub>i</sub>**.

### Output

If there is a solution output **N** space separated integer numbers - the sequence that you have restored. If there are several solutions you can output any of them. However, any number in the output should be no more than  $10^{14}$  by its absolute value. If it is impossible to restore the sequence output "-1" without quotes.

### Constraints

- $2 \leq N \leq 65536$
- $1 \leq M \leq 200000$
- $1 \leq X_i \leq Y_i \leq N$
- $|Z_i| \leq 10000000000 (10^9)$

- If It is possible to restore the sequence then there is way to do this such that every number in the restored sequence is in range [-10000; 10000]
- 

## Example

### Input 1:

4 4

1 2 3

2 3 1

3 4 -2

1 4 1

### Output 1:

1 2 -1 -1

### Input 2:

5 3

1 3 4

4 5 6

1 5 9

### Output 2:

-1

---

## Explanation

In the first test case the answer is not unique. Possible answers are also **{0, 3, -2, 0}, {2, 1, 0, -2}** and many others.

In the second test case first two conditions imply  $A[1] + A[2] + A[3] = 4$  and  $A[4] + A[5] = 6$  and hence  $A[1] + A[2] + A[3] + A[4] + A[5] = (A[1] + A[2] + A[3]) + (A[4] + A[5]) = 4 + 6 = 10$ . And it contradicts to the third condition. Hence it is impossible to restore the recipe.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Smart Chef vs Evil Chef

## Smart Chef vs Evil Chef Problem Code: SMVSEVIL

Evil Chef recently challenged Smart Chef in the following way: Evil Chef will design a maze, and Smart Chef will provide a route through the maze. The maze consists of a square grid of cells, one of which is marked as the starting point, and one of which is marked as the exit. A robot will be placed at the starting point, which will attempt to execute each move of Smart Chef's route one at a time. If the robot cannot move in the specified direction (either because the cell in that direction is impassable, or because doing so would move the robot off the grid) the move will be ignored. Otherwise it will move one space in the specified direction. It then checks to see if it has exited the maze, and if so ignores the rest of the route. If the robot makes it to the exit, Smart Chef wins. Otherwise Evil Chef wins.

But Smart Chef's task is not as easy as it seems. Smart Chef must provide the route without knowing what the maze looks like. Furthermore, Evil Chef may cheat and change the layout of the maze after he finds out Smart Chef's route. Luckily, Evil Chef is not very smart, and only knows how to make a limited number of mazes. And Smart Chef is so smart that he knows all the mazes Evil Chef is capable of making. He's also smart enough to know when to ask for help, and has asked for your help in designing the route.

Your task is to design a route guarantees Smart Chef will win. In other words, your route must be able to solve all of Evil Chef's mazes. Your goal is to minimize the length of the route, but it is not necessary to find the shortest possible route (see the scoring section for additional details). As long as your route solves all the mazes, does not exceed 10000 moves in length, and no prefix of your route solves all mazes, your solution will be judged as correct.

---

### Input

Input will begin with an integer  $T$ , the number of test cases that follow. Each test case begins with an integer  $K$ , the number of mazes.  $K$  maze descriptions follow. Each maze description begins with 2 integers  $M$  and  $N$ , the height and width of the maze, respectively.  $M$  lines follow with  $N$  characters each. A '.' character indicates a passable cell. A '#' character indicates an impassable cell. A 'S' character indicates the starting point, and a 'E' character indicates the exit point. There will be exactly one 'S' character and one 'E' character per maze.

---

### Output

Output one line per test case containing the route Smart Chef should use, as a sequence of 'N', 'E', 'S', and 'W' characters, corresponding to the directions North, East, South, and West, respectively. Your route must not exceed 10000 moves in length.

---

## Scoring

Your score for each test case is the length of your route. Your total score for each test file is the sum of your scores on the individual test cases. Your overall score is the average of your scores on the individual test files.

---

## Sample Input

```
2
2
6 7
.####.##
..#...#
#S#.#..
.....#.
#.##..#
E...#..
6 7
.##.#.#
.....#
#.#.#.#
..E#S..
.##..#.
#...#..
3
```

5 6

...E#.

#.##..

...##.

.##.#.

.....S

5 6

##.###

#...##

E##..#

.##.#.

..S...

6 6

#..###

.#..##

.S.#.#

.#.#.E

#..#.#

##....

---

## Sample Output

SSSWENENNWWWSSE

ESSSWWWNNNNESEEEWNNEE

This sample output would score 16+22=38. Note that better scores may be possible.

---

## Test case generation

For each official test file,  $T$  is 10.  $K$  is chosen uniformly between 2 and 25, inclusive. An integer  $L$  is calculated as  $\text{floor}(45/\sqrt{K})$  and  $H$  as  $\text{floor}(70/\sqrt{K})$ . Then for each maze  $M$  and  $N$  are each chosen uniformly between  $L$  and  $H$ , inclusive.

Each maze is generated using a random walk algorithm. Initially all cells are marked as impassable. One cell, chosen randomly, is marked as passable, and a marker is placed in this cell. The following process repeated 10000000 times: The marker is moved to a random adjacent cell. Then, if the marker is in a cell adjacent to exactly one passable cell, the cell with the marker becomes passable.

Finally, the exit is chosen randomly among all passable cells with only one passable neighbor, and the start cell is chosen randomly among all remaining passable cells.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Can you do it better

## Can you do it better Problem Code: CYDB

---

Your task is to write a program which computes exactly the same result as Chef's programs below. He was kind enough to provide you with the same program written in three different programming languages. However, your solution should be much faster.

### C++

```
#include <iostream>
#include <string>
using namespace std;

int main() {
    string a, b, c;
    int na, nb, r;
    cin >> a;
    na = a.size();
    cin >> b;
    nb = b.size();
    r = nb - na;
    cout << na << " " << nb << " " << r;
}
```

```

nb = b.size();

cin >> c;

r = 0;

for (int i = 0; i < nb; i++) {

    for (int j = 1; j < min(na+1, nb-i+1); j++) {

        bool f = true;

        int d = 0;

        for (int k = 0; k < j; k++) {

            if (a[k] != b[i+k]) {

                if (c[k] == '1') { d += 1; }

                else { f = false; }

            }

        }

        if (f && d <= 2) { r = (r + j*j) % 1000000007; }

    }

}

cout << r << endl;

return 0;
}

```

**Java**

```

import java.util.Scanner;

public class Main {

```

```
static String a, b, c;

static int na, nb, r;

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    a = sc.next();

    na = a.length();

    b = sc.next();

    nb = b.length();

    c = sc.next();

    r = 0;

    for (int i = 0; i < nb; i++) {

        for (int j = 1; j < Math.min(na+1, nb-i+1); j++) {

            boolean f = true;

            int d = 0;

            for (int k = 0; k < j; k++) {

                if (a.charAt(k) != b.charAt(i+k)) {

                    if (c.charAt(k) == '1') { d += 1; }

                    else { f = false; }

                }

            }

            if (f && d <= 2) { r = (r + j*j) % 1000000007; }

        }

    }

}
```

```

    }

    System.out.println(r);

}

}

```

## Python

```

a = raw_input()

na = len(a)

b = raw_input()

nb = len(b)

c = raw_input()

r = 0

for i in range(nb):

    for j in range(1, min(na+1, nb-i+1)):

        f = True

        d = 0

        for k in range(j):

            if (a[k] != b[i+k]):

                if (c[k] == '1'): d +=1

            else: f = False

        if (f and d <= 2): r = (r + j*j) % 10000000007

print r

```

---

## Input

The strings A, B and C are provided in a compressed form to keep the input small enough. Sequence 00000 is encoded with letter 'a', 00001 with 'b' and so on until 'z'. 11010 is represented by 'A' and 11111 by 'F'.

The input consists of three lines of letters a-z, A-F. First line contains at most 200 characters and the second one at most 2 000 000. Third line has the same length as the first line and will contain at most 40 digits 1 when decompressed.

## Example

**Input:**

aBFn

ygFBg

bbdb

**Output:**

2990

## Explanation

After decompression we get strings 0000011011111101101, 110000011011111101100110 and 00001000010001100001. If we run chef's program with this decompressed input, we will get 2990 as a result.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Find a Subsequence

## Find a Subsequence

Problem Code: FINDSEQ

Given an array **A** of **N** integers **A[0], A[1], ..., A[N-1]** and a string **B** which is a permutation of "12345". You have to find a subsequence of 5 elements from **A** having **distinct values** such that their relative order is same as **B**.

Meaning that, if **(i<sub>0</sub>, i<sub>1</sub>, i<sub>2</sub>, i<sub>3</sub>, i<sub>4</sub>)** are the indexes of such a subsequence (**0 <= i<sub>0</sub> < i<sub>1</sub> < i<sub>2</sub> < i<sub>3</sub> < i<sub>4</sub> < N**) then:

- **A[i<sub>0</sub>]** is the **B[0]-th** smallest element among the 5 numbers.
- **A[i<sub>1</sub>]** is the **B[1]-th** smallest element among the 5 numbers.
- **A[i<sub>2</sub>]** is the **B[2]-th** smallest element among the 5 numbers.
- **A[i<sub>3</sub>]** is the **B[3]-th** smallest element among the 5 numbers.
- **A[i<sub>4</sub>]** is the **B[4]-th** smallest element among the 5 numbers.

## Input

The first line of input contains **T( $\leq 60$ )** which is the number of tests cases. The first line of each test case will be an integer **N ( $5 \leq N \leq 1000$ )** and a string **B** containing a permutation of "12345". Next line will contain **N** integers **A[0], A[1], ..., A[N-1]**. Each of them will be between **-10<sup>9</sup> and +10<sup>9</sup>** (inclusive).

---

## Output

For each test case output five space separated integers in ascending order which are the indexes **(i<sub>0</sub>, i<sub>1</sub>, i<sub>2</sub>, i<sub>3</sub>, i<sub>4</sub>)** of a five length subsequence described before. You may output any solution. If there is no solution just output "-1" without quotes.

---

## Example

### Input:

2

7 3 2 1 4 5

6 1 7 5 3 1 3 8 10

7 1 2 3 4 5

10 20 30 40 40 20 10

### Output:

0 2 3 5 6

-1

**Explanation of 1st sample:** {0, 2, 3, 5, 6} is a valid solution because the values of those indexes {6, 5, 3, 8, 10} will have ranks {3, 2, 1, 4, 5} after sorting. But {0, 2, 3, 4, 5} is not a solution as the values {6, 5, 3, 13, 8} will have ranks {3, 2, 1, 5, 4} after sorting.

**Explanation of 2nd sample:** There is no five length subsequence in the array that will have ranks {1, 2, 3, 4, 5} after sorting.

[Home](#) » [Compete](#) » [February 2012 Challenge](#) » Flight Distance

## Flight Distance Problem Code: FLYDIST

---

The chef is stuck at the airport since all flights are grounded because of a blizzard. He is trying to pass the time by analyzing a chart of distances between cities where the Spaghetti

airline is flying and he noticed something strange about it. Sometimes the distance of a direct flight from city A to city B is longer than a flight with intermediate stops in some other cities. This should be impossible under assumption that the flights always take the shortest path from start to destination. The chef is determined to fix this.

There are N cities numbered from 0 to N-1 and M flights between these cities. Each flight is characterized by its two destination points (it is always possible to fly in both directions) and the distance of the flight. The chef will change (increase or decrease) the distance of the flight  $i$  by  $d_i$  ( $d_i$  can be any rational number provided that the distance covered by this flight remains positive). After he makes all the changes, there should not exist a pair of cities A and B such that it is possible to fly through intermediate cities and reach the destination by covering less distance than with a direct flight. He is looking for a valid set of changes with minimum sum.

## Input

The number of cities N and the number of flights M are given on the first line of the input. The following M lines describe the flights with endpoints  $A_i$ ,  $B_i$  and the distance  $D_i$  covered by this flight. No flight will connect a city to itself and there will be at most one flight between any pair of cities.

## Output

Output the minimum sum of modifications as a reduced fraction "X/Y" - greatest common divisor of X and Y must be 1.

## Constraints

- $2 \leq N \leq 10$
- $1 \leq M \leq 45$
- $1 \leq D_i \leq 20$

## Example

**Input:**

4 4

0 1 1

1 2 1

2 3 1

0 3 8

**Output:**

5/1

## Explanation

One possible scenario is to decrease the length of flight (0,3) by 1, increase (0,1) by 3 and (1,2) by 1.

COOK19	<a href="#">February 2012 Cook-off</a>	19 Feb 2012 21:30:00	2 hours 30 minutes	549
--------	--	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [February 2012 Cook-off](#) » Daily Train

## Daily Train Problem Code: DAILY

A daily train consists of **N** cars. Let's consider one particular car. It has 54 places numbered consecutively from 1 to 54, some of which are already booked and some are still free. The places are numbered in the following fashion:

1	2	3	4	5	6	7	8	9
1	3	5	7	9	11	13	15	17
2	4	6	8	10	12	14	16	18
<hr/>								
54	53	52	51	50	49	48	47	46
45	44	43	42	41	40	39	38	37

The car is separated into 9 compartments of 6 places each, as shown in the picture. So, the 1st compartment consists of places 1, 2, 3, 4, 53 and 54, the 2nd compartment consists of places 5, 6, 7, 8, 51 and 52, and so on.

A group of **X** friends wants to buy tickets for free places, all of which are in one compartment (it's much funnier to travel together). You are given the information about free and booked places in each of the **N** cars. Find the number of ways to sell the friends exactly **X** tickets in one compartment (note that the order in which the tickets are sold doesn't matter).

---

## Input

The first line of the input contains two integers **X** and **N** ( $1 \leq X \leq 6$ ,  $1 \leq N \leq 10$ ) separated by a single space. Each of the following **N** lines contains the information about one car which is a string of length 54 consisting of '0' and '1'. The **i**-th character (numbered from 1) is '0' if place **i** in the corresponding car is free, and is '1' if place **i** is already booked.

---

## Output

Output just one integer -- the requested number of ways.

---

## Example

Input:

1 3

1001011000000101100000111110010011110010010111000101

00101000000010111100000000000001110101010111111010

01111001111000000101010010111000101111010001001111010

Output:

85

Input:

6 3

1001011000000101100000111110010011110010010111000101

00101000000010111100000000000001110101010111111010

011110011100000101010010111000101111010001001111010

**Output:**

1

**Input:**

3 2

000

000

**Output:**

360

**Explanation:**

In the first test case, any of the free places can be sold. In the second test case, the only free compartment in the train is compartment 3 in the first car (places 9, 10, 11, 12, 49 and 50 are all free). In the third test case, the train is still absolutely free; as there are 20 ways to sell 3 tickets in an empty compartment, the answer is  $2 * 9 * 20 = 360$ .

[Home](#) » [Compete](#) » [February 2012 Cook-off](#) » Equation Solver

## Equation Solver Problem Code: EQUATIO

$$x^*y = a + b^*lcm(x,y) + c^*gcd(x,y)$$

It's easy: you are to write a program which for given **a**, **b** and **c** finds the number of pairs of positive integers (**x**, **y**) satisfying this equation.

Here **\*** stands for multiplication, **gcd(x,y)** stands for the greatest common divisor of **x** and **y**, while **lcm(x,y)** stands for the least common multiple of **x** and **y**.

---

### Input

The first line of the input file contains one integer **T** -- the number of test cases (no more than 10). Each of the next **T** lines contains exactly three space-separated integers **a**, **b** and **c** ( $0 \leq a, b, c \leq 10^6$ ).

---

## Output

For each test case output one line containing the sought number of solutions to the equation. If there is an infinite number of solutions, output **-1** instead.

---

## Example

**Input:**

3

2 1 1

160 0 90

300 7 5

**Output:**

2

8

4

**Explanation:**

In the first test case, the only pairs are (2,4) and (4,2).

[Home](#) » [Compete](#) » [February 2012 Cook-off](#) » Careful Calculation

## Careful Calculation Problem Code: CAREFUL

---

You are given a single integer **N**. It's very large, so it's given as a product of several prime powers:  $N = p_1^{k_1} p_2^{k_2} \dots p_m^{k_m}$ .

Let's define  $\varphi(N)$  as [Euler's totient function](#) -- the number of positive integers less than or equal to  $N$  that are relatively prime to  $N$ .

Let  $N_1 = \varphi(N)$ . Let  $N_2 = \varphi(N_1)$ . Further, let  $N_X = \varphi(N_{X-1})$  for  $X > 2$ .

Your task is to find the smallest positive integer  $X$  such that  $N_X = 1$ . Only careful calculation might help... or will it be enough?

---

## Input

The first line of the input file contains one integer  $T$  -- the number of test cases (no more than 10). Each test case is described by a line containing one positive integer  $m$  followed by  $m$  lines, each containing two integers  $p_i$  and  $k_i$  ( $1 < p_i < 100000$ ,  $1 \leq k_i \leq 10^9$ ) separated by a single space. It's guaranteed that all  $p_i$  are pairwise distinct prime numbers in each test case (note that the upper limit on  $m$  can be determined from this constraint).

---

## Output

For each test case output just one line containing the required smallest positive integer  $X$ .

---

## Example

### Input:

```
1
2
2 2
3 1
```

### Output:

```
3
```

### Explanation:

In the example test case  $N = 2^2 * 3^1 = 12$ . Then  $N_1 = 4$ ,  $N_2 = 2$  and  $N_X = 1$  for  $X \geq 3$ , so the answer is 3.

## Bears and Bees Problem Code: BEARS

*It's a very funny thought that, if Bears were Bees,  
 They'd build their nests at the bottom of trees.  
 And that being so (if the Bees were Bears),  
 We shouldn't have to climb up all these stairs.*

Winnie-the-Pooh, a Complaining Song

Have you ever thought, when given an undirected graph in some problem, that it would be easier to solve it if the graph's edges were actually its vertices and the graph's vertices were its edges? This problem is right about this -- unfortunately, not about bears and bees (but if you want, you may think of vertices as of bears and of edges as of bees (or even vice versa)).

Suppose you're given an undirected graph  $G_0$  with  $N$  vertices and  $M$  edges. Let's perform a *simple transformation* on graph  $G_0$  to obtain graph  $G_1$  with  $M$  vertices so that each vertex of  $G_1$  corresponds to a unique edge of  $G_0$  and a pair of vertices in  $G_1$  is connected by a single edge if and only if the corresponding edges of  $G_0$  share a common vertex. Similarly, let's perform a simple transformation on graph  $G_1$  to obtain graph  $G_2$ , and let's perform a simple transformation on graph  $G_2$  to obtain graph  $G_3$ .

All you have to do is to output the number of vertices and edges in  $G_3$ .

---

### Input

The first line of the input file contains one integer  $T$  -- the number of test cases (no more than 10). Each test case is described by a line containing two integers  $N$  and  $M$  ( $1 \leq N, M \leq 1000$ ) followed by  $M$  lines, each containing two integers between 1 and  $N$ , inclusive, separated by a single space and describing an edge of graph  $G_0$ . It's guaranteed that each edge connects two distinct vertices and each pair of vertices is directly connected by at most one edge.

---

### Output

For each test case output just one line containing two integers -- the number of vertices and edges in  $G_3$ .

---

### Example

**Input:**

2

3 3

1 2

1 3

2 3

4 4

1 2

1 3

2 3

3 4

**Output:**

3 3

8 18

**Explanation:**

In the first test case the given graph is a "triangle". It's easy to see that a simple transformation on a triangle results in the same triangle (as it contains three pairwise connected vertices and three pairwise "connected" edges).

[Home](#) » [Compete](#) » [February 2012 Cook-off](#) » Authentication Failed

## Authentication Failed Problem Code: AUTHEN

Several days ago Chef decided to register at one of the programming sites. For registering he was asked to choose a nickname and a password. There was no problem with choosing a nickname ("Chef" is his favorite nickname), but choosing a password in a secure way seemed to be a real problem for Chef. Therefore, he decided to write a program which would generate the password of length **N** consisting of small Latin letters *a..z*. Then Chef successfully registered at the site and saved the password in a file (as it was too hard to remember).

Today Chef decided to visit the site once again. He entered his nickname, copied the password from the file... "Authentication failed!" was the answer. Trying to understand the reason of this, he noticed that the password in his file had length  $N+K$  instead of  $N!$  Sure enough of the source of the problem, Chef went straight to his young brother.

And Chef was right, it was his brother who had inserted  $K$  random small Latin letters at some random positions (possibly at the beginning or at the end) of the password. Chef's brother didn't remember what exactly changes he had made at all, but he promised that he had done nothing besides inserting letters.

As there is no other way to recover the password, Chef is now starting to remove every possible combination of  $K$  letters from the password trying to enter (when Chef obtains the same password as one of the previously entered passwords, he doesn't try to enter using this password again). Now the question is: what is the number of times Chef will receive "Authentication failed!" as the answer before successful entering in the worst case? As the answer might be quite large, output its remainder of division by 1009419529.

## Input

The first line of the input file contains one integer  $T$  -- the number of test cases (no more than 10). Each test case is described by a line containing two integers  $N$  and  $K$  ( $1 \leq N \leq 10000$ ,  $1 \leq K \leq 100$ ) separated by a single space, followed by a line containing a string of length  $N+K$  consisting of small Latin letters a..z.

## Output

For each test case output just one line containing the required number modulo 1009419529.

## Example

Input:

3

2 1

aaa

3 1

abcd

4 2

ababab

**Output:**

0  
3  
10

**Explanation:**

In the first test case, the password is definitely "aa". In the second test case, it can be "abc", "abd", "acd" or "bcd", so in the worst case Chef will guess the correct option from the fourth attempt, thus making 3 unsuccessful attempts.

MARCH12	<a href="#">March 2012 Challenge</a>	01 Mar 2012 15:00:00	10 days	1646
---------	--------------------------------------	-------------------------	---------	------

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Spoon in Matrix

## Spoon in Matrix Problem Code: SPOON

Chef recently saw the movie Matrix. He loved the movie overall but he didn't agree with some things in it. Particularly he didn't agree with the bald boy when he declared - [There is no spoon](#). Being a chef, he understands the importance of the spoon and realizes that the universe can't survive without it. Furthermore, he is sure there is a spoon; he saw it in his kitchen this morning. So he has set out to prove the bald boy is wrong and find a spoon in the matrix. He has even obtained a digital map already. Can you help him?

Formally you're given a matrix of lowercase and uppercase Latin letters. Your job is to find out if the word "Spoon" occurs somewhere in the matrix or not. A word is said to be occurred in the matrix if it is presented in some row from left to right or in some column from top to bottom. Note that match performed has to be case insensitive.

---

### Input

The first line of input contains a positive integer **T**, the number of test cases. After that **T** test cases follow. The first line of each test case contains two space separated integers **R** and **C**, the number of rows and the number of columns of the matrix **M** respectively. Thereafter **R** lines follow each containing **C** characters, the actual digital map itself.

## Output

For each test case print one line. If a "Spoon" is found in Matrix, output "There is a spoon!" else output "There is indeed no spoon!" (Quotes only for clarity).

---

## Constraints

$1 \leq T \leq 100$

$1 \leq R, C \leq 100$

---

## Sample Input

3

3 6

abDefb

bSpoon

NIKHil

6 6

aaaaaa

SSSSSS

xuisdP

oooooo

ioowoo

bdylan

6 5

bdfhj

cacac

opqrs

ddddd

india

yucky

## Sample Output

There is a spoon!

There is a spoon!

There is indeed no spoon!

## Explanation

In the first test case, "Spoon" occurs in the second row. In the second test case, "spOon" occurs in the last column.

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Free Shuttle Service

## Free Shuttle Service Problem Code: SHUTTLE

Chef has decided to arrange the free shuttle service for his employees. City of Bhiwani has a strange layout - all of its **N** shuttle boarding points are arranged in a circle, numbered from 1 to **N** in clockwise direction. Chef's restaurant is at boarding point number 1. There is a single ring road that runs over the circumference of this circle and connects all the boarding points. There are also **N - 1** different shuttle agencies available in Bhiwani.

For every different boarding points **A** and **B** there is exactly one shuttle that connects these points and it belongs to **K<sup>th</sup>** shuttle agency where **K** is the distance between **A** and **B** in clockwise direction, that is, there are exactly **K - 1** boarding points between points **A** and **B** in clockwise direction. Denote this shuttle as **(A, B)**. So if **N = 4**, first agency has shuttles (1,2), (2,3), (3,4), (4,1), second agency has shuttles (1,3), (2,4) and the shuttles of third agency are (1,4), (2,1), (3,2), (4,3). If the shuttle connects points **A** and **B**, it is possible to go from **A** to **B** as well as from **B** to **A** using this shuttle.

Chef is planning to make a contract with one of the agencies so that all of his employees are able to travel in shuttles of that agency for free. He therefore wants to choose such a shuttle agency so that people from any boarding point can reach his restaurant only using shuttles of the chosen agency possibly using some intermediate boarding points. Your task is to find how many such shuttle agencies are there.

## Input

First line contains an integer **T** denoting number of test cases. After that **T** lines follow each containing a single integer **N** denoting number of shuttle boarding points in Bhiwani.

---

## Output

For every test case, output the number of shuttle agencies our chef could choose.

---

## Constraints

$1 \leq T \leq 100$

$2 \leq N \leq 10000$

---

## Example

### Input:

```
3
2
3
4
```

### Output:

```
1
2
2
```

**Description:** In third case, there are 4 shuttle boarding points and there are  $4 - 1 = 3$  shuttle agencies. Using shuttles of only second agency, one can move between points (1,3) and points (2,4). So a person starting from point 2 can't reach restaurant using these shuttles. Each of the other two agencies connects all the points with the restaurant possibly through intermediate boarding points.

# Home Delivery Problem Code: HDELIVER

---

Chef has decided to start home delivery from his restaurant. He hopes that he will get a lot of orders for delivery, however there is a concern. He doesn't have enough work forces for all the deliveries. For this he has came up with an idea - he will group together all those orders which have to be delivered in nearby areas.

In particular, he has identified certain bidirectional roads which he calls as fast roads. They are short and usually traffic free, so the fast travel is possible along these roads. His plan is that he will send orders to locations **A** and **B** together if and only if it is possible to travel between **A** and **B** using only fast roads. Your task is, given the configuration of fast roads, identify which orders are to be sent together.

---

## Input

First line of input contains an integer **T**, the number of test cases. Then **T** test cases follow. First line of each test case contains two space separated integers **N** and **M**, denoting number of locations and the number of fast roads. Then **M** lines follow each containing two space separated integers **A** and **B**, denoting that there is a fast road between locations **A** and **B**. Assume that locations are indexed by numbers from 0 to **N-1**.

Next line contains an integer **Q** denoting the number of queries. Each of the next **Q** lines contain two integers **X** and **Y**. For each query you have to find out if orders meant for locations **X** and **Y** are to be sent together or not.

Note that there might be multiple fast roads between same pair of locations, also there might be a fast road that links a location to itself.

---

## Output

For each test case print **Q** lines - one for each query. Output "YO" if the orders are to be delivered together and "NO" otherwise (quotes for clarity).

---

## Constraints

$$1 \leq T \leq 100$$

$$1 \leq N \leq 100$$

$$1 \leq M \leq 1000$$

$$0 \leq A, B, X, Y \leq N-1$$

$$1 \leq Q \leq 3000$$

## Example

Input:

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

Output:

```
YO
NO
YO
```

---

## Warning!

There are large input and output files in this problem. Make sure you use fast enough I/O methods.

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Longest Weird Subsequence

## Longest Weird Subsequence Problem Code: LWS

---

Finding the longest increasing subsequence is an old and well-known problem now. Here you will have to do something similar. You need to find the longest *weird* subsequence (LWS) of the given string. The subsequence is called *weird* if it can be split into two disjoint subsequences, one of which is non-decreasing and the other one is non-increasing.

Just for clarity, by subsequence of the given string **S** we mean any string that can be obtained from **S** by erasing from it zero or more characters. So empty string is a subsequence of any string and any string is a subsequence of itself. Further, note that we consider only strings composed of lowercase Latin letters and these letters compared by their ASCII codes. So, for example, 'a' is smaller than 'b' and 'p' is larger than 'h'.

Now let's consider some example. Let **S**="aabcazcczba". Then "abczz" is its some non-decreasing subsequence, "zccb" is its some non-increasing subsequence and "aabczcczba" is its some weird subsequence since it can be split into non-decreasing subsequence "aabzz" and non-increasing subsequence "cccba": "AABcZccZba" (first subsequence is shown by capital letters just for clarity).

## Input

The first line contains a single positive integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains a non-empty string **S** composed of lowercase Latin letters.

## Output

For every test case, output the length of the LWS of the given string.

## Constraints

$1 \leq T \leq 10$

$1 \leq \text{length of } S \leq 2000$

## Example

### Input

3

abc

cbazyzabc

ddaabbaacc

**Output**

3

6

10

**Explanation**

**First case:** The string itself is LWS since it can be split into non-decreasing subsequence "**abc**" and non-increasing empty subsequence.

**Second case:** One of the possible LWS is "**cbaabc**" since it can be split as "**cbaABC**". Here we indicate by capital letters non-decreasing subsequence and by lowercase letters non-increasing one. Other possible LWS's are "**cbaZZa**", "**AzyaBC**".

**Third case:** Here the desired splitting is "**ddAABBaaCC**".

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » [Lucky Number](#)

## Lucky Number Problem Code: LUCKY2

Chef loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47**, **744**, **4** are lucky and **5**, **17**, **467** are not.

Let **F(X)** equals to the number of lucky digits in decimal representation of **X**. Chef wants to know the number of such integers **X**, that **L ≤ X ≤ R** and **F(X)** is a lucky number. Help him and calculate that number modulo **10<sup>9</sup>+7**.

**Input**

First line contains one integer **T**, the number of test cases. Each of the following **T** lines contains two space separated positive integers **L** and **R**.

**Output**

For each of the **T** test cases print one integer, the number of such **X**, that **L ≤ X ≤ R** and **F(X)** is a lucky number, modulo **1000000007**.

**Constraints**

$1 \leq T \leq 10$

$1 \leq L \leq R \leq 10^{1000}$

---

## Example

**Input:**

4

1 100

1 10000

1 100000

4444 4447

**Output:**

0

16

640

2

---

## Notes

First test case: of course, any number of less than 4 digits can't contain lucky number of lucky digits, so the answer is 0.

Second test case: 16 required numbers are **4444 4447 4474 4477 4744 4747 4774 4777 7444 7447 7474 7477 7744 7747 7774 7777**.

Third test case: there are 640 required lucky numbers. Some of them are **4474, 14747, 41474, 77277, 44407, 74749**.

Fourth test case: the only two required numbers are **4444** and **4447**.

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Cosine Partition Function

# Cosine Partition Function

Problem Code: PARCOS

Chef Ciel is participating an arithmetic contest now. Why? Because of the top prize for the contest, limited edition chopsticks.

She must calculate the values  $f(M, N, X)$  of function named *cosine partition function* to be the first place. The cosine partition function  $f(M, N, X)$  is defined by

$$\sum_{\substack{k_1 + k_2 + \dots + k_M = N \\ k_1, \dots, k_M \text{ are nonnegative integers}}} \cos(k_1 X/N) \cos(k_2 X/N) \dots \cos(k_M X/N)$$

Examples are following:

$$f(1, 3, X) = \cos(3X/3) = \cos(X)$$

$$f(2, 3, X) = \cos(0/3) \cos(3X/3) + \cos(X/3) \cos(2X/3) + \cos(2X/3) \cos(X/3) + \cos(3X/3) \cos(0/3) = 2 \cos(X/3) \cos(2X/3) + 2 \cos(X)$$

$$f(3, 1, X) = \cos(0) \cos(0) \cos(X) + \cos(0) \cos(X) \cos(0) + \cos(X) \cos(0) \cos(0) = 3 \cos(X)$$

Ciel is a great chef, however she is not good at arithmetic. For given  $M$ ,  $N$  and  $X$ , your work is calculating the value of  $f(M, N, X)$ .

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case has 2 integers  $M$ ,  $N$  and a real number  $X$ .  $X$  has at most two digits after the decimal point.

## Output

For each test case, print the value of  $f(M, N, X)$ . This value must have an absolute or relative error no more than  $10^{-6}$ . You can safely assume the magnitude of the answer is at most  $10^{300}$ .

## Constraints

$1 \leq T \leq 50$   
 $1 \leq M \leq 1000000000 (10^9)$   
 $1 \leq N \leq 30$   
 $0 \leq X \leq 1$

---

## Sample Input

4

1 3 1.0

2 3 0

3 0 1 0.1

1 1 0.5

---

## Sample Output

0.5403023059

4.0000000000

29.8501249583

0.8775825619

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Xor it

## Xor it Problem Code: XOR

Chef has given you a sequence  $A[1], A[2], \dots, A[N]$  composed of  $N$  nonnegative integer numbers. Then, for each pair  $(i; j)$  such that  $1 \leq i < j \leq N$ , we have written a number that equals to  $A[i] \text{ xor } A[j]$  (xor is exclusive or, "xor" in Pascal, "^" in C++). Thus, we have obtained  $N*(N-1)/2$  numbers. Your task is to find  $K$  minimal numbers among them.

---

### Input

The first line of the input contains two space separated integers  $N$  and  $K$ . Each of the next  $N$  lines contains one integer,  $i^{\text{th}}$  line contains number  $A[i]$ .

---

### Output

In the only line of output print space separated sequence of  $K$  numbers, the answer to the problem. Numbers should be in non-decreasing order.

---

### Constraints

$2 \leq N \leq 100000$

$1 \leq K \leq \min\{250000, N*(N-1)/2\}$

$0 \leq A[i] < 2^{31}$

---

## Example

Input:

4 5

1

1

3

4

Output:

0 2 2 5 5

---

## Explanation

In the sample input we have **4** numbers: **1, 1, 3, 4**. Therefore, there are **(4\*3)/2 = 6** pairwise XOR's.

These XOR's are:

$1 \text{ xor } 1 = 0$  ( $A[1] \text{ xor } A[2]$ )

$1 \text{ xor } 3 = 2$  ( $A[1] \text{ xor } A[3]$ )

$1 \text{ xor } 4 = 5$  ( $A[1] \text{ xor } A[4]$ )

$1 \text{ xor } 3 = 2$  ( $A[2] \text{ xor } A[3]$ )

$1 \text{ xor } 4 = 5$  ( $A[2] \text{ xor } A[4]$ )

$3 \text{ xor } 4 = 7$  ( $A[3] \text{ xor } A[4]$ )

If we sort these numbers we will obtain: **0, 2, 2, 5, 5, 7**. The first **5** minimal numbers are: **0, 2, 2, 5, 5**.

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Tom And Jerry

## Tom And Jerry

Problem Code: **TOMJERRY**

---

Note: The test data for this problem is not final yet. More tests may be uploaded later!

Here comes the famous story of Tom and Jerry again!

One day, Jerry is on his way back home. His home has lots of doors scattered at different cells in a rectangular board of size **M x N**. Entering any door leads Jerry to his home. There are obstacles in some cells of the board and Jerry cannot step on these cells. In each step Jerry will move to one of the four neighboring cells. Of course, Jerry is really clever, so he always goes home along a shortest path. If there are multiple choices, Jerry will prioritize his actions in the following order: moving upward, moving rightward, moving leftward and moving downward.

Needless to say that Tom does not want Jerry to get back home. And Tom is very clever too. In each step Tom will put an obstacle in an empty cell, and of course this empty cell cannot have Jerry standing in it. He cannot put an obstacle on top of an entrance to Jerry's house. Also, in each step Tom places exactly one obstacle. Tom wants to put obstacles so that there's no way for Jerry to come back home!

Write a program to help Tom put as few obstacles as possible to achieve his (cruel) objective! Given that, in the beginning of each step Tom will put an obstacle first and then Jerry makes a move. You may sympathize with poor Jerry, but nonetheless it is your job to write this program!

---

## Input

The first line contains two space separated positive integers **M** and **N**, the sizes of the board.

Each of the next **M** lines contains **N** characters, representing the rectangular board. Each character can be '.' - an empty cell, '0' (zero) - a door to Jerry's home, '#' - an obstacle or '\*' - the initial position of Jerry. The input is guaranteed to have a single '\*' on the board. There will be at least one door to Jerry's home. There may or may not be any obstacles in the beginning.

You may assume that in each test case Jerry has a way to escape in the beginning; and there is a way for Tom to put obstacles such that Tom achieves his objective within 100 steps.

---

## Output

The first line should contains a non-negative integer **K**, the number of obstacles that Tom will put. If **K** is more than 200, the judge will return Wrong Answer.

Each of the next **K** lines should contain two space separated positive integers **x** and **y** ( $1 \leq x \leq M$ ,  $1 \leq y \leq N$ ), the coordinates of the obstacles that Tom will put in the current step. The rows of the board are numbered by numbers from 1 to **M** from top to bottom. The columns of the board are numbered by numbers from 1 to **N** from left to right. If at some move Tom will put obstacle outside the board or at already occupied cell (by Jerry or by door or by another obstacle), the judge will return Wrong Answer.

---

## Scoring

Your score for each test case is  $K^2$  if Jerry does not have any route to escape after placing the obstacles. If in the end Jerry can still find his way back home, you will get 2000000 (2e6) score for the corresponding test case. The total score is the sum of score for all test cases (or files).

Your objective, of course, is to minimize the total score.

---

## Constraints

$1 \leq M, N \leq 50$

---

## Example

### Input

3 3

..0

...

\*..

### Output

3

2 2

1 1

3 2

This sample output would score  $3 \times 3 = 9$ . Note that better scores may be possible.

## Evil Book Problem Code: EVILBOOK

Chef Ciel got enormous cooking power from the *Evil Book*. Instead, Ciel must devote 666 magical power to the Evil Book.

Other than Ciel, there are  $N$  chefs in the world. The  $i$ -th chef has  $C_i$  cooking power and  $M_i$  magical power. Ciel may get  $M_i$  magical power by defeating the  $i$ -th chef in a cooking battle, which needs  $C_i$  effort. After that, the  $i$ -th chef's magical power  $M_i$  becomes 0.

The Evil Book can help her, if she gives  $X$  magical power to the Evil Book. That is, by consuming  $X$  magical power, she can choose some  $i$  and multiply by  $1/3$  both the  $i$ -th chef's cooking power  $C_i$  and his magical power  $M_i$ . Note that she can't take the help of the Evil Book unless she has at least  $X$  magical power.

Firstly, Ciel has no magical power. What's the minimum total effort for collecting at least 666 magical power?

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case contains  $N+1$  lines. The first line for each test case has 2 integers  $N$  and  $X$ . The  $(i+1)$ -th line for each test case has 2 integers  $C_i$  and  $M_i$ .

### Output

For each test case, print the minimum total effort (rounded to the nearest integer) for collecting at least 666 magical power. If it is impossible to collect 666 magical power, print "impossible" without quotes.

### Constraints

$1 \leq T \leq 5$   
 $1 \leq N \leq 10$   
 $10 \leq X < 666$   
 $0 \leq C_i \leq 1000000000 (10^9)$   
 $0 \leq M_i \leq 1000000000 (10^9)$

### Sample Input

1 30

110 500

200 700

190 800

2 665

1 250

2 415

2 10

1000000000 1000000000

1000000000 1000000000

2 10

1000000000 1000000000

0 130

---

## Sample Output

168

impossible

1000000000

1882

---

## Output details

In the first case, the optimal way is the following.

Initially, Ciel's magical power = 0, Total effort = 0

$C_1 = 1, M_1 = 30$

$C_2 = 110, M_2 = 500$

$C_3 = 200, M_3 = 700$

$C_4 = 190, M_4 = 800$

After defeating the 1st chef:

Ciel's magical power = 30, Total effort = 1

$C_1 = 1, M_1 = 0$

$C_2 = 110, M_2 = 500$

$C_3 = 200, M_3 = 700$

$C_4 = 190, M_4 = 800$

After using 3 helps of the Evil Book ( $i = 2, 3, 4$ ):

Ciel's magical power = 0, Total effort = 1

$C_1 = 1, M_1 = 0$

$C_2 = 36+2/3, M_2 = 166+2/3$

$C_3 = 66+2/3, M_3 = 233+1/3$

$C_4 = 63+1/3, M_4 = 266+2/3$

After defeating the 2nd, 3rd and 4th chefs:

Ciel's magical power =  $666+2/3$ , Total effort =  $167+2/3$  ( $1 + 36+2/3 + 66+2/3 + 63+1/3$ )

$C_1 = 10, M_1 = 0$

$C_2 = 36+2/3, M_2 = 0$

$C_3 = 66+2/3, M_3 = 0$

$C_4 = 63+1/3, M_4 = 0$

In the fourth case, Ciel should defeat the 2nd chef at first. Then, after using 12 helps of the Evil Book, the 1st chef's cooking power and magical power become  $10^9 / 3^{12} = 1881.67642\dots$

[Home](#) » [Compete](#) » [March 2012 Challenge](#) » Ciel and Earthquake

## Ciel and Earthquake Problem Code: CIELQUAK

The country in which Chef Ciel lives has many earthquakes. Since Ciel's restaurant is far away from an evacuation center, Ciel is afraid of earthquakes. So, Ciel would like to know the probability that Ciel can reach from Ciel's restaurant to the evacuation center when an earthquake occurs. Your task is calculating the probability under the following assumptions.

Ciel's city has  $R \times C$  junctions ( $i, j$ ) for  $1 \leq i \leq R, 1 \leq j \leq C$ . There is a two-way road between the junctions  $(r_1, c_1)$  and  $(r_2, c_2)$  if and only if  $|r_1 - r_2| + |c_1 - c_2| = 1$ . When an earthquake occurs, each road is destroyed with probability  $p$ , and these events are statistically independent of each other. Ciel's restaurant is in the junction  $(1, 1)$ , and the evacuation center is in the junction  $(R, C)$ . Ciel only considers a big earthquake, so you can assume that  $p$  is not less than 0.1.

---

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case has 3 numbers  $R, C$  and  $p$ , where  $R$  and  $C$  are integers.

---

## Output

For each test case output the required probability. Your answer must have an absolute error no more than 0.000001 ( $10^{-6}$ ).

---

## Constraints

$1 \leq T \leq 50$   
 $1 \leq R \leq 8$   
 $1 \leq C \leq 10000000000000000000 (10^{18})$   
 $0.1 \leq p \leq 1$   
p has at most four digits after the decimal point.

---

## Sample Input

5  
2 2 0.5  
3 2 0.7  
2 3 0.7  
1 1 0.2  
7 7 0.8

---

## Sample Output

0.4375  
0.0768204  
0.0768204  
1  
0.000003962379607

COOK20	March Cook-Off 2012	18 Mar 2012 21:30:00	2 hours 30 minutes	619
--------	------------------------	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [March Cook-Off 2012](#) » Ciel Numbers II

# Ciel Numbers II

Problem Code: CIELNUM2

Recently, chef Ciel often hears about *lucky numbers*.

---

*Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits 4 and 7. For example, numbers 47, 744, 4 are lucky and 5, 17, 467 are not.*

Ciel decides to make *Ciel numbers*. As you know, Ciel likes the digit 8 very much. And then, Ciel likes the digits 5 and 3. So Ciel defines Ciel numbers as the positive integers  $k$  such that  $d(k, 8) \geq d(k, 5) \geq d(k, 3)$  and  $d(k, i) = 0$  for all  $i = 0, 1, 2, 4, 6, 7, 9$ , where  $d(k, i)$  denotes the number of the digit  $i$  in the decimal representation of the integer  $k$ . For example, the first few Ciel numbers are 8, 58, 85, 88, 358, 385, 538, 583, 588, 835, 853, 858, 885, 888, ....

Ciel's restaurant has  $N$  menus. And Ciel want to know how many menus have Ciel numbers as their price. Your task is to find it.

---

## Input

The first line contains an integer  $N$ . Then  $N$  lines follow. Each line has the name  $S_i$  of the menu and its price  $P_i$  separated by a single space.

---

## Output

Print the number of menus whose prices are one of Ciel numbers.

---

## Constraints

$1 \leq N \leq 1000$

$1 \leq |S_i| \leq 100$ , where  $|S_i|$  denotes the length of  $S_i$

Each letter of  $S_i$  is either an alphabetical letter or a digit or a single quotation mark or a space.

$1 \leq P_i < 1000000$  ( $10^6$ )

$P_i$  contains no leading zeros.

---

## Sample Input

6

milk 58

Ciel's Drink 80

The curry 2nd edition 888888

rice omelet 85855

unagi 1

The first and last letters can be a space 358

---

## Sample Output

3

---

### Output details

58 and 888888 and 358 are Ciel numbers. 80 and 85855 and 1 are not Ciel numbers.

---

### Notes

Different operating systems have different ways of representing a newline; do not assume one particular way will be used.

[Home](#) » [Compete](#) » [March Cook-Off 2012](#) » Ciel Numbers I

## Ciel Numbers I Problem Code: CIELNUM1

---

Recently, chef Ciel often hears about *lucky numbers*.

---

*Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits 4 and 7. For example, numbers 47, 744, 4 are lucky and 5, 17, 467 are not.*

---

Ciel decides to make *Ciel numbers*. As you know, Ciel likes the digit 8 very much. And then, Ciel likes the digits 5 and 3. So Ciel defines Ciel numbers as the positive integers  $k$  such that  $d(k, 8) \geq d(k, 5) \geq d(k, 3)$  and  $d(k, i) = 0$  for all  $i = 0, 1, 2, 4, 6, 7, 9$ , where  $d(k, i)$  denotes the number of the digit  $i$  in the decimal representation of the integer  $k$ . For example, the first few Ciel numbers are 8, 58, 85, 88, 358, 385, 538, 583, 588, 835, 853, 858, 885, 888, ....

Ciel would like to know about Ciel numbers. Your task is to find the first 50000 Ciel numbers.

---

### Input

This problem has no inputs.

---

### Output

Print the first 50000 Ciel numbers in order of increasing.

---

## Sample Output

8

58

85

88

... (49996 lines)

[Home](#) » [Compete](#) » [March Cook-Off 2012](#) » Ciel and a new island

## Ciel and a new island Problem Code: CIELLAND

---

Chef Ciel develops a new island with her restaurants. In the island, Ciel intends to built N restaurants, and the coordinate of the i-th restaurant will be  $(x_i, y_i)$ . In addition, Ciel is going to create K roads, whose location is not decided yet. Each road must be a infinitely long straight line.

Let  $d_i$  be the distance between the i-th restaurant and the nearest road from the i-th restaurant. Ciel would like to create K roads which minimize  $\max(d_1, d_2, \dots, d_N)$ . Your task is to calculate the minimal value of  $\max(d_1, d_2, \dots, d_N)$ .

---

### Input

The first line contains an integer T, the number of test cases in the input file. Then T test cases follow. The next line contains 2 integers N and K. Then next N lines contain 2 integers  $x_i$  and  $y_i$ .

---

### Output

For each test case, print the minimal value of  $\max(d_1, d_2, \dots, d_N)$ . Your answer must have an absolute error no more than  $0.000001 (10^{-6})$ .

---

### Constraints

$1 \leq K \leq N \leq 13$

$-100 \leq x_i, y_i \leq 100$

If  $i \neq j$ , then  $(x_i, y_i) \neq (x_j, y_j)$

The sum of N in one input file does not exceed 30.

---

### Sample Input

2  
8 1  
20 1  
20 -1  
-20 1  
-20 -1

1 20  
1 -20  
-1 20  
-1 -20

8 2  
20 1  
20 -1  
-20 1  
-20 -1  
1 20  
1 -20  
-1 20  
-1 -20

---

### Sample Output

14.849242404868

0.707106781137

## Ciel and Genjiko Problem Code: CIELGAME

*Genjiko* is one of the Japanese traditional games. Chef Ciel will take place a variant version of genjiko in her restaurant.

The rules of the game are the following:

- The challenger must be blindfolded.
- Ciel feeds the challenger one of the  $N$  foods menus of her restaurant,  $M$  times.
- The challenger must answer whether the  $i$ -th food and the  $j$ -th food are same or not, for all  $1 \leq i < j \leq M$ .
- For making the game difficult, Ciel cannot use the same menu for both the  $i$ -th menu and the  $(i+j)$ -th menu for all  $1 \leq j \leq K$ . (The challenger knows this)

Ciel wonders how many patterns can be the challenger's answer. Your task is calculating the number of the patterns modulo  $1000000009$  ( $10^9+9$ ).

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case has 3 integers  $N$ ,  $M$  and  $K$ .

### Output

Print the number of the patterns of the possible answers modulo  $1000000009$  ( $10^9+9$ ).

### Constraints

$1 \leq T \leq 100$   
 $1 \leq N \leq 200$   
 $1 \leq M \leq 1000000000$  ( $10^9$ )  
 $0 \leq K \leq 200$

### Sample Input

5

200 5 0

2 3 1

2 3 0

3 3 1

1 1000000000 200

## Sample Output

52

1

4

2

0

## Output details

The first case is the same as genjiko. In the traditional game genjiko, we make connections between each possible answer and the name of a book of *The Tale of Genji*. However The Tale of Genji is composed of 54 books, so the names of the first book and the last book are unused.

Let the letters A, B, C, ... denote the N menus, and let "ABA" mean that Ciel uses A for the 1st food, and uses B for the 2nd food, and uses A for the 3rd food. In the second case, Ciel cannot use the same menu consecutively. So Ciel can use only 2 patterns "ABA" and "BAB". However there is only one possible answer [1st  $\neq$  2nd, 1st = 3rd, 2nd  $\neq$  3rd].

In the third case, Ciel can use 8 patterns "AAA", "AAB", "ABA", "ABB", "BAA", "BAB", "BBA" and "BBB". And there are four possible answers [1st = 2nd, 1st = 3rd, 2nd = 3rd], [1st = 2nd, 1st  $\neq$  3rd, 2nd  $\neq$  3rd], [1st  $\neq$  2nd, 1st = 3rd, 2nd  $\neq$  3rd] and [1st  $\neq$  2nd, 1st  $\neq$  3rd, 2nd  $\neq$  3rd].

In the fifth case, Ciel can use no patterns.

[Home](#) » [Compete](#) » [March Cook-Off 2012](#) » Ciel and Communications System

# Ciel and Communications System

Problem**Code: CIELWEB**

Ciel has N restaurants on a long straight road. The coordinate of the i-th restaurant is  $X_i$ , and  $X_1 \leq X_2 \leq \dots \leq X_N$ . Note that more than one restaurant can have the same coordinate. (They may be in the same building.)

When the  $i$ -th restaurant communicates with the  $j$ -th restaurant ( $i < j$ ), a bit strange method is used. In the method, some towers are used. Each tower  $t$  has four parameters  $Y[t]$ ,  $R[t]$ ,  $C[t]$  and  $T[t]$ . Here  $Y[t]$  denotes the coordinate of the tower,  $R[t]$  denotes the range of the tower,  $C[t]$  denotes the cost for using the tower if the distance between the tower and the restaurant is 1, and  $T[t]$  denotes the type of the tower. The  $i$ -th restaurant can use the tower  $t$  if and only if  $|X_i - Y[t]| \leq R[t]$ . Here  $T[t]$  is represented as an integer  $1 \leq T[t] \leq M$ , where  $M$  is the number of types.

For some security reasons, the  $k$ -th restaurant can communicate with only the  $(k \pm 1)$ -th restaurant directly. So, it is necessary that the  $k$ -th restaurant communicates with the  $(k+1)$ -th restaurant for all  $i \leq k < j$ . For each communication between the  $k$ -th and  $(k+1)$ -th restaurants, each restaurant chooses the newest available tower. Let  $\text{TOWER}_k$  be the newest available tower for the  $k$ -th restaurant, and let  $\text{TYPE}[k]$  be the type of  $\text{TOWER}_k$ . The cost for the communication is  $C[\text{TOWER}_k] * |Y[\text{TOWER}_k] - X_k| + C[\text{TOWER}_{k+1}] * |Y[\text{TOWER}_{k+1}] - X_{k+1}|$ . If  $\text{TOWER}_k \neq \text{TOWER}_{k+1}$ , then additionally the cost  $U_{\text{TYPE}[k], \text{TYPE}[k+1]}$  is needed. The total cost of the communication between the  $i$ -th and  $j$ -th restaurants is the sum of the cost of the communication between the  $k$ -th and  $(k+1)$ -th restaurants over all  $i \leq k < j$ . If there should exist some  $i \leq k \leq j$  such that the  $k$ -th restaurant has no available towers, the communication is impossible.

Now Ciel has the history of communications and constructions of towers. Your task is to calculate the cost for each communication in the history. In the beginning of the history, you should assume that there are no towers.

## Input

The first line contains an integer  $N$ ,  $M$  and  $Q$ , where  $Q$  denotes the number of the lines of the history. Then next line contains  $N$  integers, where the  $i$ -th integer denotes  $X_i$ . Then next  $M$  lines contain  $M$  integers, where the  $j$ -th integer of the  $i$ -th line denotes  $U_{i,j}$ . Then next  $Q$  lines denote the history of communications and constructions. The history is composed of the two types of lines. The line "1 A B" (without quotes) means that the  $A$ -th restaurant communicates with the  $B$ -th restaurant. The line "2 Y R C T" (without quotes) means that the tower whose parameters are  $Y$ ,  $R$ ,  $C$  and  $T$  is built.

## Output

For each communication, print the cost. If it is impossible, print "impossible" without quotes.

## Constraints

$2 \leq N \leq 100000 (10^5)$   
 $1 \leq M \leq 50$   
 $1 \leq Q \leq 50000 (5 * 10^4)$   
 $0 \leq X_1 \leq X_2 \leq \dots \leq X_N \leq 1000000000 (10^9)$   
 $0 \leq U_{i,j} \leq 1000000000 (10^9)$

$1 \leq A < B \leq N$   
 $0 \leq Y \leq 10000000000 (10^9)$   
 $1 \leq R \leq 10000000000 (10^9)$   
 $1 \leq C \leq 10000 (10^4)$   
 $1 \leq T \leq M$   
 $U_{i,j} = U_{j,i}$

---

## Sample Input

5 2 6

1 3 5 7 10

1 3

3 1

1 1 2

2 5 5 1 1

1 1 2

1 1 4

2 4 1 2 1

1 1 4

---

## Sample Output

impossible

6

10

16

APRIL12

[April Challenge 2012](#)

01 Apr 2012  
15:00:00

10 days

2405

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Double Strings

## Double Strings Problem Code: DOUBLE

---

The **palindrome** is a string that can be read the same way from left to right and from right to left. For example, strings "aaaaa", "1221", "bbaabb" are **palindromes**, however the string "chef" is **not a palindrome** because if we read it from right to left, we will obtain "fehc" that is not the same as "chef".

We call a string a "**double string**" if it has an even length and the first half of this string is equal to the second half of this string, for example "abab" is a **double string** because the first half "ab" is equal to the second half "ab", however the string "abba" is **not a double string** because the first half "ab" is not equal to the second half "ba". The empty string "" is a **double string**, and its length is **0**.

Chef doesn't like palindromes, however he likes "double strings". He often likes to change the order of letters in some palindrome and sometimes to remove some symbols from it.

Now he wonders: if a **palindrome** of length **N** is given, what is the maximal possible number of characters in a "double string" that can be obtained by removing and changing the order of symbols in it?

---

### Input

Several test cases are given.

The first line of the sample input contains an integer **T** - the number of test cases.

Then, **T** lines follow.

Each line consists of a single integer **N** - the length of a palindrome.

---

### Output

For each test case output a single integer - answer to the problem.

---

### Constraints

- $1 \leq T \leq 10000$
  - $1 \leq N \leq 1000000000$
- 

### Example

**Input:**

2

2

4

**Output:**

2

4

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Fit to Play

## Fit to Play Problem Code: PLAYFIT

*Who's interested in football?*

Rayne Wooney has been one of the top players for his football club for the last few years. But unfortunately, he got injured during a game a few months back and has been out of play ever since.

He's got proper treatment and is eager to go out and play for his team again. Before doing that, he has to prove to his fitness to the coach and manager of the team. Rayne has been playing practice matches for the past few days. He's played N practice matches in all.

He wants to convince the coach and the manager that he's improved over time and that his injury no longer affects his game. To increase his chances of getting back into the team, he's decided to show them stats of any 2 of his practice games. The coach and manager will look into the goals scored in both the games and see how much he's improved. If the number of goals scored in the 2nd game(the game which took place later) is greater than that in 1st, then he has a chance of getting in. Tell Rayne what is the maximum improvement in terms of goal difference that he can show to maximize his chances of getting into the team. If he hasn't improved over time, he's not fit to play. Scoring equal number of goals in 2 matches will not be considered an improvement. Also, he will be declared unfit if he doesn't have enough matches to show an improvement.

**Input:**

The first line of the input contains a single integer T, the number of test cases. Each test case begins with a single integer N, the number of practice matches Rayne has played.

The next line contains N integers. The ith integer,  $g_i$ , on this line represents the number of goals Rayne scored in his ith practice match. The matches are given in chronological order i.e.  $j > i$  means match number j took place after match number i.

**Output:**

For each test case output a single line containing the maximum goal difference that Rayne can show to his coach and manager. If he's not fit yet, print "UNFIT".

---

## Constraints:

$1 \leq T \leq 10$   
 $1 \leq N \leq 100000$   
 $0 \leq g_i \leq 1000000$  (Well, Rayne's a legend! You can expect him to score so many goals!)

---

## Example:

### Input:

```
3
6
3 7 1 4 2 4
5
5 4 3 2 1
5
4 3 2 2 3
```

### Output:

```
4
UNFIT
1
```

### Explanation:

In the first test case, Rayne can choose the first and second game. Thus he gets a difference of  $7-3=4$  goals. Any other pair would give him a lower improvement. In the second test case, Rayne has not been improving in any match. Thus he's declared UNFIT.

**Note:** Large input data. Use faster I/O methods. Prefer scanf,printf over cin/cout.

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Stacking Pancakes

## Stacking Pancakes Problem Code: PANSTACK

Chef is good at making pancakes. Generally he gets requests to serve **N** pancakes at once. He serves them in the form of a stack. A pancake can be treated as a circular disk with some radius.

Chef needs to take care that when he places a pancake on the top of the stack the radius of the pancake should not exceed the radius of the largest pancake in the stack by more than 1. Additionally all radii should be positive integers, and the bottom most pancake should have its radius as 1. Chef wants you to find out in how many ways can he create a stack containing **N** pancakes.

### Input

First line of the input contains **T (T <= 1000)** denoting the number of test cases.

**T** lines follow each containing a single integer **N (1 <= N <= 1000)** denoting the size of the required stack.

## Output

For each case the output should be a single integer representing the number of ways a stack of size **N** can be created. As the answer can be large print it modulo **1000000007**.

## Example

### Input

```
2
1
2
```

### Output

```
1
2
```

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Greatest Dumpling Fight

## Greatest Dumpling Fight Problem Code: DUMPLING

Chef Shifu and Chef Po are participating in the Greatest Dumpling Fight of 2012. Of course, Masterchef Oogway has formed the rules of the fight.

There is a long horizontal rope of infinite length with a center point P. Initially both Chef Shifu and Chef Po will stand on the center P of the rope facing each other. Don't worry, the rope is thick enough to hold Chef Po and Chef Shifu at the same place and at the same time. Chef Shifu can jump either **A** or **B** units to the left or right in one move. Chef Po can jump either **C** or **D** units to the left or right in one move.

Masterchef Oogway wants to place exactly one dumpling on the rope such that both Chef Shifu and Chef Po will be able to reach it independently in one or more moves. Also the dumpling can be placed at most **K** units away from the center of the rope. Masterchef Oogway will let you watch the fight if you can decide the number of possible positions on the rope to place the dumpling.

---

### Input

First line contains **T**, the number of test cases. Each of the next **T** lines contains five positive integers, **A B C D K**.

$1 \leq T \leq 1000$

$1 \leq A, B, C, D, K \leq 10^{18}$

---

## Output

For each test case, output on a newline, the number of possible positions to place the dumpling on the rope.

---

## Example

**Input:**

3

2 4 3 6 7

1 2 4 5 1

10 12 3 9 16

**Output:**

3

3

5

**Explanation:**

For the second case,

Chef Po jumps 2 units to the right and then 1 unit to the left.

Chef Shifu jumps 5 units to the right and then 4 units to the left to reach 1 unit right from the center.

Chef Po jumps 2 units to the left and then 1 unit to the right.

Chef Shifu jumps 5 units to the left and then 4 units to the right to reach 1 unit left from the center.

Dumpling can also be placed at the center as a chef can reach it in 2 moves.

Thus, there are three different positions at most 1 unit away from the center that are reachable by both the chefs in one or more moves.

[Home](#) » [Compete](#) » [April Challenge 2012](#) » [Similar Graphs](#)

## Similar Graphs Problem Code: SIMGRAPH

Chef recently developed an affinity for undirected graphs. He likes pairs of graphs that are similar in structure. However, Chef discovered that when the vertices of a graph are reorganized, it's often the case that the resulting graph, although still structurally similar to the original, can look completely different. Chef wants you to help him find similarities in pairs of graphs.

Chef only considers pairs of graphs where each graph has the same number of vertices (say  $N$ ). Chef then labels each vertex of each graph with an integer between 1 and  $N$  (inclusive), using each integer exactly once per graph. Chef then defines the similarity of the graphs as  $2 * \text{COMMON} / \text{TOTAL}$ , where  $\text{COMMON}$  is the number of edges appearing in both graphs (that is, the number of unordered pairs  $\{A, B\}$  such that in both graphs there exists an edge between the vertex labelled  $A$  and the vertex labelled  $B$ ), and  $\text{TOTAL}$  is the total number of edges in both graphs.

Chef's measure of similarity depends on how the vertices are labelled. Chef wants you to help him find a labelling that maximizes the similarity. Optimal solutions are not required, but better solutions will earn more points.

---

### Input

Input will begin with an integer  $T$ , the number of test cases. Each test case will begin with an integer  $N$ , the number of vertices in both graphs.  $2 * N$  lines follow. The first  $N$  lines

describe the first graph, and the next  $N$  lines the second graph. Each graph description consists of  $N$  lines of  $N$  integers each. The  $i$ -th integer on the  $j$ -th line will be 1 if there is an edge between vertices  $i$  and  $j$ , and 0 otherwise. The  $i$ -th integer on the  $j$ -th line will always be equal to the  $j$ -th integer on the  $i$ -th line, and the  $i$ -th integer on the  $i$ -th line will always be 0.

---

## Output

For each test case, output 2 lines with  $N$  integers each. Each line must contain a permutation of the integers 1 through  $N$ , and indicates how Chef should label the corresponding graph.

---

## Scoring

Your score for each test case is the similarity of the 2 graphs using the labelling you provide. Your overall score is the average of your scores on the individual test cases.

---

## Sample Input

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

0 0 1 1

0 0 0 0

1 0 0 0

1 0 0 0

## Sample Output

1 2 3

1 2 3

1 4 2 3

2 4 1 3

This output would score  $2*0/3 = 0.0$  on the first test case, and  $2*2/4 = 1.0$  on the second test case, for an overall score of 0.5. Note that better scores are possible.

## Test case generation

For each official test file, T is 5. For each test case, N is randomly chosen between 30 and 75. A real number D is randomly chosen between 0.05 and 0.5. For each pair of vertices, an edge is added with probability D. This graph is output as the first graph. An integer C is randomly chosen between 0 and  $N*(N-1)/2$ . C distinct pairs of vertices are chosen. For each pair, if an edge currently exists between them, the edge is removed with probability  $(1-D)$ . If no edge exists between them, one is added with probability D. Then, a random permutation is applied to the vertices of the graph, and it is output as the second graph. You may safely assume there will be no test cases where TOTAL is 0.

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Parallel Computing

## Parallel Computing Problem Code: PARALLEL

Chef needs a program to solve a simple task. The program will start with an array of integers X. After the execution it should hold values  $X[1] + X[2] + \dots + X[i]$  at all positions  $1 \leq i \leq N$ . However, he wants this done fast and efficiently. He needs your help to output a program which can be executed in parallel on N machines and will produce a correct result. Note that he needs a program for some fixed size of array  $1 \leq N \leq 1000$ .

A program consists of a series of synchronized steps. Instructions within one step are executed in parallel on multiple (at most N) machines which all have access to the shared memory where the array is located. You should not make any assumptions about the order in which these parallel instructions will be completed, but you can be sure that all instructions in the current step will be completed before any other instruction from the next step. There is only one type of instruction and it has the form "a+b=c". It reads the values  $X[a]$ ,  $X[b]$  and writes their sum in  $X[c]$ . The program should contain at most 20 steps and the total number of instructions should not exceed 2000.

---

## Input

Input contains a single positive integer N.

---

## Output

Output the number of steps in your program. In the following lines describe these steps. First output the number of instructions which should be executed in parallel on this step followed by a list of instructions. All items should be separated by a single space.

---

## Example

**Input:**

6

**Output:**

3

2 1+2=2 4+5=5

2 2+3=3 5+6=6

3 3+4=4 3+5=5 3+6=6

---

## Explanation

Lets name values in the array with letters from a to f. The table below describes contents of the array after each step:

---

1	2	3	4	5	6
---	---	---	---	---	---

initial values:	a	b	c	d	e	f
after step 1:	a	a+b	c	d	d+e	f
after step 2:	a	a+b	a+b+c	d	d+e	d+e+f
after step 3:	a	a+b	a+b+c	a+b+c+d	a+b+c+d+e	a+b+c+d+e+f

[Home](#) » [Compete](#) » [April Challenge 2012](#) » PDS Number

## PDS Number Problem Code: PDSNUM

A positive integer is called a "PDS Number" if the product of its digits is divisible by the sum of its digits. Let PDS(N) be the N-th PDS Number (indexed from 1), you are requested to calculate it.

### Input

There are several test cases (at most 10000), each formed as follows:

- The first and only line contains a positive integer N ( $N \leq 10^9$ ).  
The input is ended by  $N = 0$ .

### Output

For each test case, output on a line the respective PDS(N) calculated.

### Example

#### Input:

```
1
20
0
```

#### Output:

```
1
66
```

## Lucky Array Problem Code: LUCKY4

Chef loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47, 744, 4** are lucky and **5, 17, 467** are not.

Chef likes sequences of **n** positive integers, each of which does not exceed **m**. The total number of such sequences is equal to  **$m^n$** .

Chef has an array **C** of **n-1** integers. Let **F(x)** be equal to the number of lucky digits in decimal representation of integer **x**. Let there be a sequence **A** of **n** integers (1-based numeration). Chef calls it lucky when the following hold: if for each **i** (**1 <= i < n**) if **C[i]** equals to **1**, then **F(A[i])** must be equal to **F(A[i+1])** and if **C[i]** equals to **0**, then **F(A[i])** must not be equal to **F(A[i+1])**.

Chef has integers **n, m, k** and an array **C**. He wants to find out the **k-th** lucky sequence for a given array **C**. Help him. If the **k-th** lucky sequence does not exist, print the only integer - **1**.

The **k-th** lucky sequence is the **k-th** (1-based numeration) sequence in lexicographically sorted list of all lucky sequences for given integers **n, m** and an array **C**. The sequence **A** is lexicographic less than the sequence **B** if there exists integer **x** (**1 <= x <= n**) that **A[x] < B[x]** and **A[y] = B[y]** for all **y** (**1 <= y < x**).

---

### Input

First line contains one number **T** - the number of test cases. Each test is formed as following: first line contains **3** integers **n, m**, and **k**. Next line contains **n-1** integers - array **C** for corresponding test.

---

### Output

For each **T** test cases print sequence of integers - answer for corresponding test.

---

### Constraints

$2 \leq T \leq 10$

$2 \leq n \leq 50$

$1 \leq m, k \leq 10^9$

$0 \leq C[i] \leq 1$

---

## Example

Input:

2

2 4 7

0

3 7 4

0 1

Output:

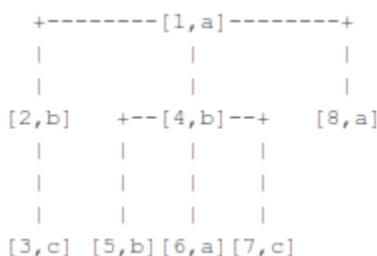
-1

1 7 7

[Home](#) » [Compete](#) » [April Challenge 2012](#) » Substrings on a Tree

## Substrings on a Tree Problem Code: TSUBSTR

Chefland scientists have made a new invention! They developed a new way to represent a string with **N** symbols: consider a tree with **N** vertices, rooted at the first vertice. For each vertice, a single latin letter is written. So we have obtained a "**treestring**". The scientists haven't decided yet how the treestring should be pronounced, but they have invented a definition of a **substring for a treestring**. A string is a **substring of a treestring** if and only it can be obtained by moving from some vertice to its descendant and writing out all the letters from vertices that occurred on this path in the order they have appeared. For example, consider the following treestring :



The string "ba" is a **substring** of a given treestring because it can be obtained by moving from vertice **4** to vertice **6**, the string "abb" is also a substring of this treestring - it can be obtained by moving from the root to vertice **5**. However the string "cb" is **not a substring** of

this treestring because there is no way from any vertex to its descendant in such a way that the sequence of letters is "cb".

Now the Chefland researchers ask you to help them with the treestring research. They have given you a treestring with **N** vertices. Please output the number of **distinct** substrings of a given treestring (including the empty one). Then, **Q** queries will follow. For the **i**-th query, the permutation **P<sub>i</sub>** of **26** latin alphabet letters and an integer **K<sub>i</sub>** will be given. That means that if we sort all distinct substrings of the given treestring according to the **alphabetical order described in P<sub>i</sub>**, you will have to output the **K<sub>i</sub>**-th string. "According to the alphabetical order described in **P<sub>i</sub>**" means that letter **X** is lexicographically smaller than letter **Y** if and only **X** appears in **P<sub>i</sub>** earlier than **Y**. For example if the **alphabetical order** is "cbadefghijklmnopqrstuvwxyz", then letter "c" is **lexicographically smaller** than letter "a" because "c" is the first symbol of this permutation, and "a" is the third symbol of this permutation, therefore  $1 < 3$  and for the given arrangement, "c" is **alphabetically less** than "a". Here note that the string **A** is smaller than the string **B** (that means **A** comes earlier than **B** after sorting) if and only if **A** is a prefix of **B**, or  $A_i = B_i$  (for all  $i < k$ ) and  $A_k < B_k$  (in terms of alphabetical order) where **A<sub>i</sub>** denotes the **i**-th letter of **A**.

## Constraints

- $1 \leq N \leq 250000$
- $1 \leq Q \leq 50000$
- $1 \leq K_i \leq 9223372036854775807$  ( $2^{63}-1$ )
- Output will not exceed 800 KB.
- It is guaranteed that the **N** lowercase latin letters have been generated randomly.

## Input

The first line of input consists of two integers - **N** and **Q**.

Then, a string composed of **N** lowercase latin letters follow.

Then, **N-1** lines follow. Each line is composed of **two** numbers - **X<sub>i</sub>** and **Y<sub>i</sub>**. It means that there is an edge between vertex **X<sub>i</sub>** and vertex **Y<sub>i</sub>**.

Then, **Q** lines follow. Each line consists of a permutation of **26** lowercase latin letters **P<sub>i</sub>** and an integer **K<sub>i</sub>**.

## Output

Output **Q+1** lines. On the first line output a single integer - the number of **distinct** substrings of a given treestring. The following **Q** lines should contain answers to the queries. **I**-th line should contain an answer to **i**-th query or a string "-1" if it is impossible to find **K<sub>i</sub>**-th string for **i**-th query.

## Example

**Input:**

8 4

abcbbaca

1 2

2 3

1 4

4 5

4 6

4 7

1 8

abcdefghijklmнопqrstuvwxyz 5

abcdefghijklmнопqrstuvwxyz 1

bcadefghijklmнопqrstuvwxyz 5

abcdefghijklmнопqrstuvwxyz 100

**Output:**

12

aba

ba

-1

# Find a special connected block

Problem Code: CONNECT

Given an  $n*m$  board with a number between -1 and  $n*m$  in every entries.

And an  $n*m$  matrix  $M$  is also given, where  $M_{i,j}$  is the cost of selecting the  $(i,j)$  entry of the given board.

Your task is to find a connected block (which means these entries can reach each other by just go up, down, left and right without going out the block) in the board that contains at least  $K$  distinct positive numbers without any -1, and it must have minimum total cost for selecting these entries. Output the minimum total cost.

## Input

First line consists of three integers,  $n, m, K$  ( $1 \leq n, m \leq 15, 1 \leq K \leq 7$ ).

The followings are two  $n*m$  matrices, the first denotes the numbers on the board and the second denotes the cost of every entry.

Namely, the first  $n$  lines contain  $m$  integers, where the  $j$ th number in  $i$ th line denotes the number on the entry  $(i,j)$  of the board. These integers are in  $[-1, n*m]$ .

Next  $n$  lines contain  $m$  integers too. The  $j$ th number in  $i$ th line denotes the cost of selecting the entry  $(i,j)$  of the board. These integers are in  $[1, 100000]$ .

## Output

Only one line contains the minimum cost to finish the task. If the task is impossible, output -1 please.

## Example

### Input:

3 3 3

0 0 1

2 3 3

-1 2 1

3 1 5

4 10 1

9 3 4

**Output:**

8

COOK21	<a href="#">April Cook-Off 2012</a>	22 Apr 2012 21:30:00	2 hours 30 minutes	706
--------	-------------------------------------	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [April Cook-Off 2012](#) » Top Batsmen

## Top Batsmen Problem Code: BESTBATS

A cricket team consists of 11 players and some are good at batting, others are good at bowling and some of them are good at both batting and bowling. The batting coach wants to select exactly **K** players having maximum possible sum of scores. Given the batting score of each of the 11 players, find the number of ways in which we can select exactly **K** players such that the sum of their scores is the maximum possible. Two ways are different if there is a player who is selected in one of them is not in the other. See explanation of sample cases for more clarity.

---

### Input

First line contains **T**, number of test cases ( $1 \leq T \leq 100$ ). **T** cases follow, each having 2 lines. First line of each case contains scores of 11 players ( $1 \leq \text{score} \leq 100$ ) and the second line contains **K** ( $1 \leq K \leq 11$ )

---

### Output

For each test case, output the answer in a new line.

---

### Example

**Input:**

2

1 2 3 4 5 6 7 8 9 10 11

3

2 5 1 2 4 1 6 5 2 2 1

6

**Output:**

1

6

**Explanation:**

Case 1 : Maximum possible sum of scores =  $11 + 10 + 9 = 30$  and can be achieved only by selecting the last 3 players. Only one possible way.

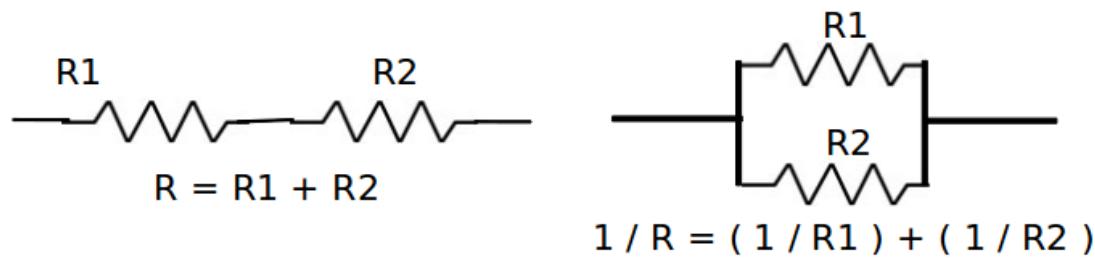
Case 2 : Maximum possible sum of scores =  $6 + 5 + 5 + 4 + 2 + 2 = 24$  and considering the players as p1 p2 p3 ... p11 in that order, the ones with maximum possible sum of scores is as follows

{p1, p2, p4, p5, p7, p8}  
 {p10, p2, p4, p5, p7, p8}  
 {p1, p2, p10, p5, p7, p8}  
 {p9, p2, p4, p5, p7, p8}  
 {p1, p2, p9, p5, p7, p8}  
 {p10, p2, p9, p5, p7, p8}

[Home](#) » [Compete](#) » [April Cook-Off 2012](#) » Resistance

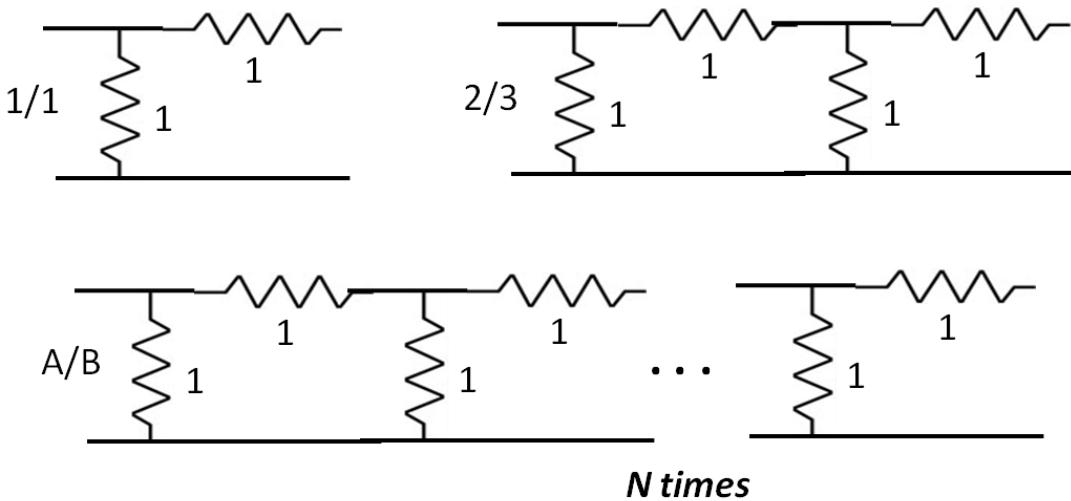
## Resistance Problem Code: RESIST

The electrical resistance is the opposition to the passage of electric current. If two resistors with resistance  $R_1$  and  $R_2$  are connected to each other, the resultant resistance  $R$  depends on how their ends are connected. If they are connected in Series, they simply add up to give  $R = R_1 + R_2$ . If they are connected in Parallel, its given by the equation  $1/R = 1/R_1 + 1/R_2$ . This is shown below.



We have a long circuit of resistors of one unit resistance each ( $R_i = 1$ ) having  $N$  blocks. Each block has 2 resistors as shown in the figure below. For a circuit with only one block ( $N=1$ ), the equivalent resistance at the ends on left-side is  $R = 1$  (note that the right-most horizontally aligned resistor has no influence, as its not part of any simple path between the

two left ends ). For a circuit with two blocks (N=2), its  $2/3$  units. This is shown in the figure below.



Given  $N$ , find the resistance  $A/B$  at the left ends of the circuit having  $N$  blocks. The values can be huge, so reduce the fraction  $A/B$  to lowest terms  $P/Q$ , where  $\text{greatest_common_divisor\_of } (P, Q) = 1$  and print  $(P\%M)/(Q\%M)$  for a given modulo  $M$ .

## Input

First line contains  $T$  ( number of test cases,  $1 \leq T \leq 10$  ). Each of the next  $T$  lines contains  $N$   $M$  ( $1 \leq N \leq 10^6$  and  $2 \leq M \leq 2^{30}$  )

## Output

For each test case, output the equivalent resistance at the left ends, in the form  $(P\%M)/(Q\%M)$  as explained above, in a separate line.

## Example

**Input:**

3

1 10

2 12

100 10000

**Output:**

1/1

2/3

4301/9525

[Home](#) » [Compete](#) » [April Cook-Off 2012](#) » Crease Painting

## Crease Painting Problem Code: PAINTING

---

In cricket, the long white line before the batsman and the runner is called crease. We have a painting machine that can be programmed to paint white lines on the field. Imagine the cricket ground as an infinite grid of cells and the painting machine is initially located at the cell (0,0) which is already colored white and all other cells are in green color. Each instruction to the machine is of the form "direction distance", where direction is one of 'U' (up), 'D' (down), 'R' (right) or 'L' (left) and distance is a positive integer. On reading an instruction, the machine moves in the specified direction for specified distance and paints all the unpainted (green) cells with white color. Given a sequence of N instructions, find the number of cells that are painted white, for each of them.

---

### Input

First line contains N, number of instructions ( $1 \leq N \leq 1000$ ). Each of the next N lines contains an instruction of the form DIR DIST, where DIR is one of ( U/D/R/L ) and DIS is a positive integer ( $1 \leq DIS \leq 10^9$ )

---

### Output

For each instruction in the given order, output the number of cells that are painted white for each of them, in a separate line.

---

### Example

**Input:**

5

R 100

U 10

L 120

D 10

R 200

**Output:**

100

10

120

10

99

**Explanation:**

R 100 : all 100 cells from (1,0) to (100,0) are painted white

U 10 : all 10 cells from (100,1) to (100,10) are painted white

L 120 : all 120 cells from (99,10) to (-20,10) are painted white

D 10 : all 10 cells from (-20,9) to (-20,0) are painted white

R 200 : The cells [ (0,0) to (100,0) ] are already painted white, so only the 19 cells [ (-19,0) to (-1,0) ] + the 80 cells [ (101,0) to (180,0) ] are painted white in this step.

[Home](#) » [Compete](#) » [April Cook-Off 2012](#) » Selection for Training Camps

## Selection for Training Camps Problem Code: TRAINING

---

Cricket is almost like a religion in India and many young kids attend coaching camps every year. This year there are total N kids at the camp and the coach is forming teams so that kids within a team can practice and compete with each other. As we know that kids are afraid of tall and heavy persons, they are afraid to get in to a team having such player. More formally, kid A is afraid of another kid B if weight(A)  $\leq$  weight(B) and height(A)  $\leq$  height(B). You can assume that no two kids have both height and weight equal to each other. Here is how the teams are formed by the coach.

All the  $N$  students are lined up. For  $R = 1, 2, 3, \dots$  in order, in the  $R$ th round the coach calls all the students who are not afraid of any one in the line. They all come forward, form a team number  $R$  and leave for practice. This process is repeated as long as there are kids still in the line, yet to form a team. Given the height and weight of each of the  $N$  students, find the number of teams formed and the number of kids in each of them.

---

## Input

First line contains  $T$ , number of test cases ( $1 \leq T \leq 3$ ).  $T$  cases follow. Each case starts with an integer  $N$  (number of students,  $1 \leq N \leq 100,000$ ), each of the following  $N$  lines contains integers  $H_i W_i$  (height and weight of the  $i$ th kid,  $1 \leq H_i, W_i \leq 10^9$ . No two kids have their corresponding  $H_i$  and  $W_i$  equal to each other).

---

## Output

For each case, output  $K$ , the total number of teams formed, in the first line. In the second line, output  $K$  integers  $S_1 S_2 \dots S_K$ , where  $S_i$  is the number of kids in the  $i$ th team, separated by space.

---

## Example

Input:

2

3

3 4

5 5

4 3

6

7 7

4 7

5 5

1 2

3 4

2 1

**Output:**

2

1 2

4

1 2 1 2

**Explanation:**

Case 1 : There are 3 kids initially in the line, (3,4) (5,5) (4,3)

In the first round, only (5,5) comes forward and he leaves. (3,4) and (4,3) are both afraid of (5,5).

In the second round, both (3,4) and (4,3) come forward, as they are not afraid of each other.

Case 2 :

Round 1 : Only (7,7) comes forward, as all others are afraid of him

Round 2 : (4,7) and (5,5) form a team

Round 3 : (3,4) comes forward, as (1,2) and (2,1) are afraid of him

Round 4 : (1,2) and (2,1) are not afraid of each other and so forms the last team.

[Home](#) » [Compete](#) » [April Cook-Off 2012](#) » Best Buggy Ratings

## Best Buggy Ratings Problem Code: TMRATING

A rating system for cricket players is built by Mr.Bugs Bunny and as expected, it has lot of bugs. The initial ratings of  $N$  players ( numbered 0 to  $N-1$  ) are given and we refer it as array  $V_0$ . The system is expected to answer simple queries of the form, find the top-2 maximum ratings among players from  $i$  to  $j$  ( inclusive ). Top-2 maximum means the first two elements of the ratings sorted in non-increasing order. Due to bugs in the system, an unexpected update occurs after output of each query and this creates a new version of the ratings array  $V$ . We refer the  $K$ th version of the ratings arrays as  $V_K$ . Also, for a query, the system queries on some previous version of  $V$ . Mr. Bunny gave the exact details of the bugs as follows,

Given  $Q$  queries numbered 1 to  $Q$  in order and values of  $A$   $B$   $C$   $D$   $M$ ,

- For a query number  $K$ , the query is made on the array  $V_t$ , where  $t = ( A * R1 + D ) \% K$ , and  $R1$  is the maximum rating in the query range of the previous query. For the first query, consider  $R1 = 0$ .
- For a query number  $K$ , let  $R1$  and  $R2$  be the top-2 maximum ratings ( $R1 \geq R2$ ). After it outputs this answer, the system changes the rating of a player.

Specifically, the rating of player number  $( B * R1 + D ) \% N$  is changed to  $( C * R2 + D ) \% M$ . This update is on the array  $V_{(K-1)}$  and a new version  $V_K$  is created.

You can not fix these bugs, but can you guess the output produced by this system. For more clarity, check the pseudo code below.

```

read array V0;
R1 = 0, R2 = 0;
for K = 1 to Q
    t = ( A * R1 + D ) % K
    read qi qj
    R1, R2 = top-2 Maximum ratings in range [qi..qj] in the array Vt
    Output R1 R2
    VK = Update array V(K-1) by changing V(K-1) [ ( B * R1 + D ) \% N ] = ( C * R2 + D ) \% M
end-for

```

Note: Take care of potential overflows in intermediate calculations in the equations mentioned above. The authors algorithm doesn't depend on the values of A B C D M, they are just used to generate some values.

## Input

First line contains 6 integers N M A B C D and the second line contains N integers, the initial ratings of N players in order ( $2 \leq N, A, B, C, D \leq 100,000 ; 0 \leq V_0[i] , M < 1,000,000,000 ; M \geq 2$ ). Next line contains Q ( number of queries  $1 \leq Q \leq 100,000$  ), followed by Q lines. The Kth line in this has the query number K, and has two integers qi qj ( $0 \leq q_i < q_j < N$ ).

## Output

For each query, output the top-2 maximum ratings R1 R2 ( $R1 \geq R2$ ) in a new line.

## Example

**Input:**

6 1000 2 2 2 2

1 2 3 4 5 6

4

0 5

0 3

1 3

2 5

**Output:**

6 5

4 3

12 4

12 8

**Explanation:**

$$V_0 = \{ 1, 2, 3, 4, 5, 6 \}$$

1.)  $t = 0 \rightarrow \text{top-2 max of } V_0[0..5] = 6, 5$

Updating  $V_0[2]$  with 12

$$V_1 = \{ 1, 2, 12, 4, 5, 6 \}$$

2.)  $t = ( A(=2) * R1(=6) + D(=2) ) \% 2 = 0 \rightarrow \text{top-2 max of } V_0[0..3] = 4, 3$

$$R1 = 4, R2 = 3$$

Updating index =  $( B(=2) * R1(=4) + D(=2) ) \% N(=6) = 4$  with value =  $( C(=2) * R2(=3) + D(=2) ) \% M(=1000) = 8$

$$V_2 = \{ 1, 2, 12, 4, 8, 6 \}$$

3.)  $t = 1 \rightarrow \text{top-2 max of } V_1[1..3] = 12, 4$

Updating  $V_2[2] = 10$

$$V_3 = \{ 1, 2, 10, 4, 8, 6 \}$$

4.)  $t = 2 \rightarrow \text{top-2 max of } V_2[2..5] = 12, 8$

Updating  $V_3[2] = 18$   
 $V_4 = \{ 1, 2, 18, 4, 8, 6 \}$

MAY12

[May Challenge 2012](#)01 May 2012  
15:00:00

10 days

2100

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Jewels and Stones

## Jewels and Stones Problem Code: STONES

Soma is a fashionable girl. She absolutely loves shiny stones that she can put on as jewellery accessories. She has been collecting stones since her childhood - now she has become really good with identifying which ones are fake and which ones are not. Her King requested for her help in mining precious stones, so she has told him which all stones are jewels and which are not. Given her description, your task is to count the number of jewel stones.

More formally, you're given a string  $J$  composed of latin characters where each character is a jewel. You're also given a string  $S$  composed of latin characters where each character is a mined stone. You have to find out how many characters of  $S$  are in  $J$  as well.

---

### Input

First line contains an integer  $T$  denoting the number of test cases. Then follow  $T$  test cases. Each test case consists of two lines, each of which contains a string composed of English lower case and upper characters. First of these is the jewel string  $J$  and the second one is stone string  $S$ .

You can assume that  $1 \leq T \leq 100$ ,  $1 \leq |J|, |S| \leq 100$

---

### Output

Output for each test case, a single integer, the number of jewels mined.

---

### Example

**Input:**

4

abc

abcdef

aA

abAZ

aaa

a

what

none

**Output:**

3

2

1

0

[Home](#) » [Compete](#) » [May Challenge 2012](#) » [Lucky lucky number](#)

## Lucky lucky number Problem Code: CHEFLUCK

*Every great chef knows that lucky numbers are positive integers whose decimal representations contain only the lucky digits 4 and 7. For example, numbers 47, 744, 4 are lucky and 5, 17, 467 are not.*

Our chef has recently returned from the Lucky country. He observed that every restaurant in the Lucky country had a lucky number as its name. He believes that having a lucky number as a restaurant name can indeed turn out to be very lucky.

Our chef believes that it is possible to make a lucky number having N digits even luckier. Any number following the rules below is called Lucky lucky number -

1. The number contains only digits 4 and 7.
2. Count of digit 4 in the number should be divisible by 7.
3. Count of digit 7 in the number should be divisible by 4.

Help our chef to compute the count of digit 4 in the **smallest** Lucky lucky number having N digits.

---

### Input

First line contains T, number of test cases. Each of the next T lines contains a number N, the number of digits in the Lucky lucky number to be formed.

$1 \leq T \leq 1000$   
 $1 \leq N \leq 1000000000$  ( $10^9$ )

---

## Output

If it is not possible to form a Lucky lucky number having N digits, output -1. Otherwise, output the count of digit 4 in the smallest Lucky lucky number having N digits.

---

## Example

**Input:**

5

7

4

11

1

15

**Output:**

7

0

7

-1

7

## Explanation

For the last test case,  $N = 15$ , the smallest lucky lucky number is

4444444777777777. The count of digit 4 is 7.

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Remember the recipe

## Remember the recipe Problem Code: TWSTR

---

Chef Jessie has a lot of recipes with her (**N**). She often remembered the starting few characters of the recipe and forgot the rest. As all the great chefs do, Jessie also numbered the recipes depending on the priority. So, given the list of recipes along with their priorities answer Jessie's queries.

Jessie's queries are as follows:

She gives you the first few characters of a recipe; you have to print the complete recipe with the highest priority.

### Note:

Every recipe has a unique priority

---

## Input

First line contains an integer **N** - the number of recipes.

Followed by **N** strings **Si** along with an integer each **Vi**.

**Si** stands for the recipe and **Vi** for the priority.

It is followed by an integer **Q** - the number of queries.

Followed by **Q** strings **Qi**.

Each string **Si**, **Qi** contain only lowercase Latin alphabets ('a' - 'z') and '-'.

---

## Output

**Q** – lines, each contain the answer for each of the query.

If for a query no recipe matches print "**NO**". (Without quotes)

### Constraints:

$0 \leq N \leq 1000$

$0 \leq Q \leq 1000$

$-10^9 \leq Vi \leq 10^9$

$1 \leq |Si| \leq 1000$  (length of **Si**)

$1 \leq |Qi| \leq 1000$  (length of **Qi**)

---

## Example

Input:

flour-with-eggs 100

chicken-ham -10

flour-without-eggs 200

fish-with-pepper 1100

6

f

flour-with

flour-with-

c

fl

chik

**Output:**

fish-with-pepper

flour-without-eggs

flour-with-eggs

chicken-ham

flour-without-eggs

NO

[Home](#) » [Compete](#) » [May Challenge 2012](#) » A Home for Chef

## A Home for Chef Problem Code: CHEFHOME

---

Our hardworking chef is bored of sleeping in his restaurants. He has decided to settle down. The first thing he must do is to find a suitable location to build a palatial home.

Think of the city as a two-dimensional grid. There are **N** restaurants in the city. Each of the chef's restaurant is a point denoted by (**X** , **Y**). A **house** can be located at a grid point (**R**, **S**) if the sum of the distances between this point and each of the restaurants is as small as possible. Find the number of possible house locations in the city to help out chef build a home.

More than one restaurant can be located at the same point.

Houses and restaurants can be located at the same point.

Every house must have integer co-ordinates. In other words, **R** and **S** are integers.

The distance between two points (A,B) and (C,D) is  $|A-C| + |B-D|$ . Here  $|X|$  is the absolute function.

---

## Input

First line in the input contains **T**, number of test cases.

First line of each test case contains **N**, number of restaurants.

Each of the next **N** lines contain two integers **X** and **Y** separated by a space.

**T**  $\leq 100$

**N**  $\leq 10^3$

$-10^8 \leq X \leq 10^8$

$-10^8 \leq Y \leq 10^8$

---

## Output

The number of possible locations (grid points) where houses can be built.

---

## Example

**Input:**

3

5

0 0

-1 0

1 0

0 1

0 -1

5  
31 11

30 -41  
20 14  
25 18

25 38  
2  
0 0  
1 1

**Output:**

1  
1  
4

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Divisible Pairs

## Divisible Pairs Problem Code: DIVPAIR

Given **N** and **M** Dexter wants to know how many pairs  $a, b (1 \leq a < b \leq N)$  are there such that  $(a+b)$  is divisible by **M**. For example when **N=4** and **M=3**, there are 2 possible pairs the sum of which is divisible by **M** and they are (1,2) and (2,4).

### Input

First line of input contains **T**( $\leq 100000$ ) which is the number of test cases. Each of the next **T** lines contains two integers **N**( $1 \leq N \leq 10^9$ ) and **M**( $2 \leq M \leq 10^9$ ).

### Output

Output one line per testcase, the number of pairs  $(a, b)$  as described before.

---

## Example

Input:

3

2 3

4 3

1 6

Output:

1

2

0

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Little Elephant and Median

## Little Elephant and Median Problem Code: MEDIAN

Little Elephant from Zoo of Lviv likes medians so much. Let us define **median** term for some array **A**. Let **B** be the same array **A**, but sorted in non-decreasing order. Median of **A** is **B<sub>m</sub>** (1-based indexing), where **m** equals to **(n div 2)+1**. Here 'div' is an integer division operation. So, for a sorted array with 5 elements, median is the 3rd element and for a sorted array with 6 elements, it is the 4th element.

Little Elephant has an array **A** with **n** integers. In one turn he can apply the following operation to any consecutive subarray **A[l..r]**: assign to all **A<sub>i</sub>** (**l <= i <= r**) median of subarray **A[l..r]**.

Let **max** be the maximum integer of **A**. Little Elephant wants to know the minimum number of operations needed to change **A** to an array of **n** integers each with value **max**.

For example, let **A = [1, 2, 3]**. Little Elephant wants to change it to **[3, 3, 3]**. He can do this in two operations, first for subarray **A[2..3]** (after that **A** equals to **[1, 3, 3]**), then operation to **A[1..3]**.

---

Input

First line of the input contains single integer **T** - the number of test cases. Then **T** test cases follow, each of such format: first line - integer **n**, second line - array **A** consisted of **n** integers.

---

## Output

In **T** lines print the results for each test case, one per line.

---

## Constraints

$1 \leq T \leq 100$

$1 \leq n \leq 30$

$1 \leq A[i] \leq 10^9$

---

## Example

**Input:**

2  
3  
1 2 3

4  
2 1 1 2

**Output:**

2  
1

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Killing Gs

## Killing Gs Problem Code: CKROACH

---

There are the big enemies for all chefs, that is, Gs (the plural form of G). Chef Ciel find N Gs in her restaurant. Ciel is very scared, and she is going to kill the Gs by using

insecticides. Here  $M$  insecticides are available, and the price of the  $j$ -th insecticide is  $C_j$ , and she can use the  $j$ -th insecticide for all  $N$  Gs if she buys it. Unfortunately Gs may have resistance to insecticides, so Gs may be still alive after using some insecticides. But we know that if Ciel use the  $j$ -th insecticide, the  $i$ -th G will be dead with the probability  $P_{i,j}\%$ , and these are statistically independent.

Chef Ciel would like to buy some insecticides, and then, Ciel may use the insecticides one by one. Ciel hopes that the  $i$ -th G will be dead with a probability at least 90%, for all  $i$ , after using all insecticides which she buys.

Your work is selecting a cheaper (cheapest is not necessary) set of distinct insecticides which she buys. Don't forget that the set must kill the  $i$ -th G with a probability at least 90% for all  $i$ .

---

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line for each test case has 2 integers  $N$  and  $M$ . The next line has  $M$  integers  $C_1, C_2, \dots, C_M$ . Then next  $N$  lines have  $M$  integers  $P_{i,j}$ , that is the  $j$ -th integer of the  $i$ -th line.

---

## Output

For each test case, you should print two lines. The first line should have an integer  $K$ , the number of insecticides which Ciel buys. The second line should have  $K$  integers, the numbers of insecticides which Ciel buys, in ascending order.

---

## Constraints

$T = 50$  (except for Sample Input)

$50 \leq N \leq 200$  (except for Sample Input)

$50 \leq M \leq 200$  (except for Sample Input)

$1 \leq C_j \leq 1000000 (10^6)$

$0 \leq P_{i,j} \leq 90$

The probability of killing the  $i$ -th G is at least 90% for all  $i$ , if Ciel buys all insecticides.

---

## Scoring

For each test case, the score is defined as  $(\text{basic} - \text{your}) / \text{basic}$ , where  $\text{your}$  is the cost of your set and  $\text{basic}$  is the cost of the set given by [CKROACH simplesolution.c](#). If the score is negative, it is treated as 0. Your objective is to maximize the total score.

---

## Test Case Generation

There is only one input file for this problem. Each test case generated by the below method.

$N$ ,  $M$  and  $P_{i,j}$  are chosen uniformly at random such that they satisfy constraints. After that, each  $P_{i,j}$  becomes 0 with the probability 0.75. Then an integer  $x$  is chosen from the set  $\{1, 2, 5, 10, 100\}$  uniformly at random. For each  $j$  ( $1 \leq j \leq M$ ), a real number  $y_j$  is chosen from  $[0, 1]$  uniformly at random, and  $C_j$  is set at  $(x + y_j) * (P_{1,j} + P_{2,j} + \dots + P_{N,j})$  rounded to the nearest integer. If  $C_j$  is less than 1, then  $C_j$  becomes 1. If  $C_j$  is more than 1000000, then  $C_j$  becomes 1000000.

Finally, if the generated case does not satisfy the constraint "The probability of killing the  $i$ -th  $G$  is at least 90% for all  $i$ , if Ciel buys all insecticides", this case is destroyed and do the above process once again.

A test case generator is available [here](#).

## Sample Input

```
2
2 3
100 100 190
90 0 90
0 90 90
1 5
10 20 40 60 85
10 20 40 60 85
```

## Sample Output

```
1
3
2
3 5
```

## Output details

In the first case, the third insecticide will kill the  $i$ -th G with probability 90% for all  $i$ .

In the second case, after using the third and fifth insecticides, G is still alive with the probability  $(1 - 40/100) * (1 - 85/100) = 0.6 * 0.15 = 0.09$ . Therefore the set of the third and fifth insecticides will kill the G with the probability 91%.

The output for Sample Input of [CKROACH simplesolution.c](#) is the following. (which is far from the optimal)

2

1 2

5

1 2 3 4 5

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Digits Forest

## Digits Forest Problem Code: DIGFORST

---

Given a simple undirected graph containing  $N$  vertices numbered 1 to  $N$ , each vertex containing a digit from  $\{1,2,\dots,7\}$ . Starting at the vertex 1 with an empty string  $S$ , we travel through some vertices (with no limitations) to the vertex  $N$ . For every vertex on the way, we add the respective digit to the right of the string  $S$ . At last we get  $S$  as a decimal integer. You are requested to find such a way satisfying  $S$  is divisible by all of its digits, and the sum of digits of  $S$  must be as small as possible.

---

### Input

There are several test cases (fifteen at most), each formed as follows:

- The first line contains a positive integer  $N$  ( $N \leq 100$ ).
- The second line contains  $N$  digits (separated by spaces), the  $i$ -th digit is the value of the  $i$ -th vertex.
- $N$  last lines, each contains  $N$  values of  $\{0, 1\}$  (separated by spaces), the  $j$ -th value of the  $i$ -th line is equal to 1 if there is an edge connecting two vertices  $(i, j)$ , otherwise 0.

The input is ended with  $N = 0$ .

---

### Output

For each test case, output on a line the minimum sum of digits found, or **-1** if there's no solution.

---

## Example

Input:

```
4
1 2 1 4
0 1 1 1
1 0 0 1
1 0 0 1
1 1 1 0
0
```

Output:

```
7
```

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Little Elephant and Boxes

## Little Elephant and Boxes Problem Code: LEBOXES

Little Elephant from Zoo of Lviv has  $n$  boxes. He don't know what is in the boxes, but he thinks that  $i$ -th box (0-based numeration) contains  $V_i$  dollars. The probability that  $i$ -th box will contain money is  $P_i$  percents. Instead of money  $i$ -th box may contain a single diamond (with the probability  $100 - P_i$  percents).

There are  $m$  things to buy, numbered from  $0$  to  $m-1$ ,  $j$ -th thing costs exactly  $C_j$  dollars plus  $D_j$  diamonds. Little Elephant is very smart and if he has some number of dollars and diamonds he will always buy the maximal possible number of things he can afford. Note that there is just one copy of all  $m$  things.

Help Little Elephant to find the expected number of things he will buy.

---

## Input

First line of the input contains single integer  $T$  - the number of test cases.  $T$  test cases follow. First line of each test case contains pair of integers  $n$  and  $m$ . Next  $n$  lines of each

test case contain pairs of integers  $V_i$  and  $P_i$ , one pair per line. Next  $m$  lines of each test case contain pairs of integers  $C_j$  and  $D_j$ , one pair per line.

---

## Output

In  $T$  lines print  $T$  real numbers - the answers for the corresponding test cases. Round each number to 4 digits after decimal point.

---

## Constraints

$1 \leq T \leq 5$

$2 \leq n \leq 30$

$1 \leq m \leq 30$

$1 \leq V_i, C_j \leq 10^7$

$0 \leq D_j \leq 30$

$0 \leq P_i \leq 100$

---

## Example

Input:

2

2 2

2 50

2 100

2 0

2 0

2 2

2 100

2 50

0 2

0 1

**Output:**

1.5000

0.5000

[Home](#) » [Compete](#) » [May Challenge 2012](#) » Selling Tickets

## Selling Tickets Problem Code: TICKETS

Chef has prepared a large number of unique dishes for a fancy dinner. Patrons from all over the world are interested in purchasing tickets for the dinner. Each interested patron has specified 2 dishes that they are interested in eating for dinner. As long as a patron is served at least one of their 2 preferred dishes, they will be happy. Chef wants to sell as many tickets as possible while being able to guarantee that all patrons are happy. Unfortunately, Chef has no control over who buys the tickets, only how many will be sold. How many tickets can Chef sell?

### Input

Input will begin with an integer  $T$ , the number of test cases. Each test case begins with 2 integers  $N$  and  $M$ , the number of dishes and number of patrons, respectively.  $M$  lines follow, each containing 2 distinct integers between 1 and  $N$ , inclusive. The numbers on the  $i$ -th line represent the preferred dishes of the  $i$ -th patron. Each test case is followed by a blank line.

### Output

For each test case, output a single integer indicating the maximum number of tickets Chef can sell while still being able to guarantee that every patron will be happy no matter which patrons buy tickets.

### Sample Input

3

6 4

1 2

1 2

3 4

5 6

6 5

1 2

1 2

1 2

3 4

5 6

4 5

1 2

1 3

1 4

2 3

3 4

---

### Sample Output

4

2

4

In the second example, Chef cannot sell 3 tickets because it's possible that 3 patrons will arrive all preferring dishes 1 or 2.

---

## Constraints

- $T \leq 15$
- $2 \leq N \leq 200$
- $0 \leq M \leq 500$

COOK22	<a href="#">May Cook-Off 2012</a>	20 May 2012 21:30:00	2 hours 40 minutes	701
--------	-----------------------------------	-------------------------	--------------------	-----

[Home](#) » [Compete](#) » [May Cook-Off 2012](#) » Little Elephant and Strings

## Little Elephant and Strings Problem Code: LUCKYSTR

A Little Elephant from the Zoo of Lviv likes *lucky strings*, i.e., the strings that consist only of the lucky digits **4** and **7**.

The Little Elephant has **K** favorite lucky strings **A<sub>1</sub>**, **A<sub>2</sub>**, ..., **A<sub>K</sub>**. He thinks that the lucky string **S** is good if either  $|S| \geq 47$  or for some **j** from 1 to **K** we have that **A<sub>j</sub>** is a substring of **S**.

The Little Elephant has found **N** lucky strings **B<sub>1</sub>**, **B<sub>2</sub>**, ..., **B<sub>N</sub>** under the pillow. Now he wants to know which of them are good. Help him and find for each **i** from 1 to **N** whether the string **B<sub>i</sub>** is good or not.

### Notes.

Let **S** be some lucky string. Then

- $|S|$  denotes the length of the string **S**;
- $S[i]$  ( $1 \leq i \leq |S|$ ) denotes the  $i^{\text{th}}$  character of **S** (the numeration of characters starts from 1);
- The string **T** of the length **M** is called a *substring* of **S** if for some **k** from 0 to  $|S| - M$  we have  

$$T[1] = S[k + 1], T[2] = S[k + 2], \dots, T[M] = S[k + M].$$

---

## Input

The first line of the input file contains two integers **K** and **N**, the number of favorite lucky strings of the Little Elephant and the number of strings he has found under the pillow. Each of the following **K** lines contains one favorite lucky string. Namely,  $j^{\text{th}}$  line among these **K** lines contains the string **A<sub>j</sub>**. Each of the following **N** lines contains one lucky string that was found under the pillow. Namely,  $i^{\text{th}}$  line among these **N** lines contains the string **B<sub>i</sub>**. The input file does not contain any whitespaces.

---

## Output

For each of the **N** strings that were found under the pillow print **Good** if it is good, and **Bad** otherwise.

---

## Constraints

$1 \leq K, N \leq 50$

For each string **S** in the input file we have  $1 \leq |S| \leq 50$ .

Each string in the input file consists only of the lucky digits **4** and **7**.

---

## Example

### Input:

2 4

47

744

7444

447

7774

774

### Output:

Good

Good

Bad

Good

## Explanation

The string **S = 7444** is good since the favorite string **744** is its substring.

The string **S = 447** is good since the favorite string **47** is its substring.

The string **S = 7774** is bad since none of the favorite strings **47** and **744** is a substring of **S**.

[Home](#) » [Compete](#) » [May Cook-Off 2012](#) » Little Elephant and Balance

# Little Elephant and Balance

A Little Elephant from the Zoo of Lviv likes *lucky strings*, i.e., the strings that consist only of the lucky digits 4 and 7.

The Little Elephant calls some string **T** of the length **M** *balanced* if there exists at least one integer **X** ( $1 \leq X \leq M$ ) such that the number of digits **4** in the substring **T[1, X - 1]** is equal to the number of digits **7** in the substring **T[X, M]**. For example, the string **S = 7477447** is balanced since **S[1, 4] = 7477** has 1 digit **4** and **S[5, 7] = 447** has 1 digit **7**. On the other hand, one can verify that the string **S = 7** is not balanced.

The Little Elephant has the string **S** of the length **N**. He wants to know the number of such pairs of integers **(L; R)** that  $1 \leq L \leq R \leq N$  and the substring **S[L, R]** is balanced. Help him to find this number.

## Notes.

Let  $S$  be some lucky string. Then

- $|S|$  denotes the length of the string  $S$ ;
  - $S[i]$  ( $1 \leq i \leq |S|$ ) denotes the  $i^{\text{th}}$  character of  $S$  (the numeration of characters starts from 1);
  - $S[L, R]$  ( $1 \leq L \leq R \leq |S|$ ) denotes the string with the following sequence of characters:  $S[L], S[L + 1], \dots, S[R]$ , and is called a *substring* of  $S$ . For  $L > R$  we mean by  $S[L, R]$  an empty string.

## Input

The first line of the input file contains a single integer **T**, the number of test cases. Each of the following **T** lines contains one string, the string **S** for the corresponding test case. The input file does not contain any whitespaces.

## Output

For each test case output a single line containing the answer for this test case.

---

## Constraints

$1 \leq T \leq 10$

$1 \leq |S| \leq 100000$

**S** consists only of the lucky digits **4** and **7**.

---

## Example

**Input:**

4

47

74

477

4747477

**Output:**

2

2

3

23

---

## Explanation

In the first test case balance substrings are **S[1, 1] = 4** and **S[1, 2] = 47**.

In the second test case balance substrings are **S[2, 2] = 4** and **S[1, 2] = 74**.

Unfortunately, we can't provide you with the explanations of the third and the fourth test cases. You should figure it out by yourself. Please, don't ask about this in comments.

[Home](#) » [Compete](#) » [May Cook-Off 2012](#) » Little Elephant and Swapping

## Little Elephant and Swapping Problem Code: LUCKYSWP

A Little Elephant from the Zoo of Lviv likes *lucky strings*, i.e., the strings that consist only of the lucky digits **4** and **7**.

He is now studying some special transformation defined on the set of the lucky strings. The lucky string **T** is called a *swap permutation* of the lucky string **S** with  $|S| = N$  if it can be derived from **S** by the following process.

- Choose some integers **L** and **R** such that  $1 \leq L \leq R \leq N$ .
- Denote by **A** the substring  $S[L, R]$  and by **B** the concatenation  $S[1, L - 1] + S[R + 1, N]$ .
- Choose some integer **K** such that  $0 \leq K \leq |B|$ .
- Put  $T = B[1, K] + A + B[K + 1, |B|]$ .

In other words, **T** is a swap permutation of **S** if it can be obtained from **S** by deleting some non-empty substring from **S** and then inserting it back into any position of **S**. Note that **S** is always a swap permutation of itself.

Denote by **F(S)** the length of the longest non-decreasing subsequence of **S**. In other words, this subsequence should have one of the following forms: **444...444, 777...777, 444...444777...777**.

The Little Elephant has the lucky string **S** and as an experienced theoretical scientist he is interested in some quite theoretical problem. Namely, he wants to find the maximal value of **F(T)** if **T** can be an arbitrary swap permutation of **S**. Help him and find this value.

### Notes.

Let **S** be some lucky string. Then

- $|S|$  denotes the length of the string **S**;
- $S[i]$  ( $1 \leq i \leq |S|$ ) denotes the  $i^{\text{th}}$  character of **S** (the numeration of characters starts from 1);
- $S[L, R]$  ( $1 \leq L \leq R \leq |S|$ ) denotes the string with the following sequence of characters:  $S[L], S[L + 1], \dots, S[R]$ , and is called a *substring* of **S**. For  $L > R$  we mean by  $S[L, R]$  an empty string.

For any two lucky strings **S** and **T** their *concatenation* is defined as the sequence of characters in **S** followed by the sequence of characters in **T**, and is denoted by **S + T**.

The string **T** is called a *subsequence* of the string **S** if **T** can be derived from **S** by deleting some (possibly zero) number of characters without changing the order of the remaining characters. For example, **T = 474** is a subsequence of **S = 74477747** since after deleting characters at positions **1, 2, 5, 6, 8** from **S** we obtain **T**. Note that, the empty string and the string **S** itself are always the subsequences of **S**.

---

## Input

The first line of the input file contains a single integer **T**, the number of test cases. Each of the following **T** lines contains one string, the string **S** for the corresponding test case. The input file does not contain any whitespaces.

---

## Output

For each test case, output a single line containing the answer for this test case, that is,  $\max\{F(T) : T \text{ is a swap permutation of } S\}$ .

---

## Constraints

$1 \leq T \leq 10$

$1 \leq |S| \leq 100000$

**S** consists only of the lucky digits **4** and **7**.

---

## Example

Input:

5

7474

47

47744

7744

474747447444474

Output:

3  
2  
5  
4  
11

## Explanation

In the first test case all different swap permutations of  $S = 7474$  are  $4747, 4774, 7447, 7474, 7744$ . Corresponding values of  $F(T)$  are  $3, 3, 3, 2, 2$ . Hence the answer is  $3$ .

In the second, third and fourth test cases there exists a non-decreasing swap permutation and hence the answer is equal to the length of the string. Namely,  $S = 47$  is itself non-decreasing, for  $S = 47744$  we can obtain the non-decreasing  $T = 44477$  if we delete the substring  $S[2,3] = 77$  and then insert it back into the end of the string, finally, for  $S = 7744$  we can obtain the non-decreasing  $T = 4477$  if we delete the substring  $S[1,2] = 77$  and then insert it back into the end of the string.

[Home](#) » [Compete](#) » [May Cook-Off 2012](#) » Little Elephant and Filling

## Little Elephant and Filling Problem Code: LUCKFILL

A Little Elephant from the Zoo of Lviv likes *lucky strings*, i.e., the strings that consist only of the lucky digits **4** and **7**.

The Little Elephant has the string  $S$  such that each character in  $S$  is either the lucky digit (**4** or **7**) or the question mark **?**. He can replace each question mark with one of the lucky digits in order to obtain the lucky string. He wants to know the number of different ways he can do this such that the resulting lucky string has no more than  $K$  different substrings. Help him and find this number. Note, that **he need to replace all question marks**.

Consider some example. From the string **47?4?** we can obtain four lucky strings by replacing question marks with the lucky digits: **47444**, **47447**, **47744** and **47747**. The corresponding numbers of different substrings are **11**, **11**, **12** and **11**. For example, all different substrings of the string **47447** are **4**, **7**, **44**, **47**, **74**, **447**, **474**, **744**, **4744**, **7447** and **47447**.

### Notes.

Let **S** be some lucky string. Then

- $|S|$  denotes the length of the string **S**;
- $S[i]$  ( $1 \leq i \leq |S|$ ) denotes the  $i^{\text{th}}$  character of **S** (the numeration of characters starts from 1);
- The string **T** of the length **M** is called a *substring* of **S** if for some **k** from 0 to  $|S| - M$  we have

$$T[1] = S[k + 1], T[2] = S[k + 2], \dots, T[M] = S[k + M].$$


---

## Input

The first line of the input file contains a single positive integer **T**, the number of test cases. **T** test cases follow. The first line of each test case contains two space separated integers **N** and **K**. The second line contains the string **S** of the length **N**.

---

## Output

For each test case output a single line containing the answer for the corresponding test case.

---

## Constraints

$$1 \leq T \leq 4747$$

$$1 \leq N, K \leq 50$$

For each string **S** from the input file we have that  $|S| = N$  and each character in **S** is either the lucky digit (4 or 7) or the question mark ?.

---

## Example

Input:

2

2 2

??

3 7

?4?

Output:

2

4

[Home](#) » [Compete](#) » [May Cook-Off 2012](#) » Little Elephant and CNSes

## Little Elephant and CNSes Problem Code: LUCKYCOM

A Little Elephant from the Zoo of Lviv likes *lucky strings*, i.e., the strings that consist only of the lucky digits **4** and **7**.

The Little Elephant has **N** favorite strings **S<sub>1</sub>**, **S<sub>2</sub>**, ..., **S<sub>N</sub>** and the favorite number **K**. Each character in **S<sub>i</sub>** ( $1 \leq i \leq N$ ) is either the lucky digit (**4** or **7**) or the question mark **?**.

Consider some non-decreasing lucky string **S**. In other words, **S** has one of the following forms: **444...444**, **777...777**, **444...444777...777**. The string **S** is called a *CNS* (plural form is *CNSes*) if we can replace some (possibly zero) number of question marks with the lucky digits in each of the string **S<sub>1</sub>**, **S<sub>2</sub>**, ..., **S<sub>N</sub>** in a such way that the total number of replacements for all strings does not exceed **K** and **S** is a subsequence of each of the strings derived after the replacement.

The Little Elephant wants to know the total number of all different non-empty CNSes. Help him to find this number.

### Notes.

Let **S** be some string (possibly not lucky). Then

- $|S|$  denotes the length of the string **S**;
- $S[i]$  ( $1 \leq i \leq |S|$ ) denotes the  $i^{\text{th}}$  character of **S** (the numeration of characters starts from 1);

The string **T** is called a *subsequence* of the string **S** if **T** can be derived from **S** by deleting some (possibly zero) number of characters without changing the order of the remaining characters. For example, **T = 474** is a subsequence of **S = 74477747??** since after deleting characters at positions **1, 2, 5, 6, 8, 9, 10** from **S** we obtain **T**. Note that, the empty string and the string **S** itself are always the subsequences of **S**.

---

### Input

The first line of the input file contains two space separated integers **N** and **K**, the number of strings in the set and the favorite number of the Little Elephant. Each of the following **N** lines contains one favorite string of the Little Elephant. Namely,  $i^{\text{th}}$  line among these **N** lines contains the string **S<sub>i</sub>**.

---

## Output

For each test case output a single line containing the answer for this test case.

---

## Constraints

$1 \leq N \leq 7474$

$0 \leq K \leq 10^9$

$S_i$  is non-empty for  $1 \leq i \leq N$ .

$|S_1| + |S_2| + \dots + |S_N| \leq 100000$ . In other words, the total length of all  $N$  strings does not exceed 100000.

Each character in  $S_i$  ( $1 \leq i \leq N$ ) is either the lucky digit (4 or 7) or the question mark ?.

---

## Example

Input:

3 2

4447

47????74

4?77?

Output:

5

JUNE12

[June Challenge 2012](#)

01 Jun 2012  
15:00:00

10 days

2692

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Little Elephant and Candies

## Little Elephant and Candies Problem Code: LECANDY

A Little Elephant and his friends from the Zoo of Lviv like candies very much.

There are **N** elephants in the Zoo. The elephant with number **K** ( $1 \leq K \leq N$ ) will be happy if he receives at least **A<sub>K</sub>** candies. There are **C** candies in all in the Zoo.

The Zoo staff is interested in knowing whether it is possible to make all the **N** elephants happy by giving each elephant at least as many candies as he wants, that is, the **K<sup>th</sup>** elephant should receive at least **A<sub>K</sub>** candies. Each candy can be given to only one elephant. Print **Yes** if it is possible and **No** otherwise.

---

## Input

The first line of the input file contains an integer **T**, the number of test cases. **T** test cases follow. Each test case consists of exactly 2 lines. The first line of each test case contains two space separated integers **N** and **C**, the total number of elephants and the total number of candies in the Zoo respectively. The second line contains **N** space separated integers **A<sub>1</sub>, A<sub>2</sub>, ..., A<sub>N</sub>**.

---

## Output

For each test case output exactly one line containing the string **Yes** if it possible to make all elephants happy and the string **No** otherwise. Output is case sensitive. So **do not print YES or yes.**

---

## Constraints

$1 \leq T \leq 1000$

$1 \leq N \leq 100$

$1 \leq C \leq 10^9$

$1 \leq A_K \leq 10000$ , for  $K = 1, 2, \dots, N$

---

## Example

Input:

2

2 3

1 1

3 7

4 2 2

**Output:**

Yes

No

**Explanation**

**Case 1.** We can give one candy to the first elephant and two candies to the second elephant and make them both happy. Hence the answer is **Yes**. Alternatively we can give one candy to each elephant and left one candy for ourselves but they again will be happy.

**Case 2.** Even if we give four candies to the first elephant and two candies to the second elephant we will have only one candy left and can not make last elephant happy since he needs two candies for his happiness. Hence the answer is **No**.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Arranging Cup-cakes

## Arranging Cup-cakes Problem Code: RESQ

Our Chef is catering for a big corporate office party and is busy preparing different mouth watering dishes. The host has insisted that he serves his delicious cupcakes for dessert.

On the day of the party, the Chef was over-seeing all the food arrangements as well, ensuring that every item was in its designated position. The host was satisfied with everything except the cupcakes. He noticed they were arranged neatly in the shape of a rectangle. He asks the Chef to make it as square-like as possible.

The Chef is in no mood to waste his cupcakes by transforming it into a perfect square arrangement. Instead, to fool the host, he asks you to arrange the **N** cupcakes as a rectangle so that the **difference** between the length and the width is minimized.

**Input**

The first line of the input file contains an integer **T**, the number of test cases. Each of the following **T** lines contains a single integer **N** denoting the number of cupcakes.

**Output**

Output  $T$  lines, each indicating the minimum possible difference between the length and the width in a rectangular arrangement of the cupcakes.

---

## Constraints

$1 \leq T \leq 100$

$1 \leq N \leq 10^8$

---

## Example

**Input:**

4

20

13

8

4

**Output:**

1

12

2

0

---

## Explanation

**Case 1:** 20 cupcakes can be arranged in 6 possible ways -  $1 \times 20$ ,  $2 \times 10$ ,  $4 \times 5$ ,  $5 \times 4$ ,  $10 \times 2$  and  $20 \times 1$ . The corresponding differences between the length and the width are 19, 8, 1, 1, 8 and 19 respectively. Hence, 1 is the answer.

**Case 4:** 4 cupcakes can be arranged as a  $2 \times 2$  square. Difference between the length and the width is 0. You can't do anything better than 0.

# Doom Bakes Cakes

Problem Code: CAKEDOOM

From the FAQ:

## What am I allowed to post as a comment for a problem?

- Do NOT post code.
- Do NOT post a comment asking why your solution is wrong.
- Do NOT post a comment asking if you can be given the test case your program fails on.
- Do NOT post a comment asking how your solution can be improved.
- Do NOT post a comment giving any hints or discussing approaches to the problem, or what type or speed of algorithm is required.

## Problem Statement

Chef Doom has decided to bake a circular cake. He wants to place **N** colored cherries around the cake in a circular manner. As all great chefs do, Doom doesn't want any two adjacent cherries to have the same color. Chef has unlimited supply of cherries of **K**  $\leq$  **10** different colors. Each color is denoted by the digit from the set **{0, 1, ..., K – 1}**. Different colors are denoted by different digits. Some of the cherries are already placed and the Chef wants you to place cherries in the remaining positions. He understands that there can be many such arrangements, so in the case when the answer is not unique he asks you to find the lexicographically smallest one.

What does it mean?

Let's numerate positions for the cherries by the numbers **1, 2, ..., N** starting from one of the positions in a clockwise direction. Then the current (possibly partial) arrangement of the cherries can be represented by a string of **N** characters. For each position **i** of the arrangement if the cherry of the color **C** is placed at this position then the **i<sup>th</sup>** character of the string is equal to the digit **C**. Otherwise, it is equal to the question mark **?**. We identify the arrangement with the string that represents it.

One arrangement is called *lexicographically smaller* than the other arrangement if at the first position where they differ the first one has smaller digit (we compare only complete arrangements so we don't care about relation between digits and the question mark). For example, the arrangement **1230123** is lexicographically smaller than **1231230** since they have first **3** equal characters but the **4<sup>th</sup>** character in the first arrangement is **0** and it is less than **1** which is the **4<sup>th</sup>** character of the second arrangement.

## Notes

- The cherries at the first and the last positions are adjacent to each other (recall that we have a circular cake).
- In the case  $N = 1$  any arrangement is valid as long as the color used for the only cherry of this arrangement is less than  $K$ .
- Initial arrangement can be already invalid (see the case 3 in the example).

**Just to make all things clear. You will be given a usual string of digits and question marks. Don't be confused by circular stuff we have in this problem. You don't have to rotate the answer once you have replaced all question marks by the digits. Think of the output like the usual string for which each two consecutive digits must be different but having additional condition that the first and the last digits must be also different (of course if  $N > 1$ ).**

Next, you don't have to use all colors. The only important condition is that this string should be lexicographically smaller than all other strings that can be obtained from the input string by replacement of question marks by digits and of course it must satisfy conditions on adjacent digits.

One more thing,  $K$  here is not the length of the string but the number of allowed colors. Also we emphasize that the given string can have arbitrary number of question marks. So it can have zero number of question marks as well as completely consists of question marks but of course in general situation it can have both digits and question marks.

OK. Let's try to formalize things in order to make all even more clear. You will be given an integer  $K$  and a string  $S=S[1]S[2]...S[N]$  where each  $S[i]$  is either the decimal digit less than  $K$  or the question mark. We are serious. In all tests string  $S$  can have only digits less than  $K$ . Don't ask about what to do if we have digit  $\geq K$ . There are no such tests at all! We guarantee this! OK, let's continue. Your task is to replace each question mark by some digit strictly less than  $K$ . If there were no question marks in the string skip this step. Now if  $N=1$  then your string is already valid. For  $N > 1$  it must satisfy the following  $N$  conditions  $S[1] \neq S[2]$ ,  $S[2] \neq S[3]$ , ...,  $S[N-1] \neq S[N]$ ,  $S[N] \neq S[1]$ . Among all such valid strings that can be obtained by replacement of question marks you should choose lexicographically smallest one. I hope now the problem is really clear.

## Input

The first line of the input file contains an integer  $T$ , the number of test cases.  $T$  test cases follow. Each test case consists of exactly two lines. The first line contains an integer  $K$ , the number of available colors for cherries. The second line contains a string  $S$  that represents the current arrangement of the cherries in the cake.

## Constraints

$1 \leq T \leq 1000$

$1 \leq K \leq 10$

$1 \leq |\mathbf{S}| \leq 100$ , where  $|\mathbf{S}|$  denotes the length of the string  $\mathbf{S}$

Each character in  $\mathbf{S}$  is either the digit from the set  $\{0, 1, \dots, K - 1\}$  or the question mark ?

---

## Output

For each test case output the lexicographically smallest valid arrangement of the cherries in the cake that can be obtained from the given arrangement by replacement of each question mark by some digit from **0** to **K – 1**. If it is impossible to place the cherries output **NO** (output is case sensitive).

---

## Example

Input:

```
7
1
?
2
?0
10
79259?087
2
??
3
0?1
4
?????
3
012
```

Output:

0  
10  
NO  
01  
021  
01012  
012

---

## Explanation

**Case 2.** The only possible replacement here is **10**. Note that we output **10** since we **can not rotate the answer** to obtain **01** which is smaller.

**Case 3.** Arrangement is impossible because cherries at the first and the last positions are already of the same color. Note that **K = 10** but the string has length **9**. It is normal. **K** and **|S|** don't have any connection.

**Case 4.** There are two possible arrangements: **01** and **10**. The answer is the first one since it is lexicographically smaller.

**Case 5.** There are three possible ways to replace question mark by the digit: **001**, **011** and **021**. But the first and the second strings are not valid arrangements as in both of them there exists an adjacent pair of cherries having the same color. Hence the answer is the third string.

**Case 6.** Note that here we do not use all colors. We just find the lexicographically smallest string that satisfies condition on adjacent digit.

**Case 7.** The string is already valid arrangement of digits. Hence we simply print the same string to the output.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Little Elephant and Product

# Little Elephant and Product

Problem Code: LUCKY8

A Little Elephant from the Zoo of Lviv likes lucky numbers very much. Everybody knows that the lucky numbers are positive integers whose decimal representation contains only the lucky digits **4** and **7**. For example, numbers **47, 744, 4** are lucky and **5, 17, 467** are not.

Let  $F_4(X)$  be the number of digits **4** in the decimal representation of **X**, and  $F_7(X)$  be the number of digits **7** in the decimal representation of **X**. For example,  $F_4(456) = 1$ ,  $F_4(444) = 3$ ,  $F_7(1) = 0$ ,  $F_7(747) = 2$ . The Little Elephant wants to know the largest product  $F_4(X) \cdot F_7(X)$ , where  $L \leq X \leq R$ . In other words he wants to know the value

$$\max\{F_4(X) \cdot F_7(X) : L \leq X \leq R\}.$$

## Input

The first line of the input file contains an integer **T**, the number of test cases. **T** test cases follow. The only line of each test case contains two integers **L** and **R** separated by exactly one space.

## Output

For each test case output a single line containing the answer for the corresponding test case.

## Constraints

$$1 \leq T \leq 1000$$

$$1 \leq L \leq R \leq 10^{18}$$

## Example

Input:

3

1 10

1 100

47 74

Output:

0

1

1

## Explanation

In both second and third test cases the maximal product is achieved at the numbers **47** and **74**.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Closest Points

## Closest Points Problem Code: CLOSEST

Crisis at Byte Town!

There have been 100000 (of course, in base 2) instances of bank robberies in the past month! That is, more than one robbery in a day in average and the Byte Town Police Department (BTPD) is dumb founded. Chef, the honorable President of Byte Town, has pledged to solve the problem for good.

Chef investigated and found out that Byte Town has a large number of BTPD response centers and an equally large (if not larger) number of banks. Each bank has an alert mechanism that alerts a BTPD response center assigned to the bank. That response center is then responsible for sending in a strike team. The response center assigned to some bank is called the respondent for this bank. Unfortunately, that is the problem. The current assignment of which BTPD response center responds to an alert by a bank is outdated. Chef has decided to upgrade this system, in lieu of which, you have to solve this problem.

Byte Town, being the quintessential challenge for every programmer, is Euclidean 3D space. Each response center and each bank is represented by some point in this 3D space. There are **N** response centers and **Q** banks in Byte Town. BTPD response centers are labeled by integers **0, 1, ..., N-1** and **i<sup>th</sup>** response center is represented by the point with coordinates **(X<sub>i</sub>, Y<sub>i</sub>, Z<sub>i</sub>)**. Banks are labeled by integers **0, 1, ..., Q-1** and **j<sup>th</sup>** bank is represented by the point with coordinates **(A<sub>j</sub>, B<sub>j</sub>, C<sub>j</sub>)**. **No two buildings (banks, as well as response centers) will ever be at the same point.** Your task is to assign for each bank the BTPD response center that will be the respondent for this bank. **A response center can be the respondent for more than one bank. But every bank should have exactly one respondent.**

Bluntly speaking, your goal is to minimize the Euclidian distance between bank and its respondent. The Euclidian distance between **i<sup>th</sup>** response center and **j<sup>th</sup>** bank is equal to the square root of the number

$$(X_i - A_j)^2 + (Y_i - B_j)^2 + (Z_i - C_j)^2.$$

Denote by  $C[j]$  the closest response center to the  $j^{\text{th}}$  bank and by  $D[j]$  the distance to it. Your absolute goal is to find  $C[j]$  for all banks. **But since it is the challenge problem you can get accepted even if your assignment will be not optimal.** But to get accepted you need to find  $C[j]$  for at least one value of  $j$ . To make things clear we note that  $C[j]$  could be not unique, so by the phrase "**to find  $C[j]$** " we mean to find any response center that has distance  $D[j]$  to the  $j^{\text{th}}$  bank.

---

## Input

The first line of the input file contains an integer  $N$ , the number of BTPD response centers. Each of the following  $N$  lines contains three integers  $X_i$ ,  $Y_i$ ,  $Z_i$ , the coordinates of the  $i^{\text{th}}$  response center. The next line contains an integer  $Q$ , the number of banks. Each of the following  $Q$  lines contains three integers  $A_j$ ,  $B_j$ ,  $C_j$ , the coordinates of the  $j^{\text{th}}$  bank. Each pair of consecutive numbers in every line is separated by exactly one space.

---

## Output

For each of the  $Q$  banks from the input file you should print exactly one line that contains the number from the set  $\{0, 1, \dots, N-1\}$ , the response center that you have assigned to the corresponding bank. So the  $(j+1)^{\text{th}}$  line of the output file should contain the respondent of the  $j^{\text{th}}$  bank. Just to re-iterate, **your output will be considered as correct if and only if you print exactly  $Q$  integers  $I_0, I_1, \dots, I_{Q-1}$ , each from 0 to  $N-1$  (inclusive) such that for at least one value of  $j$  from 0 to  $Q-1$  (inclusive) we have that the  $I_j^{\text{th}}$  response center is the closest response center to the  $j^{\text{th}}$  bank.**

---

## Scoring

Your score for a test file is the total number of banks for which you find the closest possible response center. Formally the score is the number of those values of  $j$  from  $0$  to  $Q-1$  such that the distance between  $j^{\text{th}}$  bank and the response center that you have assigned to it is equal to  $D[j]$ . The total score for a submission is the average score across all the test files.

---

## Constraints

$$1 \leq N, Q \leq 50000$$

Each point in the input file has coordinates with absolute value no more than  $10^9$ . That is,  $|X_i|, |Y_i|, |Z_i|, |A_j|, |B_j|, |C_j| \leq 10^9$

---

## Example

Input:

3

1 0 0

0 0 1

0 1 0

2

0 -1 0

0 0 -1

**Output:**

0

1

## Explanation

Consider the **0<sup>th</sup>** bank. It has coordinates **(0, -1, 0)**. Hence the squares of distances from this bank to response centers are

- to the **0<sup>th</sup>** center:  $(1 - 0)^2 + (0 - (-1))^2 + (0 - 0)^2 = 1 + 1 + 0 = 2$ ;
- to the **1<sup>st</sup>** center:  $(0 - 0)^2 + (0 - (-1))^2 + (1 - 0)^2 = 0 + 1 + 1 = 2$ ;
- to the **2<sup>nd</sup>** center:  $(0 - 0)^2 + (1 - (-1))^2 + (0 - 0)^2 = 0 + 2^2 + 0 = 4$ .

Thus, the closest response centers to the **0<sup>th</sup>** bank are both **0<sup>th</sup>** and **1<sup>st</sup>** response centers. Since in the output we assign **0<sup>th</sup>** response center to this bank we find one of the closest response centers to this bank and it will be counted in our resulting score.

Now consider the **1<sup>st</sup>** bank. It has coordinates **(0, 0, -1)**. Hence the squares of distances from this bank to response centers are

- to the **0<sup>th</sup>** center:  $(1 - 0)^2 + (0 - 0)^2 + (0 - (-1))^2 = 1 + 0 + 1 = 2$ ;
- to the **1<sup>st</sup>** center:  $(0 - 0)^2 + (0 - 0)^2 + (1 - (-1))^2 = 0 + 0 + 2^2 = 4$ ;
- to the **2<sup>nd</sup>** center:  $(0 - 0)^2 + (1 - 0)^2 + (0 - (-1))^2 = 0 + 1 + 1 = 2$ .

Thus, the closest response centers to the **1<sup>st</sup>** bank are both **0<sup>th</sup>** and **2<sup>nd</sup>** response centers. Since in the output we assign **1<sup>st</sup>** response center to this bank we fail to find the closest response center to this bank and it will not be counted in our resulting score.

Thus, total score for this test case would be **1**. Since we find the closest response center to at least one bank such output will be considered as correct.

---

## Test Generation

In each official test file **N = Q = 50000**. Coordinates of banks and response centers are generated using different strategies. In some tests coordinates are uniformly distributed within its range while other tests created specifically to fail certain heuristics.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Polynomial Partition Function

## Polynomial Partition Function Problem Code: PARPOLY

Chef Ciel is participating in an arithmetic contest now. Why? Because of the top prize for the contest, a limited edition kitchen knife.

But to win the contest she must calculate the values  $f(M, N, X)$  of the function named *polynomial partition function*. The polynomial partition function  $f(M, N, X)$  is defined by

$$\sum_{\substack{k_1 + k_2 + \dots + k_M = N \\ k_1, \dots, k_M \text{ are nonnegative integers}}} P(k_1 X) P(k_2 X) \dots P(k_M X)$$

where  $P$  is a given polynomial  $P(x) = C_D \cdot x^D + C_{D-1} \cdot x^{D-1} + \dots + C_1 \cdot x + C_0$ .

For example,

$$f(1, 3, X) = P(3X)$$

$$f(2, 3, X) = P(0) P(3X) + P(X) P(2X) + P(2X) P(X) + P(3X) P(0) = 2 P(X) P(2X) + 2 P(0) P(3X)$$

$$f(3, 1, X) = P(0) P(0) P(X) + P(0) P(X) P(0) + P(X) P(0) P(0) = 3 P(0)^2 P(X)$$

Ciel is a great chef. However, she is not very good at arithmetic. So she needs your help to make her win the contest. For the given values of  $P$ ,  $M$ ,  $N$  and  $X$ , your work is to calculate the value of  $f(M, N, X)$ . The answer can be very large, so you should print the answer modulo  $1000000007$  ( $10^9+7$ ), that is, you need to find the remainder of division of  $f(M, N, X)$  by  $1000000007$  ( $10^9+7$ ).

---

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line for each test case has 3 integers  $M$ ,  $N$  and  $X$ . Then the next line has  $D+2$  integers. The first integer denotes  $D$ , and the  $(i+1)$ -th integer denotes  $C_{i-1}$ .

## Output

For each test case, print the value of  $f(M, N, X)$  modulo  $1000000007$  ( $10^9+7$ ).

---

## Constraints

$1 \leq T \leq 4$   
 $1 \leq M \leq 400$   
 $1 \leq N \leq 800$   
 $0 \leq X \leq 1000000006$  ( $10^9+6$ )  
 $0 \leq D \leq 10$   
 $0 \leq C_i \leq 1000000006$  ( $10^9+6$ )  
If  $D \neq 0$ , then  $C_D \neq 0$

---

## Sample Input

3

1 3 2

2 0 1 2

2 3 0

1 1 1

3 1 1

3 1 2 3 4

---

## Sample Output

78

4

30

---

## Explanation

In the first case, the polynomial is  $P(x) = 2 \cdot x^2 + x$ . The answer is  $P(3X) = P(6) = 2 \cdot 36 + 6 = 78$ .

## Fly-overs Problem Code: FLY

The Head Chef has opened **N** new restaurants across the city. But he is exhausted since he has to travel so much when he has to go from one restaurant to another. Further, the city traffic is driving him crazy. So he wishes to construct fly-overs between restaurants. A fly-over is a road at a raised height between any two restaurants. If two restaurants are connected by a fly-over, it takes only **1** second for the Chef to move from one restaurant to the other. When commuting between restaurants, our Head Chef wishes to use one or more fly-overs. *He doesn't want to travel by the regular roads at all.*

Unfortunately, construction of fly-overs costs money. A construction company has agreed to build **1** fly-over for a payment of **C** rupees. It is up to the Chef to decide which restaurants he wants to connect using fly-overs. He asks his assistant chefs to give him suggestions. Each assistant chef proposes his own model. The Chef is now confused, he has no way of comparing models. Some models have too many fly-overs, while other models have too few.

The Sous Chef came up with a formula to estimate the **Cost of a model**. Suppose the model has **F** fly-overs. Let's label restaurants by numbers **1, 2, ..., N**. Let **T(U, V)** represents the time taken to go from the restaurant **U** to the restaurant **V** via fly-overs of the current model. Compute **T(X, Y)** for every ordered pair of distinct restaurants **(X, Y)** ( $N \cdot (N - 1)$  pairs in total) and then add them up. Denote obtained sum as **P** and called it the time penalty of the model. Note that the constructing of **F** fly-overs will cost **C · F** rupees. The Sous Chef strongly believes that saving time is equivalent to saving money, so the total cost of the model is simply **P + C · F**.

The Head Chef is somewhat satisfied. As a specialist in math and graph theory he immediately found a formula for the cost of the optimal model for a given **C**. But the construction company has a special offer now. It can construct fly-overs between **M** special pairs of restaurants for free. Now the problem of finding the optimal model becomes non-trivial for The Chef and he turns to you for help.

---

### Input

The first line of the input file contains an integer **T**, the number of test cases. **T** test cases follow. The first line of each test case contains an integer **N**, a real **C** and an integer **M**. Here **N** is the number of restaurants, **C** is the cost of the constructing of one fly-over, and **M** is the number of special pairs of restaurants. Each of the next **M** lines contains two integers **U** and **V**, the indexes of the restaurants that can be connected by the fly-over for free. Each pair of consecutive numbers in every line is separated by exactly one space.

---

### Output

For each test case output a single value, the minimum possible cost of a model. Output this value with exactly **9** digits after the decimal point. Your answer will be considered as correct if it has an absolute error less than  **$10^{-12}$** .

---

## Constraints

**$1 \leq T \leq 1000$**

**$1 \leq N \leq 10^9$**

**$0 \leq C \leq 10^9$**

**C** either will be an integer or will have at most **9** digits after the decimal point.

**$0 \leq M \leq 2$**

**$1 \leq U, V \leq N$**

**$U \neq V$**

All **M** unordered pairs **{U, V}** will be different.

**Note:** Just to re-iterate, the cost of a model is not simply the cost of constructing the fly-overs. Refer to the Sous Chef's definition of "Cost of a model" in the problem statement.

---

## Example

**Input:**

3

3 1.500 0

2 18.92 0

3 3.1415926531

1 2

1 47 0

**Output:**

10.500000000

20.920000000

11.141592653

0.000000000

## Explanation

**Case 1.** We will compare two models:  $\mathbf{M}_1$  and  $\mathbf{M}_2$ . We will denote a fly-over that connects the restaurants with indexes  $\mathbf{U}$  and  $\mathbf{V}$  by  $[\mathbf{U}, \mathbf{V}]$ .

In the model  $\mathbf{M}_1$  we have two fly-overs:  $[1, 2]$  and  $[1, 3]$ . Clearly,  $T(1, 2) = T(2, 1) = 1$  and  $T(1, 3) = T(3, 1) = 1$  while  $T(2, 3) = T(3, 2) = 2$  since traveling from 2 to 3 (and vice versa) requires traversing of two fly-overs. Thus, the time penalty is  $1+1+1+1+2+2 = 8$  seconds, while the cost of the fly-overs constructing is  $2 \cdot 1.5 = 3$  rupees. Hence, the total cost is  $8 + 3 = 11$ .

Now let's consider the model  $\mathbf{M}_2$  which has 3 fly-overs:  $[1, 2]$ ,  $[2, 3]$  and  $[1, 3]$ . Clearly,  $T(\mathbf{X}, \mathbf{Y}) = 1$  for every ordered pair  $(\mathbf{X}, \mathbf{Y})$  of the restaurants. Thus, the time penalty is  $1+1+1+1+1+1 = 6$  seconds, while the cost of the fly-overs constructing is  $3 \cdot 1.5 = 4.5$  rupees. Hence, the total cost is  $6 + 4.5 = 10.5$  which is smaller than that of  $\mathbf{M}_1$ . It can be shown that no model has a cost smaller than **10.5**.

**Case 2.** Recall that the Chef wishes to travel only via fly-overs. In the case of 2 restaurants, there obviously has to be a fly-over between them. So the cost is  $T(1, 2) + T(2, 1) + 1 \cdot 18.92 = 1 + 1 + 18.92 = 20.92$ .

**Case 3.** The optimal model here has fly-overs  $[1, 2]$  and  $[1, 3]$ . But since the first fly-over will be constructed for free, the total cost is  $8 + 1 \cdot 3.141592653 = 11.141592653$ .

**Case 4.** Note that  $\mathbf{C}$  here is represented without decimal point. Clearly, when the Head Chef has only one restaurant he doesn't need to travel at all and construct fly-overs. Hence the optimal model is an empty model that has zero total cost.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Magic Hull

## Magic Hull Problem Code: HULL

You are given a set of  $\mathbf{N}$  two-dimensional non-zero vectors  $(\mathbf{X}_1, \mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \dots, (\mathbf{X}_N, \mathbf{Y}_N)$  with integer coordinates. You can choose no more than **6** vectors from this set to form a **non-strictly convex polygon** that have sides equal to these vectors (see paragraph below for more detailed explanation). Non-strictly convex polygon is the polygon that can have flat angles (angles of **180** degrees). You can select each vector several times if

needed. Your goal is to maximize the area of this polygon. It is guaranteed that at least one such convex polygon can be constructed.

How exactly we construct a polygon from the given sequence of vectors? Suppose you've chosen the sequence of vectors  $(\mathbf{U}_1, \mathbf{V}_1), (\mathbf{U}_2, \mathbf{V}_2), \dots, (\mathbf{U}_K, \mathbf{V}_K)$ , where  $3 \leq K \leq 6$ . Each vector here should be equal to one of the given  $\mathbf{N}$  vectors but some vectors in this sequence can coincide. At first, you take some point  $(\mathbf{A}_1, \mathbf{B}_1)$  at the plane as the first vertex of your polygon. Then you put your first vector  $(\mathbf{U}_1, \mathbf{V}_1)$  to this point to obtain the second vertex  $(\mathbf{A}_2, \mathbf{B}_2) = (\mathbf{A}_1 + \mathbf{U}_1, \mathbf{B}_1 + \mathbf{V}_1)$ . Next you take the second vector and put it to the second vertex to obtain the third vertex and so on. Finally, at the last step you put vector  $(\mathbf{U}_K, \mathbf{V}_K)$  to the  $K^{\text{th}}$  vertex  $(\mathbf{A}_K, \mathbf{B}_K)$  to obtain the point  $(\mathbf{A}_{K+1}, \mathbf{B}_{K+1}) = (\mathbf{A}_K + \mathbf{U}_K, \mathbf{B}_K + \mathbf{V}_K)$  that should coincide with the first vertex  $(\mathbf{A}_1, \mathbf{B}_1)$ , otherwise we should reject this sequence of vectors. Hence in the end we obtain a polygon with vertexes  $(\mathbf{A}_1, \mathbf{B}_1), (\mathbf{A}_2, \mathbf{B}_2), \dots, (\mathbf{A}_K, \mathbf{B}_K)$ . If this polygon is simple (without self-intersections) and non-strictly convex, we should consider its area as the candidate for the answer. By self-intersection we mean that either some non-consecutive sides (considered with their ends) have common point or two consecutive sides have more than one common point.

## Input

The first line of the input file contains an integer  $\mathbf{N}$ , the number of the given vectors. Each of the following  $\mathbf{N}$  lines contains two space separated integers  $\mathbf{X}_i, \mathbf{Y}_i$ , coordinates of the corresponding vector.

## Output

In the only line of the output file print the maximal possible area of the convex polygon that can be constructed by the rules described in the problem statement with exactly one digit after the decimal point.

## Constraints

$3 \leq \mathbf{N} \leq 150$

$|\mathbf{X}_i|, |\mathbf{Y}_i| \leq 1000000$

All vectors are non-zero:  $|\mathbf{X}_i| + |\mathbf{Y}_i| > 0$

There exists at least one sequence of at most 6 vectors from this set that forms a convex polygon.

## Example

Input:

4

3 0

-1 1

-1 0

-1 -1

**Output:**

2.0

## Explanation

The only non-strictly convex polygon you can construct from these vectors by the rules described in the problem statement is the [isosceles trapezoid](#) that has height of length **1** and bases of length **1** and **3**. It has area **2.0**.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Cool Numbers

## Cool Numbers Problem Code: COOLNUM

Consider a positive integer **N**. Denote its decimal digits by  $X_1, X_2, \dots, X_k$ . **N** is called a **Cool Number** if you can select at most three (but at least one) of its digits such that the following number is divisible by **N**:

$$(X_1 + X_2 + \dots + X_k - S)^s$$

where **S** is the sum of the selected digits.

Let **LC(N)** be the largest Cool Number which is not greater than **N**, and **UC(N)** be the smallest Cool Number which is greater than **N**. So the following inequalities hold

$$LC(N) \leq N < UC(N).$$

Your task is to find for the given positive integer **N** the values of **LC(N)** and **UC(N)**. As you will see from the examples **LC(N)** and **UC(N)** exist for every positive integer **N**.

Consider some examples.

- **1458** is a Cool Number since  $((1 + 4 + 5 + 8) - (1 + 5))^{1+5} = 12^6 = 2985984$  is divisible by **1458**. Note that here we select two digits: **1** and **5**.
- All numbers of the form **10<sup>n</sup>** ( $n = 0, 1, 2, \dots$ ) are Cool. Indeed, if we select just one digit **1** (the first digit of **10<sup>n</sup>**) in the definition of Cool Number we obtain the number  $(1 + 0 + \dots + 0 - 1)^1 = 0^1 = 0$  and it is divisible by **10<sup>n</sup>** (recall that **0** is divisible by every positive integer). Now consider some positive

integer **N** of **n** digits. Then, clearly,  $10^{n-1} \leq N < 10^n$  where both  $10^{n-1}$  and  $10^n$  are Cool Numbers. Hence **LC(N)** and **UC(N)** always exist and, moreover,  $LC(N) \geq 10^{n-1}$  and  $UC(N) \leq 10^n$ .

---

## Input

The first line of the input file contains an integer **T**, the number of test cases. Each of the following **T** lines contains a positive integer **N**.

---

## Output

For each test case output on a single line values of **LC(N)** and **UC(N)** separated by a single space.

---

## Constraints

$1 \leq T \leq 100000$

$1 \leq N \leq 10^{1000}$

The total number of digits in all numbers **N** in the input file does not exceed **4000000**.

---

## Example

Input:

```
4
2
999
1459
18898888
```

Output:

```
2 3
999 1000
```

1458 1460

18895680 18900000

---

## Explanation

**Case 1.** The number **2** is the largest integer which is not greater than **N = 2**, the number **3** is the smallest integer which is greater than **N = 2**, and they both are Cool Numbers, which can be shown similar to the case of **10<sup>n</sup>** in the example above.

**Case 2.** It is very similar to the first test case and we already know that **1000** is a Cool Number. So we just have to show that **999** is a Cool number. But it is clear, since we can select all the three digits so  $((9 + 9 + 9) - (9 + 9 + 9))^{9+9+9} = 0^{27} = 0$  which is divisible by **999**.

Unfortunately, we can't provide you with the explanation of the third and the fourth test cases. You should figure it out by yourself. Please, don't ask about this in comments.

---

## Warning!

There are large input and output files in this problem. Make sure you use fast enough I/O methods.

[Home](#) » [Compete](#) » [June Challenge 2012](#) » Expected Maximum Matching

## Expected Maximum Matching Problem Code: MATCH

You are given two sets of vertexes  $U = \{U[1], U[2], \dots, U[N]\}$  and  $V = \{V[1], V[2], \dots, V[M]\}$ . All  $N + M$  vertexes here are different. You are also given the matrix  $P$  of size  $N \times M$  composed of real numbers from the segment  $[0, 1]$ . The number that stands at the intersection of the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column of this matrix is denoted by  $P[i][j]$  and means the probability that the vertexes  $U[i]$  and  $V[j]$  are connected by the bidirected edge. In other words, you are given a [complete bipartite graph](#) where each edge occurs with some probability. Your task is to find **the expected size of the maximum matching** of this graph.

What exactly does it mean?

Let's call our complete bipartite graph with random edges as *random bipartite graph*. Consider some arbitrary [bipartite graph](#)  $G = (U, V, E)$ . Denote by  $P(G)$  the probability that our random bipartite graph is equal to  $G$ . How to calculate  $P(G)$ ? Consider some pair of vertexes  $(U[i], V[j])$ . If these vertexes are connected by the edge in  $G$  then put  $P_G[i][j] = P[i][j]$  otherwise put  $P_G[i][j] = 1 - P[i][j]$ . Then  $P(G)$  is equal to the product of  $P_G[i][j]$  for all  $N \cdot M$  pairs of  $(i, j)$ , where  $1 \leq i \leq N$  and  $1 \leq j \leq M$ .

Next, denote by **MM(G)** the size of the maximum matching in the graph **G**. In other words, **MM(G)** is the size of the largest set of edges of **G** such that no two edges share a common vertex.

Finally, **the expected size of the maximum matching** is the sum of products  $P(G) \cdot MM(G)$  for all possible bipartite graphs **G** with parts **U** and **V**. Note that there are  $2^{N \cdot M}$  such graphs in all. So don't try to calculate the answer directly by definition if you do not want to get verdict **Time Limit Exceeded** ;)

---

## Input

The first line of the input file contains two integers **N** and **M**. Each of the following **N** lines contains **M** real numbers.  $j^{\text{th}}$  number in the  $i^{\text{th}}$  line of these **N** lines denotes  $P[i][j]$ , the probability that the vertexes **U[i]** and **V[j]** are connected by the direct edge. Each pair of consecutive numbers in every line is separated by exactly one space.

---

## Output

In the only line of the output file print the expected size of the maximum matching. Your answer will be considered as correct if it has an absolute error less than  $10^{-6}$ .

---

## Constraints

$1 \leq N \leq 5$

$1 \leq M \leq 100$

$0 \leq P[i][j] \leq 1$

$P[i][j]$  will have exactly 5 digits after the decimal point

---

## Example

Input 1:

3 3

0.38064 0.30000 0.29486

0.41715 0.90000 0.67837

0.53316 1.00000 1.00000

Output 1:

2.575940

Input 2:

2 2

0.40000 1.00000

0.10000 1.00000

Output 2:

1.46

## Explanation

In the second example we have four different graphs with non-zero value of  $P(G)$ . Their adjacent matrices with marked maximum matching as well as the values of  $MM(G)$  and  $P(G)$  are listed in the table below.

Adjacent matrix	MM(G)	P(G)
$\begin{array}{ c c } \hline 0 & 1 \\ \hline 0 & 1 \\ \hline \end{array}$	1	$(1 - 0.4) \cdot (1 - 0.1) = 0.6 \cdot 0.9 = 0.54$
$\begin{array}{ c c } \hline 1 & 1 \\ \hline 0 & 1 \\ \hline \end{array}$	2	$0.4 \cdot (1 - 0.1) = 0.4 \cdot 0.9 = 0.36$
$\begin{array}{ c c } \hline 0 & 1 \\ \hline 1 & 1 \\ \hline \end{array}$	2	$(1 - 0.4) \cdot 0.1 = 0.6 \cdot 0.1 = 0.06$

1	1		2	0.4 · 0.1 = 0.04
1	1			

So the answer is  $0.54 \cdot 1 + 0.36 \cdot 2 + 0.06 \cdot 2 + 0.04 \cdot 2 = 1.46$ .

COOK23

[June Cook-Off  
2012](#)

17 Jun 2012  
21:30:00

2 hours 55 minutes

868

[Home](#) » [Compete](#) » [June Cook-Off 2012](#) » Closing the Tweets

## Closing the Tweets Problem Code: TWTCLOSE

Little kids, Jack and Evan like playing their favorite game Glass-and-Stone. Today they want to play something new and came across Twitter on their father's laptop.

They saw it for the first time but were already getting bored to see a bunch of sentences having at most 140 characters each. The only thing they liked to play with it is, closing and opening tweets.

There are **N** tweets on the page and each tweet can be opened by clicking on it, to see some statistics related to that tweet. Initially all the tweets are closed. Clicking on an open tweet closes it and clicking on a closed tweet opens it. There is also a button to close all the open tweets. Given a sequence of **K** clicks by Jack, Evan has to guess the total number of open tweets just after each click. Please help Evan in this game.

---

### Input

First line contains two integers **N K**, the number of tweets (numbered 1 to **N**) and the number of clicks respectively ( $1 \leq N, K \leq 1000$ ). Each of the following **K** lines has one of the following.

- CLICK *X* , where *X* is the tweet number ( $1 \leq X \leq N$ )
- CLOSEALL

---

### Output

Output **K** lines, where the *i*<sup>th</sup> line should contain the number of open tweets just after the *i*<sup>th</sup> click.

---

### Example

Input:

3 6

CLICK 1

CLICK 2

CLICK 3

CLICK 2

CLOSEALL

CLICK 1

**Output:**

1

2

3

2

0

1

**Explanation:**

Let  $\text{open}[x] = 1$  if the  $x^{\text{th}}$  tweet is open and 0 if its closed.

Initially  $\text{open}[1..3] = \{ 0, 0, 0 \}$ . Here is the state of  $\text{open}[1..3]$  after each click and corresponding count of open tweets.

CLICK 1 :  $\{ 1, 0, 0 \}$ , open count = 1

CLICK 2 :  $\{ 1, 1, 0 \}$ , open count = 2

CLICK 3 :  $\{ 1, 1, 1 \}$ , open count = 3

CLICK 2 :  $\{ 1, 0, 1 \}$ , open count = 2

CLOSEALL :  $\{ 0, 0, 0 \}$ , open count = 0

CLICK 1 :  $\{ 1, 0, 0 \}$ , open count = 1

# Connecting Soldiers

Problem Code: NOKIA

To protect people from evil, a long and tall wall was constructed a few years ago. But just a wall is not safe, there should also be soldiers on it, always keeping vigil. The wall is very long and connects the left and the right towers. There are exactly **N** spots (numbered 1 to **N**) on the wall for soldiers. The **K**<sup>th</sup> spot is **K** miles far from the left tower and **(N+1-K)** miles from the right tower.

Given a permutation of spots **P** of {1, 2, ..., **N**}, soldiers occupy the **N** spots in that order. The **P[i]**<sup>th</sup> spot is occupied before the **P[i+1]**<sup>th</sup> spot. When a soldier occupies a spot, he is connected to his nearest soldier already placed to his left. If there is no soldier to his left, he is connected to the left tower. The same is the case with right side. A connection between two spots requires a wire of length equal to the distance between the two.

The realm has already purchased a wire of **M** miles long from Nokia, possibly the wire will be cut into smaller length wires. As we can observe, the total length of the used wire depends on the permutation of the spots **P**. Help the realm in minimizing the length of the unused wire. If there is not enough wire, output -1.

## Input

First line contains an integer **T** (number of test cases,  $1 \leq T \leq 10$  ). Each of the next **T** lines contains two integers **N M**, as explained in the problem statement ( $1 \leq N \leq 30$  ,  $1 \leq M \leq 1000$ ).

## Output

For each test case, output the minimum length of the unused wire, or -1 if the the wire is not sufficient.

## Example

Input:

4

3 8

3 9

2 4

5 25

**Output:**

0

0

-1

5

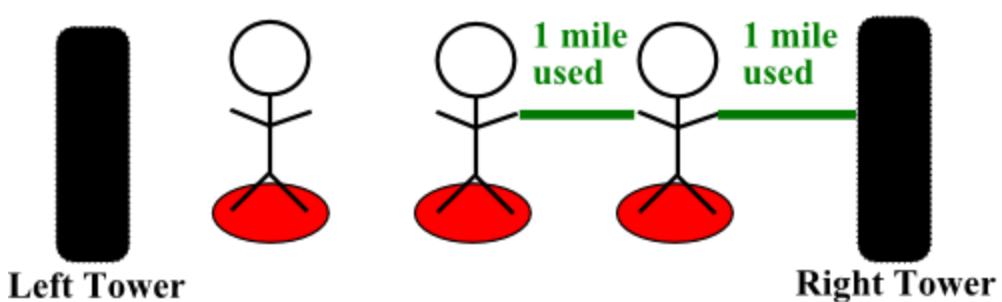
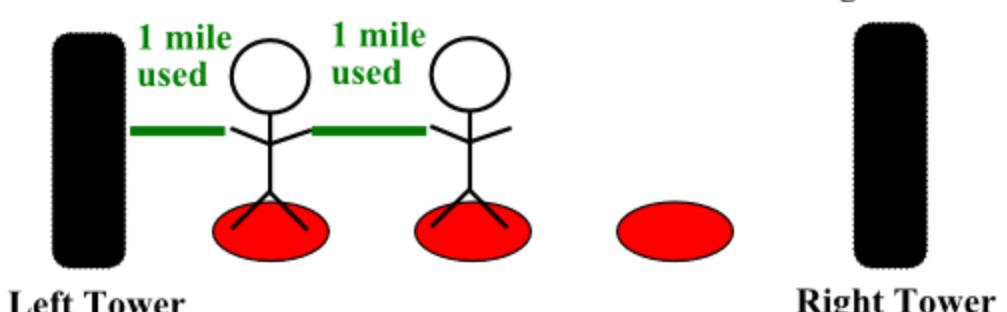
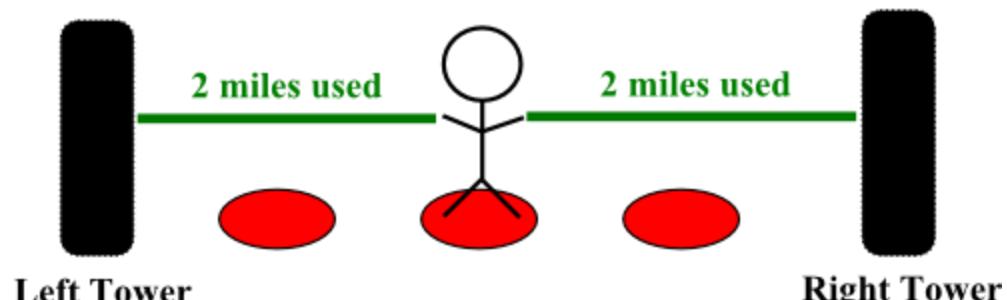
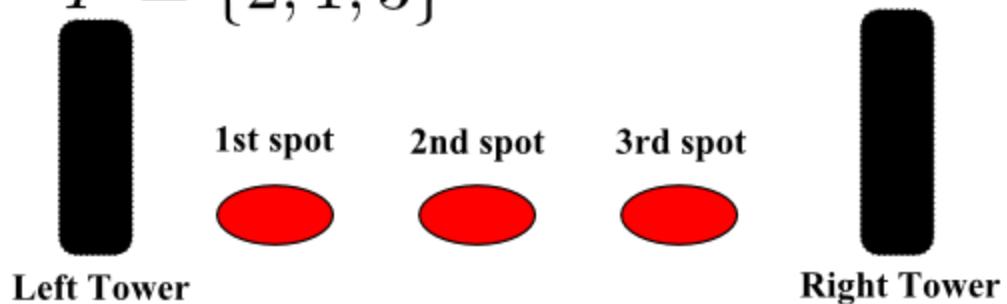
**Explanation:**

In the 1st case, for example, the permutation  $P = \{2, 1, 3\}$  will use the exact 8 miles wires in total.

In the 2nd case, for example, the permutation  $P = \{1, 3, 2\}$  will use the exact 9 miles wires in total.

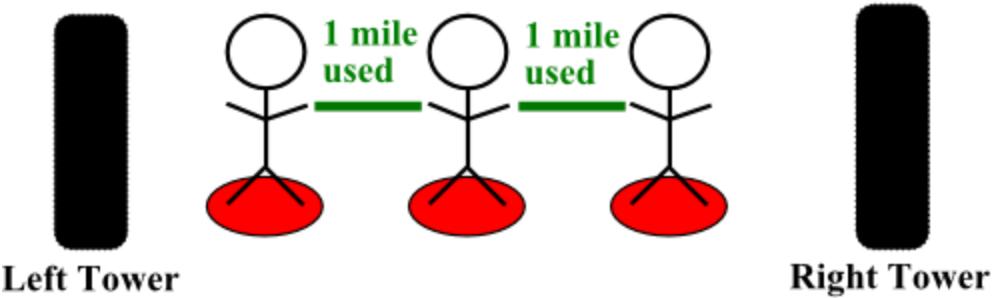
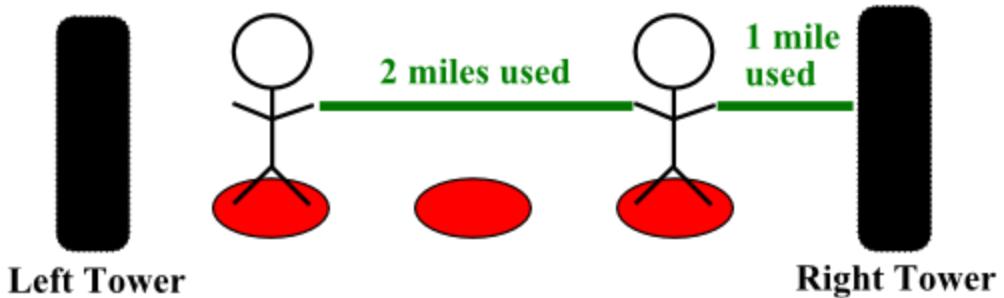
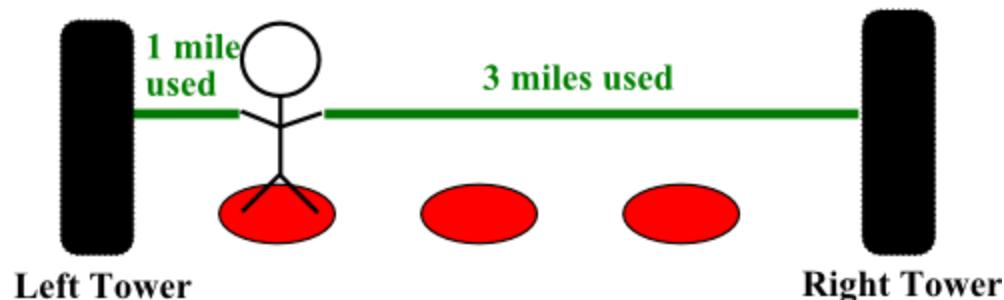
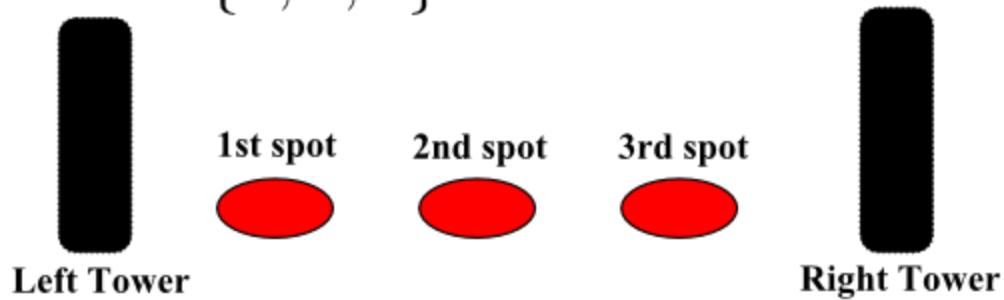
To understand the first two cases, you can see the following figures:

$$P = \{2, 1, 3\}$$



**In total:  $2+2+1+1+1+1 = 8$  miles used**

$$P = \{1, 3, 2\}$$



**In total:  $1+3+2+1+1+1 = 9$  miles used**

In the 3rd case, the minimum length of wire required is 5, for any of the permutations {1,2} or {2,1}, so length 4 is not sufficient.

In the 4th case, for the permutation  $\{1, 2, 3, 4, 5\}$  we need the maximum length of the wire = 20. So minimum possible unused wire length =  $25 - 20 = 5$ .

## Golden Trees Problem Code: GROWGOLD

---

Long ago there was a beautiful kingdom in the island of Sona, the golden island, deep inside Africa. The trees in Sona island are made of gold and farmers are the richest group of people and are also heavy tax payers.

As you know that price of gold increases every year, the minister of Sona has proposed the following tax policy.

- Pay **initTax** units of gold in the first year.
- In each of the next **slot1** years, pay one unit of gold more than the previous year.
- In each of the next **slot2** years, pay double the units of gold of the previous year.
- In each of the following years, pay number of gold units equal to the product of the number of units paid in **K** recent years.

Given an integer **N**, find the number of units of gold to be paid in the **N<sup>th</sup>** year. This result can be huge, so output the result modulo  $100000007 (10^8+7)$ .

---

### Input

First line has an integer **T** (number of test cases,  $1 \leq T \leq 3$ ). Each of the next **T** lines has 5 integers, **initTax slot1 slot2 K N**.

$1 \leq \text{initTax, slot1, slot2} \leq 50$

$1 \leq K \leq \text{slot1 + slot2 + 1}$

$1 \leq N \leq 1000000000 (10^9)$

---

### Output

For each test case, output the tax in units of gold to be paid in the **N<sup>th</sup>** year modulo  $100000007 (10^8+7)$ .

---

### Example

**Input:**

3

1 3 2 4 4

1 3 2 4 7

1 3 2 4 9

**Output:**

4

1536

18811834

**Explanation:**

Let  $\text{tax}[i]$  be the tax paid in  $i^{\text{th}}$  year.

Init :  $\text{tax}[1] = 1$

Next 3 years in slot1 ( $\text{tax}[i] = \text{tax}[i-1] + 1$ ) :  $\text{tax}[2] = 2$  ,  $\text{tax}[3] = 3$ ,  $\text{tax}[4] = 4$

Next 2 years in slot2 ( $\text{tax}[i] = 2 * \text{tax}[i-1]$ ) :  $\text{tax}[5] = 8$  ,  $\text{tax}[6] = 16$

We have  $\text{tax}[1..6] = \{ 1, 2, 3, 4, 8, 16 \}$

$\text{tax}[7] = \text{tax}[3] * \text{tax}[4] * \text{tax}[5] * \text{tax}[6] = 3 * 4 * 8 * 16 = 1536$

similarly,  $\text{tax}[9] = \text{tax}[5] * \text{tax}[6] * \text{tax}[7] * \text{tax}[8]$ .

Do not forget to print only the remainder of the result when divided by  $100000007$  ( $10^8 + 7$ ).

[Home](#) » [Compete](#) » [June Cook-Off 2012](#) » Alien Chefs

## Alien Chefs

Problem Code: [DOWNLOAD](#)

---

The aliens living in outer space are very advanced in technology, intelligence and everything, except one, and that is Cooking. Each year they spend millions of dollars in research, to crack famous recipes prepared by humans.

Recently they came to know about Khana-Academy, a non-profit organization streaming free cooking lesson videos on earth. There are **N** recipes, numbered 1 to **N**, and the video of the  $i^{\text{th}}$  recipe is live in the time interval  $[\mathbf{S}_i, \mathbf{E}_i]$ . An alien can visit earth but can not survive for more than just a small moment (earth is so advanced in pollution). An alien visits the earth at an integer time **t** and instantly downloads the complete video of all the lessons that are live at that moment of time **t** and leaves earth immediately. You are given the visiting times of a small group of **K** aliens. Find the number of different recipes aliens can learn by watching the downloaded videos. Not just one group of aliens, there are **Q** such groups, so you have to find the answer for each of these **Q** groups.

## Input

The first line has an integer **N**. Each of the following **N** lines has two integers **S<sub>i</sub>** **E<sub>i</sub>**. The next line has an integer **Q**, the number of groups. Each of the following **Q** lines has information of a group of aliens. The first integer is **K**, the number of aliens in that group, followed by **K** integers in the same line, the integer visiting times **t** of the aliens.

$1 \leq N \leq 100000 (10^5)$   
 $1 \leq Q \leq 5000 (5 \cdot 10^3)$   
 $1 \leq K \leq 20$   
 $1 \leq S_i, E_i, t \leq 1000000000 (10^9)$   
 $S_i < E_i$

---

## Output

For each of the **Q** groups, output the number of different recipes that group of aliens can learn by watching the downloaded videos.

---

## Example

Input:

4  
1 4  
3 10  
2 6  
5 8  
3  
1 5  
2 2 6  
3 1 10 9

Output:

3

4

2

**Explanation:**

Given videos of 4 recipes in the following closed intervals.

1. [ 1 , 4 ]
2. [ 3 , 10 ]
3. [ 2 , 6 ]
4. [ 5 , 8 ]

In the first query, only one alien arrives at  $t = 5$  and can download 3 recipes 2, 3, 4.

In the second query, two aliens arrive at  $t = 2$  and 6. They can learn all the 4 recipes.

In the third query, three aliens arrive at  $t = 1, 10$  and 9. They can learn only two recipes, 1 and 2.

[Home](#) » [Compete](#) » [June Cook-Off 2012](#) » [Gifts at Olympics](#)

## **Gifts at Olympics** Problem Code: OLYMPIC

---

The Olympic Games, world's foremost sports competition, will be hosted by London next month. Tens of thousands of participants from over 200 nations will be participating this year. Each game has only 3 winners, so the organizers have decided to give special chocolate gift pack to each participant.

A huge number of chocolates have just arrived from **N** different villages, numbered 1 to **N**. The organizers have decided the number of chocolates to be given to each participant. This is given in **count**, where **count**[*i*] is the number of chocolates from *i*<sup>th</sup> village, to be placed in each gift pack. Now they need to order boxes for these chocolates. A box of size **S** can store at most **S** chocolates. A gift pack is made as follows.

Each of the **N** villages is assigned a unique colored box and chocolates from a village are placed only in boxes assigned to it. The organizers always want to minimize the number of boxes in each gift pack. E.g., for given **N** = 3 villages and **count** = {3, 12, 7}, suppose colors assigned are A, B, C. If size of each box is **S** = 4, then we need 1 box of color A, 3 boxes of color B and 2 boxes of color C. All the boxes for a gift pack are then packed in a row using transparent wrapper. So in the above example, a gift pack can be "ABBBCC" or "ABCBCB" etc.

To make everyone feel special, no two participants are given same gift packs. Two gift packs are different if and only if they have the different order of colored boxes, even in reverse order. For example "ABBBCC" is same as "ABBBCC" and also same as "CCBBBA". Given **Q** queries, each specifying the size of a box **S**, find the total number of different gift packs possible if we use boxes of size **S** each. Output the result modulo 1000000007 ( $10^9 + 7$ ).

## Input

The first line has an integer **N**. The second line has the array **count**, **N** space separated integers. The next line has integer **Q**, followed by **Q** lines, each having an integer size **S**.

$1 \leq N, \text{count}[i], Q \leq 100000 (10^5)$   
 $1 \leq \text{sum of all } \text{count}[i] \leq 100000 (10^5)$   
 $1 \leq S \leq 1000000000 (10^9)$

---

## Output

Output **Q** lines, one for each query, the total number of different gift packs possible, using boxes of size **S**, specified in that query modulo  $1000000007 (10^9+7)$ .

---

## Example

**Input:**

3  
5 3 12  
4  
4  
5  
2  
12

**Output:**

30  
10  
2320  
3

**Explanation:**

In the last query, size of each box **S** = 12.

Suppose the colors assigned to the 3 villages are **A**, **B**, **C** respectively, we need 1 box of each color.

Only 3 different gift packs can be made. They are "**ABC**", "**ACB**", "**BAC**".

JULY12	<a href="#">July Challenge 2012</a>	01 Jul 2012 15:00:00	10 days	1901
--------	-------------------------------------	-------------------------	---------	------

[Home](#) » [Compete](#) » [July Challenge 2012](#) » [Gift Rift](#)

## **Gift Rift** Problem Code: SAD

---

Our Chef is very happy that his son was selected for training in one of the finest culinary schools of the world. So he and his wife decide to buy a gift for the kid as a token of appreciation. Unfortunately, the Chef hasn't been doing good business lately, and is in no mood on splurging money. On the other hand, the boy's mother wants to buy something big and expensive. To settle the matter like reasonable parents, they play a game.

They spend the whole day thinking of various gifts and write them down in a huge matrix. Each cell of the matrix contains the gift's cost. Then they decide that the mother will choose a row number **r** while the father will choose a column number **c**, the item from the corresponding cell will be gifted to the kid in a couple of days.

The boy observes all of this secretly. He is smart enough to understand that his parents will ultimately choose a gift whose cost is **smallest** in its **row**, but **largest** in its **column**. If no such gift exists, then our little chef has no option but to keep guessing. As the matrix is huge, he turns to you for help.

He knows that sometimes the gift is not determined uniquely even if a gift exists whose cost is smallest in its row, but largest in its column. However, since the boy is so smart, he realizes that the gift's **cost** is determined **uniquely**. Your task is to tell him the gift's **cost** which is smallest in its row, but largest in its column, or to tell him no such gift exists.

---

### **Input**

First line contains two integers **R** and **C**, the number of rows and columns in the matrix respectively. Then follow **R** lines, each containing **C** space separated integers - the costs of different gifts.

---

### **Output**

Print a single integer - a value in the matrix that is smallest in its row but highest in its column. If no such value exists, then print "GUESS" (without quotes of course)

---

## Constraints

$1 \leq R, C \leq 100$

All gift costs are positive and less than 100000000 (10^8)

---

## Example 1

Input:

2 3

9 8 8

2 6 11

Output:

8

---

## Example 2

Input:

3 3

9 8 11

2 6 34

5 9 11

Output:

GUESS

---

### Example 3

Input:

2 2

10 10

10 10

Output:

10

---

### Explanation of Sample Cases

**Example 1:** The first row contains 9, 8, 8. Observe that both 8 are the minimum. Considering the first 8, look at the corresponding column (containing 8 and 6). Here, 8 is the largest element in that column. So it will be chosen.

**Example 2:** There is no value in the matrix that is smallest in its row but largest in its column.

**Example 3:** The required gift in matrix is not determined uniquely, but the required cost is determined uniquely.

[Home](#) » [Compete](#) » [July Challenge 2012](#) » My Fair Coins

## My Fair Coins Problem Code: CSUMD

---

There are coins of 22 different denominations in Crazyland, 11-cent coins and 22-cent coins. They have two faces - heads and tails.

Your task is to find out the the number of ways to create a linear arrangement of these coins so that their sum is  $NN$  cents. The only condition on the linear arrangement is that the first coin in the arrangement should always have heads up. All other coins could have either tails or heads up.

Take  $N=2N=2$  as an example. The possible arrangements are  $(1H,1H)(1H,1H)$ ,  $(2H)(2H)$ ,  $(1H,1T)(1H,1T)$ , where  $HH$  is heads and  $TT$  is tails. Therefore there are 33 possible arrangements that sum up to 22-cents.

Note: While counting make sure to differentiate between heads and tails.

---

## Input

- First line contains a single integer -  $T$ , the number of test cases.
  - $T$  lines follow, each containing a single integer  $N$ , the required sum.
- 

## Constraints

- $0 \leq T \leq 10$
  - $1 \leq N \leq 10^9$
- 

## Output

For each case the output should be a single integer representing the number of such arrangements possible. As the answer can be very large, print it modulo  $10^9 + 7$ .

---

## Example Input

3  
1  
2  
3

---

## Example Output

1  
3  
8

[Home](#) » [Compete](#) » [July Challenge 2012](#) » Chef's Dream

## Chef's Dream Problem Code: DREAM

---

The Chef is sleeping now. He tries to cook new kind of meals in his dream.

These meals are arranged in a row and numbered from 1 to  $N$  consecutively. For each meal  $i$  ( $1 \leq i \leq N$ ) there is given one integer  $f(i)$  which denotes the time needed to cook it. Initially, all meals are uncooked. Each assistant of The Chef (there are infinite number of

them) can help him with cooking.

The abilities of all assistants are same. There can be at most one assistant cooking at each moment. He must choose some continuous subsequence of meals with length  $K$  (any such subsequence can be chosen). And if there are uncooked meals in it, he will cook all uncooked meals which has the minimum cooking time among uncooked meals in the chosen subsequence. Nothing done to another meals.

The dream was so interesting that he tried to solve such a problem: What is the minimum number of assistants which can cook all the meals assuming that each of them will cook at most once? But since the bell rings and Chef's friends has come to visit him, he will wake up after 2 seconds. Your program should calculate the answer before The Chef will come to himself.

## Input

First line of input file contains two integers  $N$  ( $1 \leq N \leq 10^5$ ) and  $K$  ( $1 \leq K \leq N$ ), followed by a line containing  $N$  integers. The  $i^{th}$  integer denotes  $f(i)$ -the cooking time of meal number  $i$  ( $1 \leq f(i) \leq 10^9$ )

## Output

Print minimum number of assistants which can cook all the meals in one line.

## Example

Input:

5 3

40 30 40 30 40

Output:

3

### Explanation:

3 assistants are enough to cook all the meals. They can work in following schedule:

1<sup>st</sup> assistant chooses interval [2,4] and cooks meals 2 and 4.

2<sup>nd</sup> assistant chooses interval [1,3] and cooks meals 1 and 3.

3<sup>rd</sup> assistant chooses interval [3,5] and cooks meal 5.

Other schedules can also be possible.

# Restaurant Rating

Problem Code: RRATING

Chef has opened up a new restaurant. Like every other restaurant, critics critique this place. The Chef wants to gather as much positive publicity as he can. Also, he is very aware of the fact that people generally do not tend to go through all the reviews. So he picks out the positive reviews and posts it on the website of the restaurant. A review is represented by an integer which is the overall rating of the restaurant as calculated by that particular review. A review is considered to be positive if it is among the top one-third of the total reviews when they are sorted by their rating. For example, suppose the ratings given by 8 different reviews are as follows:

2 8 3 1 6 4 5 7

Then the top one-third reviews will be 8 and 7. Note that we considered one-third to be  $8/3=2$  top reviews according to integer division. (see Notes)

So here is what the Chef wants from you: Given the reviews(ratings) by different critics, the Chef needs you to tell him what is the minimum rating that his website will be displaying. For example in the above case, the minimum rating that will be displayed is 7. Also, different critics keep reviewing the restaurant continuously. So the new reviews keep coming in. The Chef wants your website ratings to be up-to-date. So you should keep updating the ratings there. At any point of time, the Chef might want to know the minimum rating being displayed. You'll have to answer that query. An interesting thing to note here is that a review might be in the website for some time and get knocked out later because of new better reviews and vice-versa.

Notes: To be precise, the number of reviews displayed website will be  $\text{floor}(n / 3)$ , where n denotes the current number of all reviews.

---

## Input

First line of the input file consists of a single integer N, the number of operations to follow. The next N lines contain one operation each on a single line. An operation can be of 2 types:

- 1 x : Add a review with rating 'x' to the existing list of reviews (x is an integer)
  - 2 : Report the current minimum rating on the website
- 

## Output

For every test case, output a single integer each for every operation of type 2 mentioned above. If no review qualifies as a positive review, print "No reviews yet".

---

## Constraints

$1 \leq N \leq 250000$

$1 \leq x \leq 10000000000$

---

## Example

**Input:**

10

1 1

1 7

2

1 9

1 21

1 8

1 5

2

1 9

2

**Output:**

No reviews yet

9

9

**Explanation:**

Before the first query of the Chef, i.e. the first operation of type 2 in the input, the only ratings were 1 & 7. Thus, there will be total of  $2/3 = 0$  positive ratings. For the next two, the ratings list now looks

like: 1 5 7 8 9 21. Hence, top one-third will have  $6/3 = 2$  ratings as positive. And lowest displayed rating will be 9. Similarly for the last operation of type 2. Note that there are two ratings of the same value 9 now and only one of them can be in the top reviews. In such a case, you can choose any one of them

[Home](#) » [Compete](#) » [July Challenge 2012](#) » The Gray-Similar Code

## The Gray-Similar Code Problem Code: GRAYSC

---

The Gray code (see [wikipedia](#) for more details) is a well-known concept. One of its important properties is that every two adjacent numbers have exactly one different digit in their binary representation.

In this problem, we will give you  $n$  non-negative integers in a sequence  $A[1..n]$  ( $0 \leq A[i] < 2^{64}$ ), such that every two adjacent integers have exactly one different digit in their binary representation, similar to the Gray code.

Your task is to check whether there exist 4 numbers  $A[i1], A[i2], A[i3], A[i4]$  ( $1 \leq i1 < i2 < i3 < i4 \leq n$ ) out of the given  $n$  numbers such that  $A[i1] \text{ xor } A[i2] \text{ xor } A[i3] \text{ xor } A[i4] = 0$ . Here **xor** is a [bitwise operation](#) which is same as `^` in C, C++, Java and **xor** in Pascal.

---

### Input

First line contains one integer  $n$  ( $4 \leq n \leq 100000$ ). Second line contains  $n$  space separated non-negative integers denoting the sequence  $A$ .

---

### Output

Output "Yes" (quotes exclusive) if there exist four distinct indices  $i1, i2, i3, i4$  such that  $A[i1] \text{ xor } A[i2] \text{ xor } A[i3] \text{ xor } A[i4] = 0$ . Otherwise, output "No" (quotes exclusive) please.

---

### Example

Input:

5

1 0 2 3 7

**Output:**

Yes

[Home](#) » [Compete](#) » [July Challenge 2012](#) » Little Elephant and Bubble Sort

## Little Elephant and Bubble Sort Problem Code: LEBOBBLE

Little Elephant loves bubble sorting.

Bubble sorting for any array **A** of **n** integers works in the following way:

```
var int i, j;
for i from n downto 1
{
for j from 1 to i-1
{
if (A[j] > A[j+1])
swap(A[j], A[j+1])
}
}
```

You are given an array **B** of **n** integers. Then the array **A** is created using array **B** as following : for each **i** ( $1 \leq i \leq n$ ), we set  $A_i = B_i + d$  with the probability  $P_i$ , otherwise  $A_i = B_i$ .

Help Little Elephant to find the expect number of swaps that bubble sorting will make when the array **A** is sorted with the above bubble sorting algorithm.

---

### Input

First line of the input contains single integer **T** - the number of test cases. **T** test cases follows. First line of each test case contains pair of integers **n** and **d**. Next line of each test case contains **n** integers - array **B**. Next line contains **n** integers - array **P**.

---

### Output

In **T** lines print **T** real numbers - the answers for the corresponding test case. Please round all numbers to exactly 4 digits after decimal point.

---

### Constraint

$1 \leq T \leq 5$

$1 \leq n \leq 50000$

$1 \leq B_i, d \leq 10^9$

$0 \leq P_i \leq 100$

## Example

Input:

2

2 7

4 7

5050

4 7

5 6 1 7

25 74 47 99

Output:

0.2500

1.6049

[Home](#) » [Compete](#) » [July Challenge 2012](#) » Addition chains

## Addition chains Problem Code: ADDCHAIN

An *addition chain* for a positive integer  $N$  is a sequence of positive integers  $C = (A_0, A_1, \dots, A_L)$  such that

- $1 = A_0 < A_1 < \dots < A_{L-1} < A_L = N$ ,
- for each  $i$  such that  $1 \leq i \leq L$  there exist integers  $j$  and  $k$  such that  $0 \leq j, k < i$  and  $A_i = A_j + A_k$ .

The integer  $L$  is called the *length* of the chain  $C$ .

For example,  $C = (1, 2, 3, 6, 12, 15)$  is an addition chain of length 5 for  $N = 15$ . Indeed,

$$\begin{aligned}
 A_0 &= && && && & 1, \\
 A_1 &= 2 & = & 1 & + & 1 & = & A_0 + & A_0, \\
 A_2 &= 3 & = & 2 & + & 1 & = & A_1 + & A_0, \\
 A_3 &= 6 & = & 3 & + & 3 & = & A_2 + & A_2, \\
 A_4 &= 12 & = & 6 & + & 6 & = & A_3 + & A_3, \\
 A_5 &= 15 = 3 + 12 = A_2 + A_4.
 \end{aligned}$$

It is, in fact, the shortest addition chain for  $N = 15$ .

Your task is to find some addition chain for a given positive integer  $N$ .

**The shorter chain you will find the better score you will get.**

---

## Input

The only line of the input file contains an integer  $N$ , the number for which you need to find an addition chain.

---

## Output

The first line of the output file should contain an integer  $L \leq 500$ , the length of your addition chain for the value of  $N$  given in the input file.  $L$  lines should follow.  $i^{\text{th}}$  line should contain two non-negative integers  $j[i]$  and  $k[i]$  both less than  $i$  separated by exactly one space. Their meaning is that for your addition chain,  $A_i$  is equal to  $A_{j[i]} + A_{k[i]}$ . More specifically, your output will be considered as correct if and only if all of the following conditions hold:

- $1 \leq L \leq 500$ ;
- $0 \leq j[i], k[i] < i$  for all  $i$  such that  $1 \leq i \leq L$ ;
- for the sequence  $(A_0, A_1, \dots, A_L)$  defined by the rules  $A_0 = 1$  and  $A_i = A_{j[i]} + A_{k[i]}$  for  $i = 1, 2, \dots, L$  we have  $1 = A_0 < A_1 < \dots < A_{L-1} < A_L = N$ .

**Your program will get Accepted if and only if it returns correct output for each test case within a given time limit.**

---

## Scoring

Your score for a test file is the length of your addition chain, that is, an integer  $L$ . The total score for a submission is the average score across all test files. Your goal is to minimize the total score.

---

## Constraints

$1 < N < 10^{100}$

---

## Example

Input:

15

Output:

6

0 0

1 0

2 0

0 3

4 4

4 5

---

## Explanation

Sample output represents an addition chain **(1, 2, 3, 4, 5, 10, 15)** for **N = 15**. Your score for such output will be **6**. Note that according to the problem statement shorter chain is possible for this value of **N** but you don't need to find the shortest chain since this is a challenge problem. Also note that order in which we write numbers **j[i]** and **k[i]** for the given **i** doesn't matter.

---

## Test Generation

Values of **N** are generated using different strategies. There are **50** official test files in all. Among them there are **8** files where **N < 100000**, **5** files where **N** is a random decimal string of the length **100** (that is, each digit of **N** is distributed uniformly in the range **[0, 9]** except the first digit which is distributed in the range **[1, 9]**), **8** files where binary representation of **N** is a random string of the random length from **300** to **332** (that is, at first an integer **L** is chosen randomly in the range **[300, 332]** and then **L** digits **d<sub>1</sub>, d<sub>2</sub>, ..., d<sub>L</sub>** are chosen so that **d<sub>1</sub> = 1** and each **d<sub>i</sub>, 2 ≤ i ≤ L**, is uniformly distributed in the range **[0, 1]**. After that **N** is set to **d<sub>1</sub> · 2<sup>L-1</sup> + d<sub>2</sub> · 2<sup>L-2</sup> + ... + d<sub>L-1</sub> · 2 + d<sub>L</sub>**, that is, **d<sub>1</sub>d<sub>2</sub>...d<sub>L</sub>** is the binary representation of **N**). Other **29** files generated using some special strategies which we prefer to keep in secret.

# Favourite Numbers

Problem Code: FAVNUM

Chef likes numbers and number theory, we all know that. There are **N** digit strings that he particularly likes. He likes them so much that he defines some numbers to be **beautiful numbers** based on these digit strings.

Beautiful numbers are those numbers whose decimal representation contains **at least one** of chef's favorite digit strings as a substring. Your task is to calculate the **K<sup>th</sup>** smallest number amongst the beautiful numbers in the range from **L** to **R** (both inclusive). If the number of beautiful numbers between **L** and **R** is less than **K**, then output "**no such number**".

## Input

In the first line of input there will be integers **L**, **R**, **K** and **N**. Then **N** lines follow. Each line will contain a single string of decimal digits.

## Output

Output one integer - the solution to the problem described above or a string "**no such number**" if there is no such number.

## Constraints

- $1 \leq L \leq R \leq 10^{18}$
- $1 \leq K \leq R - L + 1$
- $1 \leq N \leq 62$
- $1 \leq \text{The length of any Chef's favourite digit string} \leq 18$ . Each string begins with a nonzero digit.

## Example

Input:

1 1000000000 4 2

62

63

**Output:**

163

**Input:**

1 1 1 1

2

**Output:**

no such number

**Input:**

1 1000 15 2

6

22

**Output:**

67

[Home](#) » [Compete](#) » [July Challenge 2012](#) » Dynamic GCD

## Dynamic GCD Problem Code: DGCD

You're given a [tree](#) on **N** vertices. Each vertex has a positive integer written on it, number on the  $i^{th}$  vertex being  $v_i$ . Your program must process two types of queries :

1. Find query represented by **F u v** : Find out [gcd](#) of all numbers on the unique path between vertices **u** and **v** in the tree (both inclusive).

2. Change query represented by **C u v d** : Add **d** to the number written on all vertices along the unique path between vertices **u** and **v** in the tree (both inclusive).
- 

## Input

First line of input contains an integer **N** denoting the size of the vertex set of the tree. Then follow **N - 1** lines, *i*<sup>th</sup> of which contains two integers **a<sub>i</sub>** and **b<sub>i</sub>** denoting an edge between vertices **a<sub>i</sub>** and **b<sub>i</sub>** in the tree. After this follow **N** space separated integers in a single line denoting initial values **v<sub>i</sub>** at each of these nodes. Then follows a single integer **Q** on a line by itself, denoting the number of queries to follow. Then follow **Q** queries, each one on a line by itself. Each query is either a find query or a change query with format as given in problem statement. Note that all vertices are 0-based.

---

## Output

For every find query, print the answer to that query in one line by itself.

---

## Example

Input:

```
6
0 4
0 5
1 5
5 2
3 5
3 1 3 2 4 2
5
F 3 5
C 1 3 1
C 3 4 4
F 3 0
```

F 2 5

**Output:**

2

7

1

## Constraints

 $1 \leq N \leq 50000$  $1 \leq Q \leq 50000$  $0 \leq u, v \leq N-1$  $1 \leq v_i \leq 10^4$  $0 \leq d \leq 10^4$ 

[Home](#) » [Compete](#) » [July Challenge 2012](#) » Equivalent Suffix Tries

## Equivalent Suffix Tries Problem Code: EST

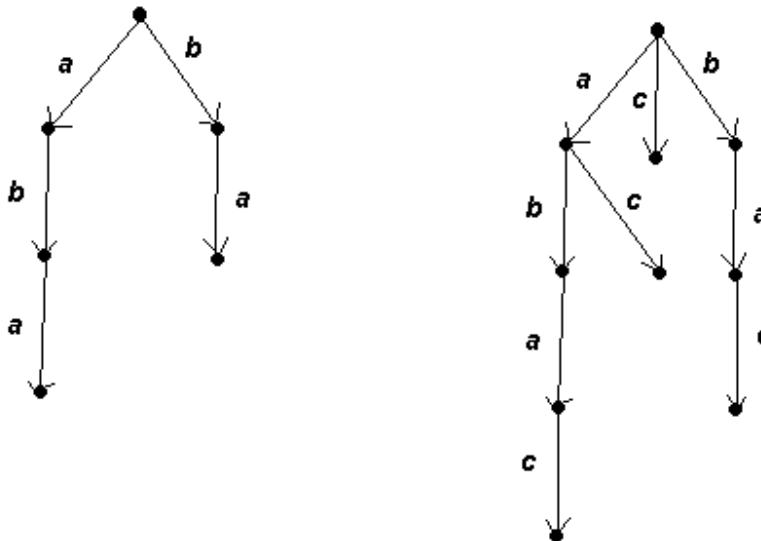
Suffix trees are very powerful data structures. Using suffix trees it's often easy to solve the hardest computer science problems on strings. In this problem we'll consider a simplified version of suffix trees -- suffix tries (that is, tries formed by all suffixes of a given string).

Generally, the suffix tree is a tree whose edges are labeled with strings. Instead, we'll only consider suffix tries whose edges are labeled with single characters. This way every edge of a suffix tree may correspond to one or more edges connected in a chain-style fashion of a suffix trie.

Another trait of suffix trees is that each suffix of the string for which the suffix tree is built corresponds to exactly one path from the tree's root to a leaf. This is usually achieved by terminating each suffix of the string with a special character, say, \$. However, in suffix tries

we won't do that. In particular, this implies that the number of leaves in a suffix trie can be smaller than the length of the original string.

In the picture the suffix tries for strings 'aba' and 'abac' are presented:



If we erase all letters from the edges of a suffix trie, it actually becomes a directed graph. Let's call two suffix tries equivalent if their corresponding directed graphs are isomorphic.

You're given a single string consisting of lowercase Latin letters. Your task is to find the number of different strings of the same length consisting of lowercase Latin letters as well such that their suffix tries are equivalent to the suffix trie of the given string. As this number can be large enough, output the remainder of its division by 42424242.

## Input

The first line contains a single number **T** -- the number of test cases (no more than 10). Each of the next **T** lines contains a single non-empty string of length no more than 100000 consisting of lowercase Latin letters a..z.

## Output

For each test case output just one line containing the sought number modulo 42424242.

## Example

**Input:**

5

a

aa

abc

aba

helloworld

**Output:**

26

26

15600

1300

6221124

**Explanation:**

In the first test case every string of length 1 satisfies the condition. In the second test case every string consisting of two equal letters is fine. In the third test case every string comprising three pairwise distinct letters adds 1 to the answer. In the fourth test case string 'bee' is one of the 1300 good strings.

COOK24

July Cook-Off  
201222 Jul 2012  
21:30:00

2 hours 30 minutes

993

[Home](#) » [Compete](#) » [July Cook-Off 2012](#) » Ciel and Receipt

## Ciel and Receipt Problem Code: CIELRCPT

Tomya is a girl. She loves Chef Ciel very much.

Tomya like a positive integer  $p$ , and now she wants to get a receipt of Ciel's restaurant whose total price is exactly  $p$ . The current menus of Ciel's restaurant are shown the following table.

Name of Menu	price
eel flavored water	1
deep-fried eel bones	2
clear soup made with eel livers	4
grilled eel livers served with grated radish	8
savory egg custard with eel	16
eel fried rice (S)	32
eel fried rice (L)	64
grilled eel wrapped in cooked egg	128
eel curry rice	256
grilled eel over rice	512
deluxe grilled eel over rice	1024
eel full-course	2048

Note that the  $i$ -th menu has the price  $2^{i-1}$  ( $1 \leq i \leq 12$ ).

Since Tomya is a pretty girl, she cannot eat a lot. So please find the minimum number of menus whose total price is exactly  $p$ . Note that if she orders the same menu twice, then it is considered as two menus are ordered. (See **Explanations** for details)

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case contains an integer  $p$ .

## Output

For each test case, print the minimum number of menus whose total price is exactly  $p$ .

## Constraints

$1 \leq T \leq 5$

$1 \leq p \leq 100000 (10^5)$

There exists combinations of menus whose total price is exactly  $p$ .

---

## Sample Input

4

10

256

255

4096

---

## Sample Output

2

1

8

2

---

## Explanations

In the first sample, examples of the menus whose total price is 10 are the following:

$1+1+1+1+1+1+1+1+1 = 10$  (10 menus)

$1+1+1+1+1+1+1+2 = 10$  (9 menus)

$2+2+2+2 = 10$  (5 menus)

$2+4+4 = 10$  (3 menus)

$2+8 = 10$  (2 menus)

Here the minimum number of menus is 2.

In the last sample, the optimal way is  $2048+2048=4096$  (2 menus). Note that there is no menu whose price is 4096.

[Home](#) » [Compete](#) » [July Cook-Off 2012](#) » Ciel and Tomya

## Ciel and Tomya

Problem Code: CIELTOMY

---

Tomya is a girl. She loves Chef Ciel very much.

Today, too, Tomya is going to Ciel's restaurant. Of course, Tomya would like to go to Ciel's restaurant as soon as possible. Therefore Tomya uses one of the shortest paths from Tomya's house to Ciel's restaurant. On the other hand, Tomya is bored now to use the same path many times. So Tomya wants to know the number of shortest paths from Tomya's house to Ciel's restaurant. Your task is to calculate the number under the following assumptions.

This town has **N** intersections and **M** two way roads. The **i**-th road connects from the **A<sub>i</sub>**-th intersection to the **B<sub>i</sub>**-th intersection, and its length is **C<sub>i</sub>**. Tomya's house is in the 1st intersection, and Ciel's restaurant is in the **N**-th intersection.

## Input

The first line contains an integer **T**, the number of test cases. Then **T** test cases follow. The first line of each test case contains 2 integers **N**, **M**. Then next **M** lines contain 3 integers denoting **A<sub>i</sub>**, **B<sub>i</sub>** and **C<sub>i</sub>**.

## Output

For each test case, print the number of shortest paths from Tomya's house to Ciel's restaurant.

## Constraints

$1 \leq T \leq 10$

$2 \leq N \leq 10$

$1 \leq M \leq N \cdot (N - 1) / 2$

$1 \leq A_i, B_i \leq N$

$1 \leq C_i \leq 10$

$A_i \neq B_i$

If  $i \neq j$  and  $A_i = A_j$ , then  $B_i \neq B_j$

There is at least one path from Tomya's house to Ciel's restaurant.

## Sample Input

2

3 3

1 2 3

2 3 6

1 3 7

3 3

1 2 3

2 3 6

1 3 9

## Sample Output

1

2

## Explanations

In the first sample, only one shortest path exists, which is 1-3.

In the second sample, both paths 1-2-3 and 1-3 are the shortest paths.

[Home](#) » [Compete](#) » [July Cook-Off 2012](#) » Ciel and Map

## Ciel and Map Problem Code: CIELMAP

Tomya is a girl. She loves Chef Ciel very much.

Tomya writes a tetragon (see the below **Notes 1** for details) in her map. The map has **N** points, whose coordinate are  $(X_i, Y_i)$ , denoting Ciel's restaurants. Each vertex of the tetragon written by Tomya is one of the Ciel's restaurants.

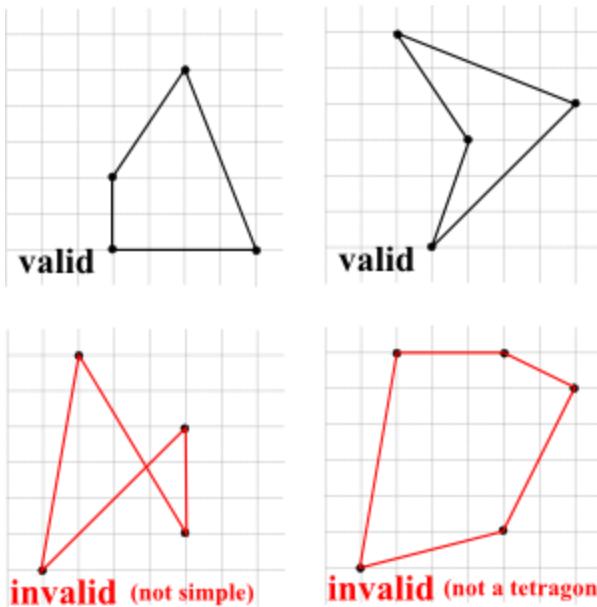
Tomya doesn't show her tetragon in the map, but she says that she draws a long segment. To guess her tetragon, please tell the maximum length of a segment which can be one of the sides in her tetragon. (See **Notes 2** for the definition of length)

### Notes 1:

A tetragon means a polygon with 4 sides.

A tetragon must be simple, that is, the boundary of the tetragon does not cross itself.

A tetragon does not have to be convex.



### Notes 2:

The length of a segment is defined as the Euclidean distance between the endpoints. That is, the length of the segment  $(x_1, y_1) - (x_2, y_2)$  is the square root of  $(x_1 - x_2)^2 + (y_1 - y_2)^2$ .

### Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line of each test case contains an integer  $N$ . Then next  $N$  lines follow. The  $i$ -th line contains 2 integers denoting  $X_i$  and  $Y_i$ .

### Output

For each test case, print the maximum length of a segment which can be one of sides in her tetragon. This value must have an absolute error no more than  $0.000001 (10^{-6})$ .

### Constraints

$$1 \leq T \leq 1250$$

$$4 \leq N \leq 1000$$

$$1 \leq X_i, Y_i \leq 5000$$

The sum of  $N$  in one test file does not exceed 5000.

No three points  $(X_i, Y_i)$ ,  $(X_j, Y_j)$ ,  $(X_k, Y_k)$  lie in the same line ( $1 \leq i < j < k \leq N$ )

### Sample Input

6  
3 1  
5 2  
3 3  
6 3  
6 6  
5 6

---

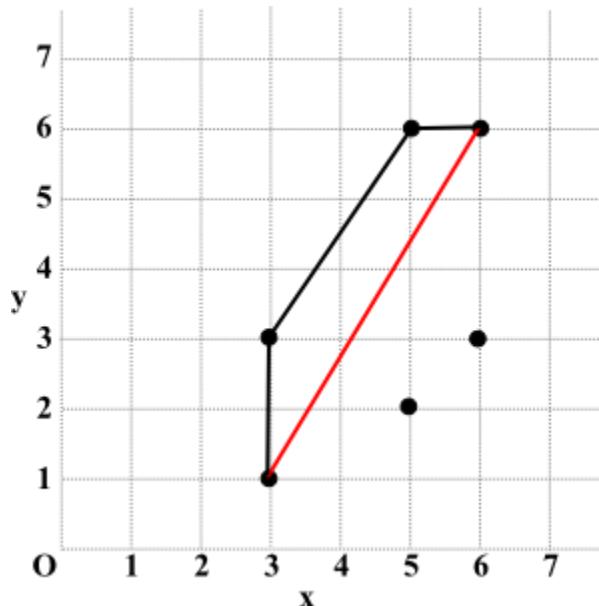
## Sample Output

5.8309518948

---

## Explanations

The below figure denotes one of the valid tetragons for the sample input. The red segment is the longest segment which can be one of the sides in her tetragon.



[Home](#) » [Compete](#) » [July Cook-Off 2012](#) » Ciel Numbers III

## Ciel Numbers III

Problem Code: CIELNUM3

---

Tomya is a girl. She loves Chef Ciel very much.

Tomya also loves *Ciel numbers*. The definition of Ciel numbers are following.

Ciel numbers are defined as the positive integers  $k$  such that  $d(k, 8) \geq d(k, 5) \geq d(k, 3)$  and  $d(k, i) = 0$  for all  $i = 0, 1, 2, 4, 6, 7, 9$ , where  $d(k, i)$  denotes the number of the digit  $i$  in the decimal representation of the integer  $k$ . For example, the first few Ciel numbers are 8, 58, 85, 88, 358, 385, 538, 583, 588, 835, 853, 858, 885, 888, ....

Now Tomya defines the  $x$ -pseudo Ciel numbers as follows for non-negative integers  $x$ :

- A positive integer  $k$  is a 0-pseudo Ciel numbers if and only if  $k$  is a Ciel number.
- For  $x \geq 1$ , a positive integer  $k$  is a  $x$ -pseudo Ciel numbers if and only if  $k$  is not a  $y$ -pseudo Ciel number for all  $0 \leq y < x$  and, there exists a  $(x-1)$ -pseudo Ciel number  $s$  such that  $s$  have the same number of digits as  $k$  and exactly one digit differ from  $k$  in their decimal notations.

For example,

583, 58588, 355388, and 5358388 are 0-pseudo Ciel numbers, and 523, 58558, 355308, and 5354388 are 1-pseudo Ciel numbers, and 123, 55558, 353308, and 5314388 are 2-pseudo Ciel numbers, and 124, 55555, 333308, and 1314388 are 3-pseudo Ciel numbers.

In this problem, your task is to find the largest  $x$ -pseudo Ciel number less than or equal to  $N$ .

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. Each test case contains 2 integers  $x$  and  $N$  in one line.

## Output

For each test case, print the largest  $x$ -pseudo Ciel number less than or equal to  $N$ . If there are no such numbers, print "-1" (excluding quotes).

## Constraints

$$1 \leq T \leq 10000 (10^4)$$

$$0 \leq x \leq 1000000000 (10^9)$$

$$1 \leq N < 10^{100000}$$

The sum of the length of  $N$  in one judge input file does not exceed 1000000 ( $10^6$ ).

## Sample Input

0 58600

2 128

1 5354389

1 1

2 1

---

## Sample Output

58588

128

5354388

1

-1

[Home](#) » [Compete](#) » [July Cook-Off 2012](#) » Ciel and Ball Guessing Game

# Ciel and Ball Guessing Game Problem Code: CIELBALL

Tomya is a girl. She loves Chef Ciel very much.

Since Tomya has completed collecting the stamps in Ciel's restaurant, she now challenges the bonus game in her restaurant. There are colored balls in a box. The number of colors is **N**, and the number of balls color **i** in the box is **S<sub>i</sub>** for **i** = 1, 2, ..., **N** initially. At first, Tomya has **M** coins. And Tomya's aim in this game is to get many coins at the end of the game. This game will run as follows.

Tomya chooses a color, and Tomya bets some coins (suppose **x** coins are bet) on the color. Here **x** must be non-negative integer at most **L**, and Tomya must have **x** coins before betting. Then Ciel chooses a ball from the box, checks its color and discards the ball. If the ball has the same color as Tomya's bet, then Tomya gets **B**·**x** coins, otherwise Tomya must pay **x** coins. Note that, if Tomya wins, then she receives back her bet as well as **B**·**x** coins. While the box has at least one ball, the above process are repeated. When the box is empty, the game is over.

The game is very kind, because Tomya always can increase her coins. Instead of this, Ciel is not kind in this game, she chooses the balls which minimize Tomya's coins at the end of the game (Yes, she can choose balls as she likes!). Your task is to calculate the maximum number of coins that can be obtained by Tomya at the end of the game.

## Input

The first line contains an integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line of each test case contains 4 integers  $N, B, M, L$ . Then the next line contains  $N$  integers  $S_1, S_2, \dots, S_N$ .

---

## Output

For each test case, print the maximum number of coins that can be obtained by Tomya at the end of the game.

---

## Constraints

$1 \leq T \leq 5$   
 $1 \leq N, B \leq 30$   
 $1 \leq M, L \leq 1000000000000 (10^{12})$   
 $1 \leq S_i \leq 30$

---

## Sample Input

```
2
2 1 30 100
2 2
3 1 1 58
30 28 2
```

---

## Sample Output

```
80
2
```

---

## Explanations

In the first sample, one of the strategy to obtain 80 coins is as follow:  
At first, Tomya bets 0 coins. Let the color 2 be chosen by Ciel. (Now,  $S_1 = 2, S_2 = 1, \text{Coins} = 30$ )  
Then, Tomya bets 10 coins on the color 1.

--- If the color 1 is chosen, she will get 10 coins.  
 --- (Now,  $S_1 = 1$ ,  $S_2 = 1$ , Coins = 40)  
 ----- In next turn, Tomya bets 0 coins, and let color 2 be chosen.  
 ----- (Now,  $S_1 = 1$ ,  $S_2 = 0$ , Coins = 40)  
 ----- Lastly, Tomya bets all 40 coins on the color 1, and she will get 40 coins.  
 ----- (Game over with  $S_1 = 0$ ,  $S_2 = 0$ , Coins = 80)  
 --- If the color 2 is chosen in the second turn, she must lose 10 coins.  
 --- (Now,  $S_1 = 2$ ,  $S_2 = 0$ , Coins = 20)  
 ----- In this case, Tomya bets all 20 coins on the color 1, and she will get 20 coins.  
 ----- (Now,  $S_1 = 1$ ,  $S_2 = 0$ , Coins = 40)  
 ----- Then, once more, Tomya bets all 40 coins on the color 1, and she will get 40 coins.  
 ----- (Game over with  $S_1 = 0$ ,  $S_2 = 0$ , Coins = 80)

In the second sample, Tomya has only 1 coin. Therefore Tomya must not bet her coin while the box contains balls of at least 2 different colors. Ciel must always choose a ball of the most popular color. So while the box has at least two balls there always exist two balls of different colors in the box. So Tomya can bet her coin only on the last ball.

AUG12

[August Challenge 2012](#)01 Aug 2012  
15:00:00

10 days

2847

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Little Elephant and Bombs

## Little Elephant and Bombs Problem Code: LEBOMBS

The Little Elephant from the Zoo of Lviv currently is on the military mission. There are  $N$  enemy buildings placed in a row and numbered from left to right starting from  $0$ . Each building  $i$  (except the first and the last) has exactly two adjacent buildings with indices  $i-1$  and  $i+1$ . The first and the last buildings have just a single adjacent building.

Some of the buildings contain bombs. When bomb explodes in some building it destroys it and all adjacent to it buildings.

You are given the string  $S$  of length  $N$ , where  $S_i$  is  $1$  if the  $i$ -th building contains bomb,  $0$  otherwise. Find for the Little Elephant the number of buildings that will not be destroyed after all bombs explode. Please note that all bombs explode simultaneously.

---

### Input

The first line contains single integer  $T$  - the number of test cases.  $T$  test cases follow. The first line of each test case contains the single integer  $N$  - the number of buildings. The next line contains the string  $S$  of length  $N$  consisted only of digits  $0$  and  $1$ .

---

### Output

In  $T$  lines print  $T$  integers - the answers for the corresponding test cases.

---

## Constraints

$1 \leq T \leq 100$

$1 \leq N \leq 1000$

---

## Example

**Input:**

3

3

010

5

10001

7

0000000

**Output:**

0

1

7

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Delivery Boy

## Delivery Boy Problem Code: HOMDEL

---

Chef has started Home Delivery scheme in one of his restaurants. As the scheme is new , Chef appoints only one employee to deliver food to various locations. The delivery boy who has been appointed is an absent-minded chap. He always forgets to fill fuel in his delivery scooter. So what he does is that whenever Chef sends him for delivery, he goes to the gas station from the restaurant first. He gets his tank filled and then he heads towards his destination. He will do this every single time *irrespective of the destination*. The delivery boy tries his best to be on time. And to do this, he will choose those paths(from restaurant to

gas station AND gas station to destination) which cost him the *least* amount of time. Your task is to tell the Chef how much time can the delivery boy save if he had enough fuel in his scooter i.e. if he went to the destination directly without stopping for fuel (taking the path which costs him least amount of time).

The city has **N** streets numbered from **0** to **N-1**. The restaurant is on street number **S**, the gas station is on street number **G** and the food has to be delivered to street **D**. Note that **S**, **G** and **D** need **not** be distinct.

---

### Input:

First line of the input contains a single integer **N**.

Then follows an **NxN** matrix **T** which is represented in **N** lines with **N** space separated integers on each line.

**T[i][j]** denotes the time taken to move from the  $i^{\text{th}}$  street to  $j^{\text{th}}$  street. Obviously,  $T[i][i] = 0$ .

Next line contains a single integer **M**, the number of scenarios.

The following **M** lines contain 3 space separated integers **S**, **G** and **D**.

---

### Output:

For each of the **M** scenarios, output the time taken by the delivery boy to deliver the food and the time he could have saved if he went directly from **S** to **D**.

Both these values must be on the same line separated by a single space.

---

### Constraints:

$$1 \leq N \leq 250$$

$$1 \leq M \leq 10000$$

$$0 \leq T[i][j] \leq 100000$$

$$0 \leq S, G, D \leq N-1$$


---

### Example:

#### Input:

4

0 2 1 3

1 0 4 5

3 1 0 3

1 1 1 0

4

0 2 1

0 2 2

3 1 2

3 0 1

**Output:**

2 0

1 0

3 2

3 2

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Range of Data

## Range of Data Problem Code: DRANGE

Alice has **N** pieces of paper. These papers are numbered from **1** to **N**. She writes down the numbers **1** to **N** in order (one number on each paper), i.e. paper *i* has number *i* written on it. Bob messes the numbers on these papers. He either **adds** a constant to a number or **subtracts** a constant from the number. He performs **M** such operations. Each operation is of the form: **w x y z** where each of them is an integer. If **w = 1**, then Alice has to **add z** to every number on papers **x** to **y** (*both inclusive*). If **w = 2**, then Alice has to **subtract z** from every number on papers **x** to **y** (*both inclusive*). After doing this, Bob challenges Alice to tell him the range of this data, where *range* denotes the count of numbers from the smallest number to the largest (See [here](#) for more details). Your task is to help Alice in finding the range.

---

**Input:**

First line of input contains a single integer **T**, the number of test cases.  
 Each test case starts with a line containing two space separated integers **N** and **M**.  
 Then follow **M** lines. Each of these lines is of the form **w x y z**. Each separated by a single space.

---

### Output:

For each test case output a single line containing the range of the new data set after Bob's modifications.

---

### Constraints:

$1 \leq T \leq 20$

$1 \leq M \leq 10000$

$1 \leq N \leq 1000000$

$1 \leq x \leq y \leq N$

$0 \leq z \leq 100000$

---

### Example:

#### Input

1

10 2

2 3 6 4

1 5 9 1

#### Output:

11

**Explanation:** Initially the papers are as follows: 1 2 3 4 5 6 7 8 9 10. First operation decreases the numbers on paper number 3,4,5 and 6 by 4. Now, the papers look like: 1 2 -1 0 1 2 7 8 9 10. The second

operation increases the numbers on papers 5 to 9 by 1. The numbers will now be 1 2 -1 0 2 3 8 9 10 10. Thus, the range is  $10 - (-1) = 11$ .

[Home](#) » [Compete](#) » [August Challenge 2012](#) » [Lucky Driving](#)

## Lucky Driving Problem Code: LUKYDRIV

"Nine is considered a **good** number in Chinese culture because it sounds the same as the word "long-lasting". Hence many people want their car registration numbers to sum up to **9** on adding the digits recursively.

A car registration number can consist of 1,2,3 or 4 digits. Examples of some **good** car numbers are 63, 018, 9099, etc. whereas 12,129 are not **good**.

Why is **9099** **good**?

$$9+0+9+9 = 27$$

$$2+7 = 9$$

Similarly it is not difficult to see why 63, 018, 6669, 9999... are **good**."

Given a string of digits, print how many subsequences of the string result in **good** car registration numbers.

### Note:

The zeros at the starting of the number are to be accounted for, i.e. 018 and 18 are different numbers.

### Input

First line of the input contains **T** (**T**  $\leq 200$ ) denoting the number of test cases. **T** lines follow each containing a non-empty single string **S**. **S** contains only digits i.e. from **[0-9]**. Length of **S**  $\leq 10000$ .

### Output

For each case print how many subsequences of the string result in **good** car registration numbers. As the answer can be quite large print it modulo **1000000007**.

### Example

### Input

10292

0189

## Output

2

6

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Block Game

# Block Game Problem Code: BLOCKING

In the magic land, there is a smart but naughty boy Crane. Crane has  $n$  houses, and  $n$  friends (both numbered from 1 to  $n$ ). Crane's mother has scheduled a plan for all his friends such that each friend visits each house of Crane exactly once at a different time, i.e., there is at most one friend in any house at any time.

But like Crane all his friends are also naughty. Therefore his mother wants to lock away each of his friends in distinct houses. What she'll do is to choose a distinct house for each friend. As that friend visits that house, he'll be locked in that house. That is, Crane's mother wants to find a sequence block such that the  $i$ -th friend will be locked in  $\text{block}[i]$ -th house once he visits that house. As she wants to ensure that there is at maximum one person in each house, she must choose the sequence block such that if friend A visits house H at time T, and he is locked away in the house, then no other friends visit house H after time T.

Help Crane's mother to find such a sequence block.

---

## Input

First line contains a single integer  $n$  ( $1 \leq n \leq 100$ ), indicating the number of friends and houses. Then  $n$  lines follow, with each line containing  $n$  positive integers. The  $j$ -th integer in  $i$ -th line indicates the time when the  $i$ -th friend will visit the  $j$ -th house. All times will fit in signed 32-bit integer.

---

## Output

If there exists a sequence block that satisfies the constraints, then output  $\text{block}[1] \text{ } \text{block}[2] \dots \text{ } \text{block}[n]$  in one line, separated by a single space. Otherwise if there is no such sequence, output -1 in one line. If there are multiple answers, then anyone will do.

---

## Example

**Input:**

3

1 2 3

4 5 6

7 8 9

**Output:**

3 2 1

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Machine Gun

## Machine Gun Problem Code: MACGUN

Country X has a military zone framed as a grid of size  $M \times N$ . Each point of the grid could be free, occupied by a machine gun or by a protector.

A machine gun at point  $(x, y)$  could automatically attack other guns at points  $(x-2, y-2)$ ,  $(x-2, y+2)$ ,  $(x+2, y-2)$  and  $(x+2, y+2)$ . So we could not put two guns at points which make them destroy each others, unless there is a protector exactly in the middle. Eg.: We could put two guns at point  $(x, y)$  and  $(x+2, y+2)$  if and only if there is a protector at point  $(x+1, y+1)$ .

Given a map with some points which has been occupied by guns or protectors, your task is to find out the greatest number of guns to add to this map satisfying rules above. Note that we can put a machine gun at a free point, but we cannot remove any machine gun and protector, and we cannot add protectors.

---

### Input

There are several test cases, each formed as follows:

- The first line contains two positive integer  $M, N$ .
- Next  $M$  lines, each contains  $N$  characters (no spaces) of  $\{'F', 'G', 'P'\}$  (ASCII: #70, #71, #80), the  $j$ -th character of the  $i$ -th line represents the point  $(i-1, j-1)$  on the map: 'F' is a free point, 'G' is occupied by a machine gun and 'P' is occupied by a protector.

The input is ended with  $M = N = 0$ .

---

### Output

For each test case, output on a line the greatest number of guns which can be added the given map.

---

## Constraints

$1 \leq M, N \leq 700$

In the given map, any two machine guns do not attack each other.

The sum of  $M \times N$  does not exceed 490000 in one judge file.

---

## Example

**Input:**

3 4

FPFP

PFPP

GFGF

5 3

FPF

FFF

FGG

PFP

FPF

0 0

**Output:**

3

6

## Game Count Problem Code: MAXGAME

---

Tug of war is a sport that directly puts two teams against each other in a test of strength. During school days, both Chef Shifu and Chef Po were champions of tug of war. On behalf of restaurant's anniversary, Chef Shifu and Chef Po have decided to conduct a tug of war game for their customers. Master Chef Oogway has decided the following rules for the game.

1. Let **N** be the number of players participating in the game. All of these players would stand in a circle in clock wise direction.
2. There are an infinite number of long ropes available. When a rope is held by exactly two players, it is termed as bonding.
3. At least one bonding is necessary to conduct a game.
4. A player can play against multiple people simultaneously i.e he can have more than one bonding at the same time.
5. Both members of a pair of players that have a bonding must have the same number of total bondings. That is, if the player *A* makes bonding with the player *B*, then the number of total bondings of the player *A* must be the same as that of the player *B*.
6. Bondings should be created in such a fashion that ropes must not intersect each other.
7. The number of bondings of every player must be no more than **K**.

Now Master Chef Oogway asked Chef Shifu and Chef Po to find out the number of possible games. Your task is to help them find this number. As this number might become huge, you've to find it modulo  $(10^{14}+7)$ . Two games are different iff there is some bonding that is present in only of them.

---

### Input

First line contains **T**, the number of test cases. Each of **T** lines contain 2 positive integers **N** and **K** separated by a space.

---

### Output

For each test case, output the number of ways to conduct the game modulo  $100000000000000007$  ( $10^{14}+7$ ) in one line.

---

### Example

**Input:**

3

3 2

4 0

2 1

**Output:**

4

0

1

**Explanation:**

For the 1st case, there are 3 players. Let's call them p1, p2, p3. Different games possible are:

Game 1: p1-p2 (numbers of bondings of p1, p2 are  $1 \leq K = 2$ )

Game 2: p1-p3 (numbers of bondings of p1, p3 are  $1 \leq K = 2$ )

Game 3: p2-p3 (numbers of bondings of p2, p3 are  $1 \leq K = 2$ )

Game 4: p1-p2, p1-p3, p2-p3 (numbers of bondings of p1, p2, p3 are  $2 \leq K = 2$ )

For the 2nd test case, we cannot form the game, because  $K = 0$  and hence no player is allowed to make any bonding. As any game must have atleast one bonding, no game is possible here.

For the 3rd case, only possible game is:

Game 1: p1-p2 (number of bondings in p1, p2 are 1)

**Constraints**

$1 \leq T \leq 10000$

$0 \leq N \leq 10000$

$0 \leq K \leq N$

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Eastern Draughts

## Eastern Draughts Problem Code: CHECKERS

Eastern Draughts is played on an  $N \times N$  grid, consisting of cells  $(x, y)$  with  $0 \leq x, y < N$ . Some cells of the grid have pieces on them, and no cell may ever have more than one piece at a time. The number of pieces will be equal to  $P \cdot (P+1)/2$  for some positive integer  $P \leq N$ .

There are two types of moves in the game:

- A step consists of moving a piece from  $(x, y)$  to  $(x+1, y)$ ,  $(x-1, y)$ ,  $(x, y+1)$ ,  $(x, y-1)$ ,  $(x+1, y-1)$ , or  $(x-1, y+1)$ . The destination cell must be empty, and must lie within the bounds of the grid. Note that you cannot move a piece directly from  $(x, y)$  to  $(x-1, y-1)$  or  $(x+1, y+1)$ .
- A jump consists of moving a piece from  $(x, y)$  to  $(x+2, y)$ ,  $(x-2, y)$ ,  $(x, y+2)$ ,  $(x, y-2)$ ,  $(x+2, y-2)$ , or  $(x-2, y+2)$ . Not only must the destination cell be empty and lie within the bounds of the grid, but the cell between the source and destination cells must contain a piece.

A turn consists of either a single step, or a sequence of one or more jumps where the same piece is moved with each jump. The goal is to move the pieces to the goal configuration in as few turns as possible. The goal configuration is such that each cell  $(x, y)$  contains a piece if and only if  $x+y < P$ .

For this problem, you do not need to use as few turns as possible, but the fewer turns you use the higher your score will be.

---

## Input

Input will begin with an integer  $N$ , the size of the grid.  $N$  lines follow with  $N$  characters each, giving the initial positions of the pieces. A '\*' indicates a piece, and a '.' indicates an empty cell. The upper-left corner is  $(0, 0)$ , upper-right corner is  $(N-1, 0)$ , bottom-left corner is  $(0, N-1)$ , and bottom-right corner  $(N-1, N-1)$ .

---

## Output

First output an integer  $L$ , the number of moves in your solution (not number of turns). Then output  $L$  lines, each formatted as " $x_1 y_1 x_2 y_2$ ", where  $(x_1, y_1)$  is the source cell and  $(x_2, y_2)$  is the destination cell.  $L$  must be less than one million.

---

## Scoring

Let  $K$  be the number of turns in your solution, and let  $S$  be the sum of  $x+y$  over all cells initially containing pieces, and  $R$  be the sum of  $x+y$  over all cells containing pieces in the goal configuration. Then your score for each test case is  $(S - R)/K$ . Your overall score is the average of your scores on the individual input files.

---

## Sample Input 1

4

....

....

....

..\*.

---

### Sample Output

5

2 3 2 2

2 2 2 1

2 1 2 0

2 0 1 0

1 0 0 0

---

### Sample Input 2

5

....

..\*.

..\*.

....

..\*..

---

### Sample Output 2

10

2 4 2 3

2 3 4 1

4 1 2 1

3 2 3 0

3 1 1 1

3 0 1 2

1 2 1 0

2 1 0 1

0 1 0 0

1 1 0 1

The sample output uses 5 turns on the first test case, and 8 on the second. The sequence of turns on the second test case is:

1. (2 4) - (2 3)
2. (2 3) - (4 1) - (2 1)
3. (3 2) - (3 0)
4. (3 1) - (1 1)
5. (3 0) - (1 2) - (1 0)
6. (2 1) - (0 1)
7. (0 1) - (0 0)
8. (1 1) - (0 1)

For the first test case  $S = 5$  and  $R = 0$ , and for the second case  $S = 15$  and  $R = 2$ . Thus the score for the first test case is  $(5 - 0)/5 = 1$ , and the score for the second test case is  $(15 - 2)/8 = 1.625$ . The overall score is  $(1 + 1.625)/2 = 1.3125$ .

## Test Case Generation

$N$  is chosen randomly and uniformly between 10 and 30, inclusive.  $P$  is chosen randomly and uniformly between 1 and  $N$ , inclusive.  $P*(P+1)/2$  cells are chosen at random for pieces. You may safely assume there will be no test cases where the starting configuration is equal to the goal configuration.

[Home](#) » [Compete](#) » [August Challenge 2012](#) » Two Magicians

## Two Magicians Problem Code: MAGIC

Two magicians are playing a funny game in a system of  $N$  rooms and  $M$  two-way passages between them. The rooms are small and the passages are straight, so no pair of rooms can ever be connected by more than one passage. Moreover, no two passages can ever intersect. The rooms are situated in 3D pretty well, so it's still possible for all pairs of rooms to be connected by passages at once.

Initially the first magician is located at room 1 and the second magician is located at (a candy for guessing!) room 2. Each magician has **P** *telepoints* before the start. The game consists of *turns*, with the first magician taking the first turn and then both magicians taking turns alternately. Each turn consists of three *phases*. We'll call the magician taking the turn the *current* magician.

In the first phase, the current magician can walk between rooms through the existing passages as much as he wants (he can choose not to move at all as well). Residing at one of the rooms, he announces that the phase is finished. If after this phase both magicians are found residing at the same room, the game ends and the current magician wins. Otherwise, the turn is continued.

In the second phase, the current magician must create a passage between any two rooms of the system which are not directly connected by a passage yet. If there is no such pair of rooms, the game ends and the current magician loses. Otherwise, the turn is continued.

In the third phase, the current magician can either stay at the same room or, if he has a strictly positive number of telepoints remaining, he can teleport into any of the rooms. In case of teleportation the current magician loses 1 telepoint. If after this phase the current magician turns out to be residing at the same room as the other one, nothing happens (though the current magician is likely to lose pretty soon).

You are given the information about the system of rooms and passages before the game. You'll get a point in the contest rankings if you find out which of the magicians is going to win considering them both playing optimally, that is, winning whenever there exists a guaranteed winning strategy.

## Input

The first line contains a single integer **T** -- the number of test cases (no more than 100). Each test case is described by an empty line followed by a line containing three integers **N**, **M** and **P** ( $2 \leq N \leq 7777$ ,  $0 \leq M \leq 10000$ ,  $0 \leq P \leq 10000$ ) followed by **M** lines containing two integers  $X_i$  and  $Y_i$  each ( $1 \leq X_i, Y_i \leq N$ ,  $X_i \neq Y_i$ ). Each of these **M** lines describes a single two-way passage between rooms  $X_i$  and  $Y_i$ . It's guaranteed that no two rooms are directly connected by more than one passage.

## Output

For each test case output a single line with a single word *First* if the first magician has a winning strategy in this game, and *Second* otherwise.

## Example

Input:

4

2 0 0

3 1 0

1 3

4 0 1

4 1 1

3 4

**Output:**

Second

Second

Second

First

**Explanation:**

In the first two test cases, the game ends after the second turn (which is the first turn of the second magician) as the passage created by the first magician hands a win to the second magician. In the third test case, the game ends after the fourth turn (the second magician may need to use **1 telepoint here**). In the fourth test case, the game ends after the third turn (the first magician has to use **1 telepoint here**).

[Home](#) » [Compete](#) » [August Challenge 2012](#) » A Game of Thrones

# A Game of Thrones

Problem Code: GTHRONES

---

Bran and Tyrion are the last two high lords fighting for the Iron Throne. With a mutual agreement that included all knights of the realm, it was decided to settle the issue with a game of thrones, of course not a game of swords but a game of numbers, after all one of them is a cripple and other is a dwarf. Seven wisest men of the realm came forward and forged rules of this game which are as follows :

1. Initially **N** numbers were written down on a notebook (possibly with multiple copies of every number).
2. Players alternate their turns with Bran playing first.
3. In the first turn Bran gets to choose a number of his choice from the numbers written on the notebook and declare it as the current number of the game.
4. After that in every move this is what they do : let's say the current number of the game is **u**. They erase **u** from the notebook (if **u** was written multiple times, they erase it only once) and declare one of the numbers still written on the notebook **v** as current number of the game. **v** can be chosen iff prime factorization of **u** and **v** differ by exactly 1 prime factor. ( Read notes for a more formal definition)
5. The player who can't make a move loses the game.

You're one of Varys' spider and he has asked you to predict the outcome of this game beforehand so that he can devise future strategy. So you've to find out who has a winning strategy assuming both players play optimally.

**Notes:**

1) **v** can be chosen after **u** iff either of the following conditions hold :

1.  $v > u$  and  $u \mid v$  and  $(v/u)$  is prime
2.  $u > v$  and  $v \mid u$  and  $(u/v)$  is prime

2) A natural number is prime if it has exactly two distinct positive factors. 1 is not a prime number.

---

## Input

First line of the input contains a single integer **N** denoting number of different numbers written in the notebook. Then follow **N** lines. Each of the following lines contain two space

separated integers :  $u_i$  and  $c_i$  where  $u_i$  is the  $i^{\text{th}}$  distinct integer written on notebook and it has been repeated  $c_i$  number of times.

## Output

Your program should print Bran if he has a winning strategy else it should print Tyrion. Also in case Bran could win, your program must output the smallest number Bran could choose in the first turn to ensure a win. See sample output for details.

## Example

Input:

3

2 3

14 3

21 2

Output:

Bran 21

## Constraints:

$1 \leq N \leq 500$

$1 \leq u_i \leq 10^{18}$

$1 \leq c_i \leq 10^9$

All  $u_i$  are distinct.

COOK25

[August Cook-Off 2012](#)

19 Aug 2012  
21:00:00

3 hours 20 minutes

1041

[Home](#) » [Compete](#) » [August Cook-Off 2012](#) » Code Crazy Minions

## Code Crazy Minions Problem Code: NOCODING

Coding in Sprout (a programming language) is very intuitive. Chef is giving his minions a demonstration in Sprout and wants you to help him determine if they are not too difficult for them.

A program in Sprout is written using three kinds of instructions.

- Load Instruction: Load a value into buffer.
- Increment Instruction: Increment the value in buffer.
- Print Instruction: Print the value from buffer.

The buffer stores a single integer between 0 and 25, both inclusive. If the value in the buffer is  $x$ , the increment instruction makes the value  $(x+1)$ , if  $x < 25$ . At  $x = 25$  the increment instruction makes the value 0.

Load Instruction can be used to load any value between 0 and 25 (inclusive) into the buffer.

Print from the buffer prints a lowercase English character based on the value in the buffer. Precisely, it prints the  $(x+1)$ th character in the alphabet. Thus, for  $x = 0$ , 'a' is printed;  $x = 1$ , 'b' is printed and so on. For  $x = 25$ , 'z' is printed.

To keep his programs clear, he uses the load instruction only once in the beginning (before printing any character). Then he proceeds instruction after instruction. A program is simple, if the number of instructions is not more than ELEVEN times the length of the word that it prints. Given the word Chef wants his program to print and assuming he will write the shortest code (i.e. use the fewest instructions) to print it, will the program be simple?

### Input

The first Line contains a single number  $T$ , the number of test cases.

Each test case contains 1 word on a line by itself - the word that would be printed by Chef's program.

### Output

Assuming Chef writes the shortest code (with minimum instructions) to print the given word, output "YES" if this code is not more than ELEVEN times the length of the word being printed; "NO" otherwise

### Constraints

$1 \leq T \leq 100$

$1 \leq \text{length of word} \leq 1000$

---

## Sample Input

2

helloworld

mississippi

---

## Sample Output

NO

YES

---

## Explanation

The optimal program length for mississippi is 112 instructions and that is smaller than 121 (length of 'mississippi' \* 11)

[Home](#) » [Compete](#) » [August Cook-Off 2012](#) » Forced Output

## Forced Output Problem Code: YNOUTPUT

The state space of the output of this problem (and as a matter of fact, all the problems in this Cook-Off) is  $-2$  to the power  $T$  - where  $T$  is the number of test cases (so be extra careful!). Each test case consists of  $T$  lines consisting of "YES" or "NO". If a test case accurately represents the output that you would print for this file, then print "YES" for this case. Print "NO" otherwise.

The output for a file is defined as the output for all the test cases one by one. If you output "YES" for test case 'x', then your output must match the input for the test case 'x', and if and only if your output does not match the input for test case 'x', should you print "NO" for that case.

---

## Input

The first Line contains a single number  $T$ , the number of test cases.

Each test case contains  $T$  lines. Each line is either "YES" or "NO". The  $T$  lines together represent the candidate output for this problem.

---

## Output

If the candidate-output (that you got in the input) is what you are going to print then print "YES", and only if it is different, print "NO". The output for each case must be on a single line by itself.

---

## Constraints

$1 \leq T \leq 100$

There is only one unique valid output that you can print

---

## Sample Input

2

NO

NO

NO

YES

---

## Sample Output

NO

YES

---

## Explanation

Your output clearly matches the input for the second case. No other output can be valid for this file.

[Home](#) » [Compete](#) » [August Cook-Off 2012](#) » Asmany Number Verification

## Asmany Number Verification Problem Code: LEBINARY

Asmany strings are strings of '0's and '1's that have as many 00 as 11. A string such as 00110001 consists of 3 "00" and 1 "11". Of course this is not an Asmany string. 0011, 1100, 000111000111 are Asmany strings. An L'th Asmany number is the number of Asmany strings of length L for all positive integers L.

For esoteric purposes Chef had an oracle (a device) that was capable of answering whether a number that he entered was an Asmany number. The problem is that his oracle takes too

long for large numbers. Him being Chef, he wants to ask the oracle very large numbers! You tell him that you can give him a better oracle (a program) that will tell him what he wants to know in the blink of an eye.

---

## Input

The first Line contains a single number  $T$ , the number of test cases.

Each test case contains 1 positive integer  $N$ , with not more than 1000 digits.

---

## Output

Print YES if  $N$  is an Asmany number, NO otherwise.

---

## Constraints

$1 \leq T \leq 100$

$1 \leq \text{Number of digits in } N \leq 1000$

---

## Sample Input

2

3

4

---

## Sample Output

NO

YES

---

## Explanation

4 is an Asmany number. To be precise, it is the 4th Asmany number: There are 4 Asmany strings of length 4. 0011, 1100, 0101, 1010.

## Codechef Password Recovery Problem Code: CHEFPASS

Chef recently decided a very curious encryption algorithm for CodeChef passwords. He took an  $N$ -letter password and found all the possible 2-letter substrings, considering the characters of a password was written in a circle. Thus, for  $AbcD$ , he would deduce the following substrings:  $Ab$ ,  $bc$ ,  $cD$ ,  $DA$ .

Then, some of the substrings were reversed. For example, the set of substrings above could be converted to  $Ab$ ,  $cb$ ,  $Dc$ ,  $DA$ . All the substrings were then stored in a database. Whenever the need be, the substrings were recovered, and a password was generated. Of course sometimes, more than one such password could be generated, but Chef thinks this is perfectly fine!

Unfortunately, a programming error by the new CodeChef minion, led to corruption of this database and mixed all the strings together. Chef now wants to retrieve the passwords, a few at a time. He gives you a set of substrings  $A$  from some of the passwords - these you should always consider, and another set of substrings  $B$  from which you can use some. Can you tell him if it is possible to select all the substrings from  $A$  and some (0 or more) substrings from  $B$  and construct 1 or more passwords? Each selected string can be used only once, for at max 1 password. See explanation of Sample Input for clarification.

### Input

The first Line contains a single number  $T$ , the number of test cases.

Each test case contains 2 lines. The first line of the test case contains the number  $N$  ( $= |A|$ ), followed by the  $N$  substrings Chef gives in  $A$ . The next line contains  $M$  ( $= |B|$ ), followed by  $M$  substrings. Two substrings are input with a single space character between them.

### Output

Print  $T$  lines, one for each test case. Output either "YES" if it is possible to select all strings from  $A$  and some strings from  $B$  such that the overall set of strings represent the substrings for 1 or more passwords. Output "NO" otherwise.

### Constraints

$1 \leq T \leq 100$

$1 \leq N \leq 1000$

$1 \leq M \leq 1000$

All characters would be upper case or lower case English letters. Passwords are case sensitive.

---

## Sample Input

2

4 aB aB Ca Bc

6 aB Ca ca ab ba bc

4 aB aB Ca Bc

3 Ba aC bc

---

## Sample Output

YES

NO

---

## Explanation

In the first Sample Input, you can construct "aBaC" from (aB aB Ca Ca) and "aBc" from (Bc aB ca). Strings selected from A are shown in bold. Remember that substrings can be reversed. Note that although there are repeated strings, each string is used in at most one of the passwords.

[Home](#) » [Compete](#) » [August Cook-Off 2012](#) » Rebuilding Byteland

## Rebuilding Byteland Problem Code: UNFRIEND

---

Byteland is in the midst of a political crisis. The city is in almost a state of civil war among the two factions, "Coders", who rule the city - and "non-Coders" who believe they have been oppressed far too long. Chef - Byteland's President - has decided to restructure Byteland residences such that Coders and non-Coders are forced to meet everyday!

Byteland is a society of N houses - labeled 1 to N, connected by M bi-directional roads - labeled 1 to M. Each road starts at one of the houses and ends at some other house. No two roads meet anywhere, unless of course, they are connecting the same house to other houses. Chef wants to assign Coders and non-Coders among the houses, such that, when a Coder leaves his house on one of the roads, he ends up at a non-Coder's house. Similarly, when a non-Coder leaves his house on one of the roads, he must also always end up at a Coder's house.

The problem is, that under the current scheme of houses and roads, this might not be possible. Thus, Chef has decided to warp some of the roads to another dimension. Warping a road to another dimension, makes the two houses, say A and B, that the road connected - merge into a single house, say C. Roads that were connecting A to other houses are extended

to C, and roads that were connecting B to other houses are also extended to C. A and B may be the same house after some sequence of warps. In this case, if the road is warped, it simply disappears.

If it were up to Chef, he would warp every road in the City and create a Coder non-Coder paradise island; alas, he sometimes cannot do so. For example, only a few roads may be available for warping, since the other roads are too long. Also, it may be possible that after warping some roads, some road connects a house with itself. If such a road cannot be warped, it will be considered against Chef's wishes!

The task of determining a strategy as to which roads to warp has come to you.

Can you tell if it is possible to warp some of the roads that can be warped to fulfil Chef's dream of being able to assign Coders and non-Coders to houses, such that, every remaining road strictly connects a Coder's house and a non-Coder's house.

You may assume that Byteland society is constructed in a way, such that, anyone can traverse from any house to any other house by using some sequence of roads. Also initially, no two roads connect the same pair of houses.

## Input

The first Line contains a single number T, the number of test cases.

The first line of each case contains two numbers, N and M, respectively. The next M lines contain two numbers each, x and y. Both x and y are between 1 and N, inclusive. They describe a bi-directional road between house x and house y. The roads are labeled from 1 to M in the order in which they appear in the input.

The next line contains a single number W, the number of roads that can be warped. The next line contains W unique numbers, separated by a single space. Each number is the label of a road that can be warped. All such numbers are unique and within 1 and M, inclusive.

## Output

Print T lines, one for each test case. Output either "YES" if Chef's dream can be fulfilled, "NO" otherwise.

## Constraints

$$1 \leq T \leq 100$$

$$2 \leq N \leq 50$$

$$1 \leq M \leq 200$$

$1 \leq W \leq M$

---

### Sample Input

2

3 3

1 2

2 3

3 1

2

1 2

5 7

1 3

1 4

3 4

2 4

2 5

4 5

1 2

2

2 4

---

### Sample Output

YES

NO

---

## Explanation

In the first Sample Input, warping either one of Road 1 or Road 2 is sufficient. In the second Sample Input, warping both of Road 1 and Road 2 create a road that connects a house to itself and all other possibilities of warping roads are insufficient to fulfil Chef's dream.

SEP12

[September Challenge 2012](#)

01 Sep 2012  
15:00:00

10 days

3293

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Racing Horses

## Racing Horses Problem Code: HORSES

Chef is very fond of horses. He enjoys watching them race. As expected, he has a stable full of horses. He, along with his friends, goes to his stable during the weekends to watch a few of these horses race. Chef wants his friends to enjoy the race and so he wants the race to be close. This can happen only if the horses are comparable on their skill i.e. the difference in their skills is less.

There are **N** horses in the stable. The skill of the horse **i** is represented by an integer **S[i]**. The Chef needs to pick 2 horses for the race such that the difference in their skills is *minimum*. This way, he would be able to host a very interesting race. Your task is to help him do this and report the minimum difference that is possible between 2 horses in the race.

---

### Input:

First line of the input file contains a single integer **T**, the number of test cases.

Every test case starts with a line containing the integer **N**.

The next line contains **N** space separated integers where the **i**-th integer is **S[i]**.

---

### Output:

For each test case, output a single line containing the minimum difference that is possible.

---

### Constraints:

$1 \leq T \leq 10$

$2 \leq N \leq 5000$

$1 \leq S[i] \leq 1000000000$

---

## Sample Input 1

```
1
5
4 9 1 32 13
```

---

## Sample Output 1

```
3
```

---

## Explanation

The minimum difference can be achieved if we pick horses with skills 1 and 4 for the race.

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Kisses & Hugs

## Kisses & Hugs Problem Code: CKISSHUG

---

Princess Artaoelc greeted her guests by either kissing on the cheek (**K**) or hugging (**H**). From the first guest she kisses, she has a compulsion to necessarily kiss every alternate guest from that first kissed guest. That is if the guests are  $G_1, G_2, \dots, G_i, G_{i+1}, \dots, G_n$  and if she first kissed  $G_i$  then she must necessarily kiss  $G_{i+2}, G_{i+4}, G_{i+6} \dots$  till the last possible guest. Your task is to determine in how many ways she can greet **N** guests.

### Input

First line of the input contains **T** ( $T \leq 1000$ ) denoting the number of test cases.

**T** lines follow each containing a single integer **N** ( $1 \leq N \leq 10^9$ ) denoting the number of guests.

### Output

For each case the output should be a single integer representing the number of ways Artaoelc can greet **N** guests. As the answer can be large print it modulo **1000000007**.

### Example

#### Input

```
3
```

1  
2  
3

## Output

2  
4  
6

### Explanation:

In the first case the possible ways are

K, H

Second case:

KH, HK, HH, KK

Third case:

HHH, HHK, HKH, HKK, KHK, KKK

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Three is Crowd

## Three is Crowd Problem Code: CROWD

---

*Two's company, three's a crowd!*

It's been one year since Chef met his brother. Last year, his younger brother came to visit him during this time of the year. This year, the Chef is planning to go visit his brother. Chef's brother has planned to throw a "Welcome Party" for him. He wants to invite people from his neighbourhood (i.e. from the street where he lives). There are **N** houses on the street in a single line (not considering the brother's house). He wants the party to be fun and he will not like to invite people who might spoil the mood of the party. If people are invited from three consecutive houses on the street, they might create trouble. As they say, three's a crowd! He doesn't want to ruin the Chef's *Welcome Party* and so he will not want to send

invites to *any* three consecutive houses. He wants you to tell him how many ways are there for him to go wrong. Note that he can play safe by not inviting anyone to avoid a *crowd*.

---

### **Input:**

First line of the input contains a single integer **T**, the number of test cases.  
Each test case contains a line containing a single integer **N** described above.

---

### **Output:**

For each test case output a single integer denoting the number of ways the brother can go wrong with planning the party.  
The answer can get quite large. So output the total number of ways modulo  $10^9+7$ .

---

### **Constraints:**

$1 \leq T \leq 10000$

$1 \leq N \leq 10^{15}$

---

### **Example:**

#### **Input:**

2

3

4

#### **Output:**

1

3

### **Explanation:**

Case 1: The only way he can go wrong is by inviting all the houses.

Case 2: First way of getting wrong is by inviting houses (1,2,3). Second way to get wrong is by inviting houses (2,3,4). Third way of going wrong is by inviting all 4 houses i.e. (1,2,3,4).

[Home](#) » [Compete](#) » [September Challenge 2012](#) » ChefTown Parade

## ChefTown Parade Problem Code: CHEFTOWN

ChefTown is the biggest city and the capital of ChefLand. There are  $N$  beautiful buildings: restaurants, museums, living houses with large kitchens and so on. Every building has its height. For every  $i$  ( $1 \leq i \leq N$ ) there is exactly one building with height  $i$ . The buildings are located in a single line from left to right. The height of  $i$ th building is  $H(i)$ . The Mayor of ChefTown wants to organize a parade, where all great chefs will take part. A parade depends of its location. The location of a parade is a segment of consecutive buildings beginning near the building number  $L$  and ending near the building number  $R$  ( $1 \leq L \leq R \leq N$ ). Any parade won't be interesting if it is not hold on an interesting segment of buildings. The segment of buildings is interesting if following are hold:

- Imagine, that we have a segment  $[L, R]$ .
- Let  $K=R-L+1$  be the length of this segment, and  $B$  be a list of heights of the buildings that belong to this segment.
- Let's sort  $B$  in non-decreasing order.
- Segment  $[L, R]$  is interesting if  $B[i]-B[i-1]=1$  for every  $i$  ( $2 \leq i \leq K$ ).

Now the Mayor of ChefTown is interested how many ways to organize an interesting parade of length  $W$  for ChefTown citizens. Help him and find out the number of different parades of length  $W$ , which can be hold in the city. Two parades ( $[L_1, R_1]$  and  $[L_2, R_2]$ ) are considered to be different, if  $L_1 \neq L_2$  or  $R_1 \neq R_2$ .

### Input

Each input file consists of two lines, the first one contains two integers  $N$  and  $W$  ( $1 \leq N \leq 400000$ ,  $1 \leq W \leq N$ ). The second line contains  $N$  numbers  $H(i)$  ( $1 \leq i \leq N$ ) - the heights of the buildings.

### Output

For each test case output a single integer - the number of interesting segments of buildings of length  $W$ .

### Example

**Input 1:**

2 1

2 1

**Input 2:**

4 2

1 2 3 4

**Output for Input 1:**

2

**Output for Input 2:**

3

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Queries About Numbers

## Queries About Numbers Problem Code: QNUMBER

Chef loves number theory very much. Now it is time to solve a new kind of problem.

There is given a natural number **N**. Chef has to answer **Q** queries of the form **T K**.

Here **T** is the type of query and **K** is the natural number.

If **T=1**, Chef must find the number of natural numbers which is divisor of both **N** and **K**.

If **T=2**, Chef must find the number of natural numbers which is divisor of **N** and is divisible by **K**.

If **T=3**, Chef must find the number of natural numbers which is divisor of **N** and is not divisible by **K**.

Chef can solve all these queries, but you will be hungry for night if this happens, because Chef will not have free time to cook a meal. Therefore you compromise with him and decided that everyone must do his/her own job. You must program and Chef must cook.

---

### Input

There will be 2 numbers in the first line: **N** and **Q**.

**Q** lines follow with 2 numbers each: **T** and **K**

---

### Output

For each of the **Q** lines you must output the result for corresponding query in separate line.

---

## Example

**Input:**

12 6

1 6

1 14

2 4

2 3

3 12

3 14

**Output:**

4

2

2

3

5

6

---

## Explanation

Numbers for each query:

{1,2,3,6}

{1,2}

{4,12}

{3,6,12}

$\{1,2,3,4,6\}$   
 $\{1,2,3,4,6,12\}$

---

## Constraints

$1 \leq N \leq 10^{12}$

$1 \leq Q \leq 5 \cdot 10^5$

$1 \leq T \leq 3$

$1 \leq K \leq 10^{12}$

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Chef World

## Chef World Problem Code: CHEFWD

Chef Ciel lives in a long street which can be thought as of x-axis of coordinate system. Her house is at coordinate 0 whereas her restaurant is situated at coordinate  $N$ . Usually Ciel goes from home to restaurant taking a step of size 1 or 2 in forward direction. We all know how much Chef loves Fibonacci numbers.

But today, Ciel being little casual stepped in wrong direction on her way to her restaurant exactly once and of course she did not set her foot wrong at home. Now she wonders how many ways she can reach her restaurant provided that she stepped wrong once but not at home.

She does not go past her restaurant because it is altogether different world and once she reaches her restaurant she stops.

For example, if  $N$  is 3 then

$0 \rightarrow 1 \rightarrow -1 \rightarrow 0 \rightarrow 1 \rightarrow 3$ ,

$0 \rightarrow 2 \rightarrow 1 \rightarrow 3$

are some possible ways where as

$0 \rightarrow -1 \rightarrow 1 \rightarrow 3$ , (She did not set her foot wrong at her home)

$0 \rightarrow 1 \rightarrow 3$ , (She sets her foot wrong direction exactly once)

$0 \rightarrow 1 \rightarrow 0 \rightarrow 3$ , (Her steps are always size 1 or 2)

$0 \rightarrow 1 \rightarrow 2 \rightarrow 4 \rightarrow 3$ , (She does not go past her restaurant)

$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3$  (Once she reaches her restaurant, she stops)

are not.

---

## Input

First line of input contains **T**, number of test cases which is at most 10000. Then **T** lines follows each containing a positive integer **N** which is at most 1000000000000000 (10<sup>15</sup>).

---

## Output

Number of ways Ciel can reach her restaurant modulo 1000000007 (10<sup>9</sup>+7).

Sample Input:

2  
3  
4

Sample Output:

18  
44

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Lights

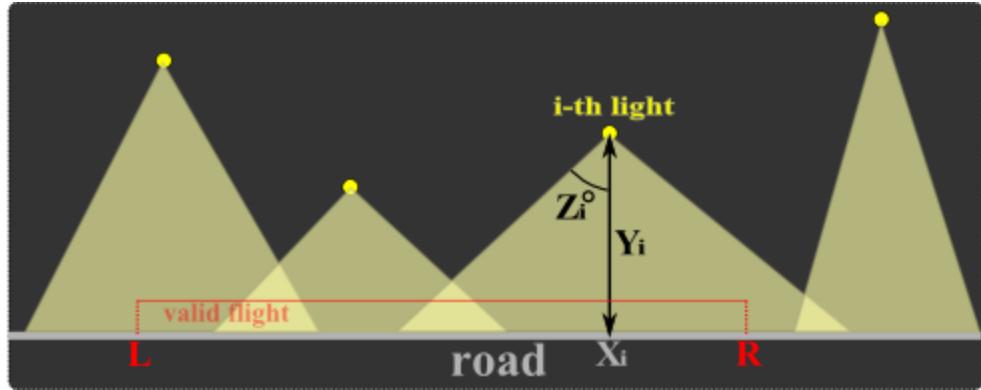
## Lights Problem Code: LIGHT

---

There are **N** lights above a road along the X-axis. The **i**-th light has **X<sub>i</sub>** as its X-coordinate, and it is hung **Y<sub>i</sub>** above the road. The **i**-th light illuminates a triangular area, which is an isosceles triangle and its bottom edge is on the X-axis. The half of the top angle of the triangle is **Z<sub>i</sub>** degrees. (See the below figure)

You have an aircraft which can fly on a fixed height. Because of some unknown reasons, this aircraft can fly only under the light. Your task is to find the maximum possible height for your flight from **X=L** to **X=R**.

The lights do not block the aircraft, e.g. the aircraft can safely fly through a light.



## Input

The first line contains three parameters **N**, **L** and **R**.

The next **N** lines give the information about each light. The **i**-th line contains three numbers **X<sub>i</sub>**, **Y<sub>i</sub>** and **Z<sub>i</sub>**.

## Output

Output the maximum height you can reach. The value must have an absolute error less than or equal to 0.000001 ( $10^{-6}$ ). It is guaranteed that you can make the flight with the positive height.

## Constraints

- $1 \leq N \leq 50000$
- $-1000 \leq L < R \leq 1000$
- $-1000 \leq X_i \leq 1000$
- $0 < Y_i \leq 1000$
- $15 \leq Z_i \leq 75$

**N** is an integer, but all other input values can be non-integers.

## Example

### Input:

2 3.2 7.3

3.2 4.7 28

7.3 4.2 75

Output:

3.300759642

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Simultaneous Nim

## Simultaneous Nim Problem Code: SIMNIM

Chef is a famous [Nim](#) player. He has his own set of **N** heaps, and the **i**-th heap contains **A<sub>i</sub>** stones. What is so special about this set is that with this set of stones Chef always wins when he's moving second. Chef's friends don't know the math behind the game at all, but the fact that Chef is playing with the same set of stones all the time seems suspicious to them.

To prove that he's a really good player even with other sets, Chef decided to split his set into **K** sets so that each of the **N** original heaps is used in exactly one set. Then he's going to play **K** Nim games simultaneously with these sets. Chef's ultimate goal is letting his opponent move first in each of these **K** games and still winning all the games! To make it even more impressive, Chef wants the value of **K** to be as large as possible.

It's of no secret that in the game of Nim a position is winning for the player moving second if XOR of all the numbers of stones in the heaps is equal to 0; for more information check [this](#). You may assume that Chef and his opponents always play optimally -- they always win whenever they have a winning strategy.

You are to help Chef with this separation. Note that this is a challenge problem: it's not required for **K** to be maximum possible, but the bigger is **K**, the more points you get.

---

### Input

The first line of the input file contains one integer **T** -- the number of test cases (no more than 10). Each test case is described by two lines. The first of them contains a single integer **N**, the second contains **N** space-separated positive integers **A<sub>i</sub>** (**i** = 1..**N**).

---

### Output

For each test case, output one line containing exactly **N** positive integers. The **i**-th of these numbers should be the index of the game in which the **i**-th heap of stones should be involved (in the order as the heaps are given in the input). The value of **K** for this test case is then calculated as the maximum of the numbers you printed. Note that each integer between 1 and **K**, inclusive, should be presented among these numbers at least once.

---

### Scoring

If any of the test cases is solved incorrectly, you'll receive Wrong Answer. Incorrect answers include outputting something else than **N** positive integers for each test case, making some sets of heaps empty and making some sets of heaps losing for Chef.

**Note** that it's possible to output the answer with **K** = 1 (i.e., making no splits of the initial set is allowed). Nevertheless, if your output corresponds to **K** = 1 for **all** test cases in a single output file, you'll receive Wrong Answer as well. It's guaranteed that each official test file contains at least one test case for which a solution with **K** > 1 exists.

If all your answers are correct, your score for each test case will be calculated as  $100 * N / K$ . Your score for each file is the average of your scores on the individual test cases in this file. Your overall score is the average of your scores on the individual test files.

Your goal is to minimize the overall score.

---

## Example

### Input:

```
2
7
1 2 3 4 5 6 7
6
1 1 2 2 3 3
```

### Output:

```
1 1 1 2 2 2 2
1 2 1 2 1 2
```

### Explanation:

Both of the answers are correct. The score is  $100 * 7 / 2 = 350$  for the first test case and  $100 * 6 / 2 = 300$  for the second test case. The score for this file is thus  $(350 + 300) / 2 = 325$ . Note that the second test case allows a better solution, 1 1 2 2 3 3.

---

## Test Case Generation

Every official input file contains exactly 10 test cases. In each test case  $N$  is chosen randomly and uniformly between 10 and 1000, inclusive, and  $M$  is chosen randomly and uniformly between 5 and 60, inclusive. Then, the values of  $A_i$  for  $1 \leq i < N$  are generated randomly and uniformly between 1 and  $2^M - 1$ , inclusive, and the value of  $A_N$  is calculated as XOR of all other  $A_i$ 's (note that this way the generated set of heaps is certainly winning for Chef). If the value of  $A_N$  appears to be 0, the whole process of test case generation is restarted.

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Knight Moving

## Knight Moving Problem Code: KNGHTMOV

---

Consider an infinitely large chess table. From the cell  $(0, 0)$ , our knight has to move to the cell  $(X, Y)$  by the rule: our knight could only move from a cell  $(u, v)$  to the cell  $(u+A_x, v+A_y)$  or  $(u+B_x, v+B_y)$  in one move. Note that it may be different from ordinary knight's move of chess.

In addition, there are  $K$  blocked cell(s) on the table where the knight could not move on.

Your task is to count how many distinct ways the knight could complete his mission. Two ways are called "distinct" if and only if they have different numbers of steps or there exists  $i$  such that they are in different cells after  $i$ -th step. Note that our knight may continue to move after he reaches the cell  $(X, Y)$ .

---

### Input

The first line contains an integer  $T$ , denoting the number of test cases. Each test case is described as follows:

- The first line contains 3 integers  $X, Y, K$ .
- The second line contains 4 integers  $A_x, A_y, B_x, B_y$ .
- The third line contains  $K$  pair(s) of integers, each represents co-ordinate of a blocked cell. This line does not exist if  $K = 0$ .

---

### Output

For each test case, output on a line the number of ways found modulo  $1000000007$  ( $10^9 + 7$ ). If there are infinitely many ways, then output -1 instead.

---

### Constraints

$1 \leq T \leq 5$

$0 \leq K \leq 15$

The absolute values of all other input values are at most 500.

$(0, 0)$  is not a blocked cell.

$(X, Y)$  is not a blocked cell.

---

## Example

**Input:**

3

3 3 0

1 2 2 1

9 9 2

1 2 2 1

1 2 6 6

1 1 0

0 0 0 0

**Output:**

2

4

0

## Explanations:

In the first and second examples, our knight's move is the similar to ordinary knight's, but only 2 directions are allowed. In the first example, there are 2 ways  $(0, 0) \rightarrow (1, 2) \rightarrow (3, 3)$  and  $(0, 0) \rightarrow (2, 1) \rightarrow (3, 3)$ .

In the third example, our knight's cannot move toward, so our knight's cannot complete his mission.

[Home](#) » [Compete](#) » [September Challenge 2012](#) » Annual Parade

## Annual Parade Problem Code: PARADE

---

In the magic land, there is an annual parade hold on each spring.

There are **N** cities in magic land, and **M** directed roads between cities.

On the parade, there will be some(may be 0) heroes travel in this land, for each hero: He start at city  $\text{begin}[i]$ , traveling to some cities, and finish at city  $\text{end}[i]$ . Note that:  $\text{begin}[i]$  may be equals to  $\text{end}[i]$ , but he must at least moved to another city during this travel. He can go on one road many times, but it will have a cost for each time.

The cost of this parade is the sum of these items:

1. The sum of costs by traveling on roads. (If a road is passed by  $k$  heroes, then it must be count  $k$  times.)
2. If for a hero, he ended at a city that not equals to his start city, i.e.  $\text{begin}[i] \neq \text{end}[i]$ , then it will cost **C** dollars to move him back to his home.
3. If for a city, there is no heroes visited, then we must pay for the citizen **C** dollars as compensate.

The value of **C** may change every year, and we can predict this value in the following **K** years. Your task is: calculate the minimal cost of each year.

---

### Input

In the first line, there are 3 integers: **N**, **M** and **K**.

In the following **M** lines:

there will be 3 integers: **S[i]**, **T[i]**, and **V[i]**, describing a directed road from **S[i]** to **T[i]**, cost **V[i]** dollars.

In the next **K** lines: There will be an integer: **C[i]**, describing the value of **C** in that year.

---

### Output

Output **K** lines: each line contain an integer, corresponding to the minimal cost of each year.

---

### Constraints

$2 \leq N \leq 250$

$1 \leq M \leq 30000$

$1 \leq K \leq 10000$

$S[i] \neq T[i]$ ,  $1 \leq S[i], T[i] \leq N$

$1 \leq V[i] \leq 10000$

$1 \leq C \leq 10000$

Note that: there may be more than 1 road between a certain pair of cities.

---

## Example

**Input:**

6 5 3

1 3 2

2 3 2

3 4 2

4 5 2

4 6 2

1

5

10

**Output:**

6

21

32

## Explanation

In the first year, since  $C$  is very small, an optimal solution is: no hero travel, we pay 1 dollar for each city as compensate.

In the second year, an optimal solution is: One hero traveling in the path: 1->3->4->5. We pay  $2+2+2=6$  dollars for the roads, 5 dollars for taking him back to city 1, and pay 5 dollars for city 2 and 6 as

compensate.

In the third year, one optimal solution is: One hero traveling in the path: 1->3->4->5, and another hero traveling in the path: 2->3->4->6.

COOK26	<a href="#">September Cook-Off 2012</a>	23 Sep 2012 21:30:00	3 hours 15 minutes	1404
--------	---	-------------------------	--------------------	------

[Home](#) » [Compete](#) » [September Cook-Off 2012](#) » [Carvans](#)

## Carvans Problem Code: CARVANS

Most problems on CodeChef highlight chef's love for food and cooking but little is known about his love for racing sports. He is an avid Formula 1 fan. He went to watch this year's Indian Grand Prix at New Delhi. He noticed that one segment of the circuit was a long straight road. It was impossible for a car to overtake other cars on this segment. Therefore, a car had to lower down its speed if there was a slower car in front of it. While watching the race, Chef started to wonder how many cars were moving at their maximum speed.

Formally, you're given the maximum speed of  $N$  cars in the order they entered the long straight segment of the circuit. Each car prefers to move at its maximum speed. If that's not possible because of the front car being slow, it might have to lower its speed. It still moves at the fastest possible speed while avoiding any collisions. For the purpose of this problem, you can assume that the straight segment is infinitely long.

Count the number of cars which were moving at their maximum speed on the straight segment.

---

### Input

The first line of the input contains a single integer  $T$  denoting the number of test cases to follow. Description of each test case contains 2 lines. The first of these lines contain a single integer  $N$ , the number of cars. The second line contains  $N$  space separated integers, denoting the maximum speed of the cars in the order they entered the long straight segment.

---

### Output

For each test case, output a single line containing the number of cars which were moving at their maximum speed on the segment.

---

### Example

**Input:**

```

3
1
10
3
8 3 6
5
4 5 1 2 3

```

**Output:**

```

1
2
2

```

---

## Constraints

$1 \leq T \leq 100$   
 $1 \leq N \leq 10,000$

All speeds are distinct positive integers that fit in a 32 bit signed integer.  
 Each input file will not be larger than 4 MB (4,000,000,000 bytes) in size.

**WARNING!** The input files are very large. Use faster I/O.

[Home](#) » [Compete](#) » [September Cook-Off 2012](#) » Recipe Reconstruction

## Recipe Reconstruction Problem Code: RRECIPE

Chef had an interesting dream last night. He dreamed of a new revolutionary chicken recipe. When he woke up today he tried very hard to reconstruct the ingredient list. But, he could only remember certain ingredients. To simplify the problem, the ingredient list can be represented by a string of lowercase characters 'a' - 'z'.

Chef can recall some characters of the ingredient list, all the others, he has forgotten. However, he is quite sure that the ingredient list was a palindrome.

You are given the ingredient list Chef dreamed last night. The forgotten characters are represented by a question mark ('?'). Count the number of ways Chef can replace the forgotten characters with characters 'a' - 'z' in such a way that resulting ingredient list is a palindrome.

---

## Input

The first line of input contains a single integer  $T$ , the number of test cases.  $T$  lines follow, each containing a single non-empty string - the ingredient list as recalled by Chef. Whatever letters he couldn't recall are represented by a '?'.

---

## Output

For each test case, output a single line containing the number of valid ways the ingredient list could be completed. Since the answers can be very large, output each answer modulo 10,000,009.

---

## Example

### Input:

```
5
?
??
ab?
a?c
aba
```

### Output:

```
26
26
1
0
```

1

---

## Constraints

$1 \leq T \leq 20$

$1 \leq \text{sum of length of all input strings} \leq 1,000,000$

Each input string contains only lowercase roman letters ('a' - 'z') or question marks.

[Home](#) » [Compete](#) » [September Cook-Off 2012](#) » Prime Permutations

## Prime Permutations Problem Code: PPERM

A permutation of  $N$  **distinct** integers between 1 and  $N$ , both inclusive, is called a prime permutation of size  $N$  iff - for all  $i$  between 1 and  $N$ , the following condition holds:

**The  $i^{\text{th}}$  integer is the  $X^{\text{th}}$  smallest integer in the first  $i$  integers, where  $X$  is either 1 or a prime number.**

Your task is to find out how many prime permutations are there of size  $N$ .

---

### Input

The first line contains a single integer  $T$ , denoting the number of test cases. Then  $T$  lines follow, each containing a single integer  $N$ .

---

### Output

For each test case, output a single line containing the number of prime permutations of size  $N$ . Since the answers can be very large, output each answer modulo 1,000,000,007.

---

### Example

**Input:**

4

1

2

3

4

Output:

1  
2  
6  
18

---

### Constraints:

$$1 \leq T \leq 500,000$$

$$1 \leq N \leq 5,000,000$$

Each input file will not be larger than 4 MB (4,000,000,000 bytes) in size.

**WARNING!** Large I/O files. Use fast I/O methods.

[Home](#) » [Compete](#) » [September Cook-Off 2012](#) » Coal Scam

## Coal Scam Problem Code: COALSCAM

Chef has entered the politics of Byteland. He is a minister in the coal department of the government. Recently, he has been assigned a project on constructing special bidirectional roads for transferring coals in such a way that the whole country is connected by a road network.

Different companies submitted their proposal of roads that they could build along with their estimated cost of the road. Chef has to select a subset of these proposed roads so that all cities of the country are connected.

Chef, corrupt as he is, managed to submit bids by his own company for certain roads. If some of his proposed roads are selected, he'd use low grade material. We can safely assume that the cost of building these roads is nothing to him. He would still receive the price he has quoted in the bid and hence, would make as much profit. He wants to choose some subset of proposed roads so that his own profit is maximised.

Byteland consists of  $N$  cities. There are  $M_1$  proposals to build roads by companies other than Chef's. Chef's company has proposed to build  $M_2$  roads. Chef wants to choose a subset of these roads in such a way that **for each pair of cities, there is exactly one path of roads connecting them**. The profit Chef will gain is equal to the sum of costs of the roads chosen from among those made by Chef's company.

Help Chef choose the roads in such a way that his profit is maximized. If there are multiple ways to achieve that, choose the way where the sum of costs of all chosen roads is minimized.

---

## Input

The first line of the input contains a number  $T$ , the number of test cases. This is followed by the input for each test case one by one. The first line of each test case contains three space-separated integers  $N$ ,  $M_1$ , and  $M_2$ . Then  $M_1$  lines follow, each containing a description of a road proposed by other companies. Each of these lines contains three space-separated integers  $u$ ,  $v$ , and  $c$ , where  $u$  and  $v$  denote the end points of the road and  $c$  denotes the associated cost. Then  $M_2$  lines follow, each containing a description of a road proposed by Chef's company in the same format.

---

## Output

Output a single line for each test case. If it is impossible to choose a subset of the roads in such a way that for each pair of cities, there is exactly one path of roads connecting them, output a single line containing "Impossible" (quotes for clarity and starts with an uppercase 'I') for that test case. Otherwise, output a single line containing two space-separated integers: the maximum profit Chef can gain and the minimum total cost of the chosen roads that yields the maximum profit.

---

## Example

Input:

2

3 2 1

0 1 5

1 2 4

0 1 10

3 1 1

0 1 1

0 1 3

Output:

10 14

Impossible

---

## Constraints

$1 \leq T \leq 5$   
 $2 \leq N \leq 5,000$   
 $1 \leq M1 \leq 20,000$   
 $1 \leq M2 \leq 20,000$

For each proposed road, the end points will be two different cities.

There may be several proposed roads between the same pair of cities.

For each proposed road,  $0 \leq c \leq 1,000,000,000$

Each input file will not be larger than 4 MB (4,000,000,000 bytes) in size.

[Home](#) » [Compete](#) » [September Cook-Off 2012](#) » Pizza Tossing

## Pizza Tossing Problem Code: PTOSS

We all know that Chef has a special pizza recipe. The pizza has been a craze all over Bhiwani. The secret of the softness of the pizza lies in the fact that Chef tosses them high in the air before putting them in the oven. He makes a lot of pizzas each day to fulfill the ever-increasing demand.

At some point of the time, Chef got bored. Therefore, he created a game for himself.

Assume that the pizza has two sides: front side denoted by F, and back side denoted by B. He owns a lucky sequence of sides; i.e., a string of length N over alphabet {F, B}.

Chef will toss the pizza many times. After every toss, he will jot down the side that comes up. All tosses are independent and each side has equal probability of coming up (50%). He has made M tosses today. He will keep tossing until the string denoting last N tosses is equal to his lucky sequence.

Count the expected number of additional tosses Chef will make.

---

## Input

The first line contain a single integer T, the number of test cases. Then T description of test cases follow. Each description consists of two lines. The first line contains a single integer N followed by a string S1, separated by a space. The second line contains a single integer M followed by a string S2, separated by a space

The string S1 is the encoded form of the lucky sequence. S1 will contain only characters 'a' - 'z' (that stand for values 0 - 25), and 'A' - 'F' (that stand for values 26 - 31). To decode S1,

concatenate the values as 5-bit binary numbers of each character of S1. Then, replace 0's with F's and 1's with B's. Chef's lucky sequence is the first N characters in the resulting string.

The string S2 is the encoded form of the sequence of sides Chef has already tossed today, in the same way.

Please see the examples for more clarity.

---

## Output

For each test case, output a single line containing the expected number of additional tosses Chef will make. It can be proved that the answers will always be an integer. Since the answers can be very large, output each answer modulo 1,000,000,007. Note that if Chef has already achieved his lucky sequence today, he can stop right away and therefore you should output 0.

---

## Example Input:

```
3
1 a
0 a
3 a
2 a
8 An
3 B
```

---

## Example Output:

```
2
8
254
```

---

## Explanation:

In the third case, for the lucky sequence, 'A' stands for 26 and its 5-bit binary number is 11010. 'n' stands for 13 and its 5-bit binary number is 01101. Concatenate them to get 1101001101. The decoded string will be BBFBFFBBFB. Chef's lucky sequence is the first eight characters, i.e., BBFBFFBB.

## Constraints:

- $1 \leq |S1|, |S2| \leq 200,000$
- $1 \leq N \leq 5 \times |S1|$
- $0 \leq M \leq 5 \times |S2|$
- (sum of  $|S1|$  for all  $S1$ )  $\leq 1,000,000$
- (sum of  $|S2|$  for all  $S2$ )  $\leq 1,000,000$
- ( $|S|$  denotes the number of characters of  $S$ .)

OCT12

[October Challenge 2012](#)01 Oct 2012  
15:00:00

14 days

2824

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Fierce Battles

## Fierce Battles Problem Code: DRGNBOOL

In the world of **DragonBool** there are fierce warriors called **Soints**. Also there are even fiercer warriors called **Sofloats** – the mortal enemies of **Soints**.

The power of each warrior is determined by the amount of chakra he possesses which is some positive integer. Warriors with zero level of chakra are dead warriors :) When the fight between **Soint** with power  $C_I$  and **Sofloat** with power  $C_F$  occurs the warrior with lower power will die and the winner will lose the amount of chakra that his enemy have possessed before the fight. So three cases are possible:

- $C_I > C_F$ . Then **Sofloat** will die while the new power of **Soint** will be  $C_I - C_F$ .
- $C_I < C_F$ . Then **Soint** will die while the new power of **Sofloat** will be  $C_F - C_I$ .
- $C_I = C_F$ . In this special case both warriors die.

Each warrior (**Soint** or **Sofloat**) has his level of skills which is denoted by some positive integer. The fight between two warriors can occur only when these warriors are **Soint** and **Sofloat** of the same level. In particular, friendly fights are not allowed, i.e., a **Soint** cannot fight with another **Soint** and the same holds for **Sofloats**.

Lets follow the following convention to denote the warriors. A **Soint** of level  $L$  and power  $C$  will be denoted as  $(I, C, L)$ , while **Sofloat** of level  $L$  and power  $C$  will be denoted as  $(F, C, L)$ . Consider some examples. If  $A = (I, 50, 1)$  fights with  $B = (F, 20, 1)$ ,  $B$  dies and  $A$  becomes  $(I, 30, 1)$ . On the other hand,  $(I, 50, 1)$  cannot fight with  $(F, 20, 2)$  as they have different levels.

There is a battle between **Soints** and **Sofloats**. There are  $N$  **Soints** and  $M$  **Sofloats** in all. The battle will consist of series of fights. As was mentioned above in each fight one **Soint** and one **Sofloat** of the same level take part and after the fight the warrior with lower power will die (or both will die if they have the same power). The battle proceeds as long as there exists at least one pair of warriors who can fight. The distribution of warriors by levels satisfies the

following condition: for every **Soint** of level **L** there exists at least one **Sofloat** of the same level **L** and vice-versa. So if for some level **L** we have at least one warrior of this level then there is at least one **Soint** of level **L** and at least one **Sofloat** of level **L**.

There is a powerful wizard, whose name is **SoChef**, on the side of **Soints**. He can increase the amount of chakra of each **Soint** by any number. **SoChef** wants the army of **Soints** to win this battle. But increasing amount of chakra of any **Soint** by one costs him a lot of his magic power. Hence he wants to minimize the total amount of additional chakra he should give to **Soints** in order for them to win. Note, however, that the win here means that all **Sofloats** should be dead regardless of whether any **Soint** is alive. Also note that the battle can proceed by different scenarios and the **SoChef** need to distribute additional chakra among the **Soints** in such a way that they will win for any possible battle scenario. Help **SoChef** and find the minimal amount of additional chakra he should give to **Soints** in order for them to win.

---

## Input

The first line of the input contains an integer **T**, the number of test cases. **T** test cases follow. The first line of each test case contains two space separated integers **N** and **M**. Here **N** is the number of **Soints** participating in the battle and **M** is the number of **Sofloats** for the same. Each of the next **N** lines contains two space separated integers **C<sub>i</sub>** and **L<sub>i</sub>**, the amount of chakra and level of **i**-th **Soint** correspondingly. The next **M** lines describe power and level of **Sofloats** participating in the battle in the same format.

---

## Output

For each test case output a single integer on a single line, the minimum amount of chakra **SoChef** should give to **Soints** in order for them to win the battle.

---

## Constraints

Each integer in the input file is positive and does not exceed **100**. That is

$1 \leq T \leq 100$   
 $1 \leq N \leq 100$   
 $1 \leq M \leq 100$   
 $1 \leq C_i \leq 100$   
 $1 \leq L_i \leq 100$

For every **Soint** of level **L** there exists at least one **Sofloat** of the same level **L** and vice-versa.

**It is guaranteed that each official test file will satisfy all these constraints. You DON'T need to verify them in your program.**

## Example

### Input:

2

2 3

10 1

20 2

5 2

5 2

18 1

5 5

73 87

69 13

36 36

77 46

43 93

49 46

74 93

78 87

99 13

59 36

### Output:

8

---

## Explanation

**Case 1.** The warriors are  $I1 = (I, 10, 1)$ ,  $I2 = (I, 20, 2)$ ,  $F1 = (F, 5, 2)$ ,  $F2 = (F, 5, 2)$ ,  $F3 = (F, 18, 1)$ . Without the **SoChef** help the battle can proceed as follows.

- $I2$  fights with  $F1$ ,  $F1$  dies,  $I2$  becomes  $(I, 15, 2)$ .
- $I2$  fights with  $F2$ ,  $F2$  dies,  $I2$  becomes  $(I, 10, 2)$ .
- $I1$  fights with  $F3$ ,  $I1$  dies,  $F3$  becomes  $(F, 8, 1)$ .

So if **SoChef** will give 8 additional units of chakra to  $I1$  the **Soints** will win the battle and even one **Soint** ( $I2$ ) will left alive. Hence the answer is 8.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » The New Scheme

## The New Scheme Problem Code: NEWSCH

---

Scheme? - Too loudly said. Just a new idea. Now Chef is expanding his business. He wants to make some new restaurants in the big city of Lviv. To make his business competitive he should interest customers. Now he knows how. But don't tell anyone - it is a secret plan. Chef knows four national Ukrainian dishes - **salo**, **borsch**, **varenyky** and **galushky**. It is too few, of course, but enough for the beginning. Every day in his restaurant will be a dish of the day among these four ones. And dishes of the consecutive days must be different. To make the scheme more refined the dish of the first day and the dish of the last day must be different too. Now he wants his assistant to make schedule for some period. Chef suspects that there is more than one possible schedule. Hence he wants his assistant to prepare all possible plans so that he can choose the best one among them. He asks you for help. At first tell him how many such schedules exist. Since the answer can be large output it modulo  $10^9 + 7$ , that is, you need to output the remainder of division of the actual answer by  $10^9 + 7$ .

---

### Input

The first line of the input contains an integer  $T$ , the number of test cases. Each of the following  $T$  lines contains a single integer  $N$  denoting the number of days for which the schedule should be made.

---

### Output

For each test case output a single integer in a separate line, the answer for the corresponding test case.

---

### Constraints

$1 \leq T \leq 100$   
 $2 \leq N \leq 10^9$

## Example

Input:

3  
2  
3  
5

Output:

12  
24  
240

## Explanation

**Case 1.** For  $N = 2$  days we have the following **12** schedules:

First day	Second day
salo	borsch
salo	varenyky
salo	galushky
borsch	salo
borsch	varenyky
borsch	galushky
varenyky	salo

varenyky	borsch
varenyky	galushky
galushky	salo
galushky	borsch
galushky	varenyky

**Case 2.** For  $N = 3$  we have the following 24 schedules:

First day	Second day	Third day
salo	borsch	varenyky
salo	borsch	galushky
salo	varenyky	borsch
salo	varenyky	galushky
salo	galushky	borsch
salo	galushky	varenyky
borsch	salo	varenyky
borsch	salo	galushky
borsch	varenyky	salo
borsch	varenyky	galushky
borsch	galushky	salo
borsch	galushky	varenyky
varenyky	salo	borsch
varenyky	salo	galushky
varenyky	borsch	salo
varenyky	borsch	galushky
varenyky	galushky	salo
varenyky	galushky	borsch
galushky	salo	borsch

galushky	salo	varenyky
galushky	borsch	salo
galushky	borsch	varenyky
galushky	varenyky	salo
galushky	varenyky	borsch

**Case 3.** Don't be afraid. This time we will not provide you with a table of 240 schedules. The only thing we want to mention here is that apart from the previous two cases schedules for other values of **N** can have equal dishes (and even must have for **N > 4**). For example the schedule (**salo, borsch, salo, borsch**) is a correct schedule for **N = 4** while the schedule (**varenyky, salo, galushky, verynky, salo**) is a correct schedule for **N = 5**.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Event Organizer

## Event Organizer Problem Code: MAXCOMP

Chef Po has given an online advertisement to provide **Event organizing services**. Chef got a huge response for his advertisement. He got various orders to conduct the events from different organizations. In turn, Chef will receive a compensation depend upon the type of event and the total numbers of persons in the event. Chef has received **N** orders for conducting events in this weekend in all. As weekend consists of two days all events will take place during the period of **48** hours. For the **i**-th order the corresponding event will start at **S<sub>i</sub>** hours, ends at **E<sub>i</sub>** hours and Chef will receive a compensation **C<sub>i</sub>** for this event. For example, if **S<sub>i</sub> = 17** and **E<sub>i</sub> = 22** then duration of event is **22 – 17 = 5** hours and its time period is **17:00 – 22:00** of Saturday. Hours of Sunday are numbered by numbers from **24** to **48**. So, for example, **10:00** of Sunday will be represented as **10 + 24 = 34**. Because Chef is a newbie, the organizations had put a condition that Chef will receive a compensation for the event if and only if he is available for the entire duration of the event. It means that he can not choose overlapping events. Note, however, that if some event starts just in the moment another event has finished the Chef can safely conduct them both.

In general Chef will obey the orders on first come first serve basis. But on weekends Chef will select the orders in such a way that the total compensation for all the events he will conduct will be the maximal. Now your task is to help Chef and find this maximal total compensation.

---

### Input

The first line of the input contains an integer **T**, the number of test cases. **T** test cases follow. The first line of each test case contains an integer **N**, the number of received orders for conducting events. Each of the next **N** lines contains three space separated integers **S<sub>i</sub>, E<sub>i</sub>, C<sub>i</sub>**, the parameters of the **i**-th event described in the problem statement.

---

### Constraints

1	$\leq$	T	$\leq$	10
1	$\leq$	N	$\leq$	2000
0	$\leq$	$S_i <$	$E_i \leq$	48
$0 \leq C_i \leq 10^6$				

---

## Output

Output for each test case should contain a single integer in a separate line, the maximal compensation Chef Po can get.

---

## Example

### Input:

```

2
4
1 2 100
2 3 200
3 4 1600
1 3 2100
3
1 10 2000
2 5 100
6 9 400

```

### Output:

```

3700
2000

```

---

## Explanation

**Case 1.** The best choice here is to conduct 3rd and 4th events. The total compensation is equal to **1600 + 2100 = 3700**. These events do not overlap since 3rd event starts just after the finish of the 4th one. Alternatively we can conduct first three events that also do not overlap. But in this case compensation will be only **100 + 200 + 1600 = 1900**.

**Case 2.** Note that first event overlaps with both second and third events, while the last two events do not overlap. Hence there are two reasonable choices available for Chef. One is to take just the first order with total compensation **2000** and the second one is to take the last two orders with total compensation **100 + 400 = 500**. Clearly the first choice is better. Hence the answer is **2000**.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Little Elephant and Order

## Little Elephant and Order Problem Code: LUCKY10

The Little Elephant loves lucky strings. Everybody knows that the lucky string is a string of digits that contains only the lucky digits **4** and **7**. For example, strings "**47**", "**744**", "**4**" are lucky while "**5**", "**17**", "**467**" are not.

The Little Elephant has the strings **A** and **B** of digits. These strings are of equal lengths, that is  $|A| = |B|$ . He wants to get some lucky string from them. For this he performs the following operations. At first he arbitrary reorders digits of **A**. Then he arbitrary reorders digits of **B**. After that he creates the string **C** such that its *i*-th digit is the maximum between the *i*-th digit of **A** and the *i*-th digit of **B**. In other words,  $C[i] = \max\{A[i], B[i]\}$  for *i* from **1** to  $|A|$ . After that he removes from **C** all non-lucky digits saving the order of the remaining (lucky) digits. So **C** now becomes a lucky string. For example, if after reordering **A** = "**754**" and **B** = "**873**", then **C** is at first "**874**" and then it becomes "**74**".

The Little Elephant wants the resulting string to be as lucky as possible. The formal definition of this is that the resulting string should be the lexicographically greatest possible string among all the strings that can be obtained from the given strings **A** and **B** by the described process.

### Notes

- $|A|$  denotes the length of the string **A**.
- $A[i]$  denotes the *i*-th digit of the string **A**. Here we numerate the digits starting from **1**. So  $1 \leq i \leq |A|$ .
- The string **A** is called lexicographically greater than the string **B** if either there exists some index *i* such that  $A[i] > B[i]$  and for each  $j < i$  we have  $A[j] = B[j]$ , or **B** is a proper prefix of **A**, that is,  $|A| > |B|$  and first  $|B|$  digits of **A** coincide with the corresponding digits of **B**.

---

### Input

The first line of the input contains a single integer **T**, the number of test cases. **T** test cases follow. Each test case consists of two lines. The first line contains the string **A**. The second line contains the string **B**.

## Output

For each test case output a single line containing the answer for the corresponding test case. Note, that the answer can be an empty string. In this case you should print an empty line for the corresponding test case.

---

## Constraints

$1 \leq T \leq 10000$

$1 \leq |A| \leq 20000$

$|A|$  Each character of  $A$  and  $B$  is a digit.  
Sum of  $|A|$  across all the tests in the input does not exceed **200000**.

---

## Example

Input:

4

4

7

435

479

7

8

1675475

9756417

Output:

7

74

777744

---

## Explanation

**Case 1.** In this case the only possible string **C** we can get is "7" and it is the lucky string.

**Case 2.** If we reorder **A** and **B** as **A** = "543" and **B** = "749" the string **C** will be at first "749" and then becomes "74". It can be shown that this is the lexicographically greatest string for the given **A** and **B**.

**Case 3.** In this case the only possible string **C** we can get is "8" and it becomes an empty string after removing of non-lucky digits.

**Case 4.** If we reorder **A** and **B** as **A** = "7765541" and **B** = "5697714" the string **C** will be at first "7797744" and then becomes "777744". Note that we can construct any lexicographically greater string for the given **A** and **B** since we have only four "sevens" and two "fours" among digits of both strings **A** and **B** as well the constructed string "777744".

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Maximum Sub-rectangle in Matrix

## Maximum Sub-rectangle in Matrix Problem

Code: MAXRECT

You are given a matrix of the size **H** × **W** with integer elements. So it has **H** rows and **W** columns. The rows are numbered by integers in the range **[0, H – 1]** while for the columns the range **[0, W – 1]** is used. Your task is to find out a sub-rectangle of this matrix with a large sum. The sum of a sub-rectangle is the sum of all its elements. However, sub-rectangle does not need to be contiguous. More formally, you have to find out a subset of rows **R** and a subset of columns **C**. Then your sub-rectangle contains all those cells **(r, c)** where **r** is in **R** and **c** is in **C**.

This is a challenge problem so you don't have to find out the optimal solution. You would get a partial score depending on how large the sum of your chosen rectangle is.

---

## Input

The first line of the input contains two space separated integers **H** and **W**, the height and the width of the matrix respectively. Each of the following **H** lines contains **W** space separated integers, the elements of the corresponding row of the matrix.

---

## Output

Your output should consist of 3 lines. The first line of the output should contain two space separated integers **R\_SIZE** and **C\_SIZE**, the sizes of your row set and the column set respectively. In the next line there should be **R\_SIZE** distinct integers in the range **[0, H – 1]** in ascending order, the set of rows you have chosen. In the third line there should be **C\_SIZE** distinct integers in the range **[0, W – 1]** in ascending order, the set of columns you have chosen.

**Your program will get accepted if and only if all of the following conditions are satisfied:**

1. Both your row set and column set are non-empty, that is, **R\_SIZE ≥ 1** and **C\_SIZE ≥ 1**.
2. The second line of your output contains **R\_SIZE** distinct integers in the range **[0, H – 1]** in ascending order.
3. The third line of your output contains **C\_SIZE** distinct integers in the range **[0, W – 1]** in ascending order.
4. The sum of your chosen sub-rectangle is positive. You are guaranteed that such a sub-rectangle always exists.

## Scoring

Let **numerator** = Sum of your chosen sub-rectangle and **denominator** = Sum of all positive elements of the matrix.

Then your score for a single file is equal to **numerator / denominator**. Your total score is the average of individual scores for each file multiplied by **10000** for convenience. Your goal is to maximize your total score.

## Constraints

**1 ≤ H, W ≤ 300**

Each element of the matrix does not exceed **10<sup>9</sup>** by the absolute value. There exists a sub-rectangle of the given matrix which has positive sum. In each official test file **H, W ≥ 200**.

## Example

**Input:**

3 3

-1 -2 4

2 3 -1

8 9 -1

**Output:**

1 2

1

1 2

**Explanation**

In this output the set of one row  $R = \{1\}$  and the set of two columns  $C = \{1, 2\}$  have been chosen. The chosen cells of the matrix are  $M[1][1] = 3$ ,  $M[1][2] = -1$  (recall that the numeration of rows and columns is 0-based). So the sum of the chosen sub-rectangle is  $3 + (-1) = 2 > 0$ . The sum of all positive elements of the matrix is  $4 + 2 + 3 + 8 + 9 = 26$ . So the score you will get for such output is  $2 / 26 = 0.0769231$ . Note that this is not the optimal solution for this matrix.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Need More Diamonds

## Need More Diamonds Problem Code: DIAMOND

**Alice** and **Bob** have decided to play the following game. Initially there are  $N$  diamonds placed in a straight line. Diamonds numbered from 1 to  $N$  in consecutive order. Each diamond has a specific value. The value of  $i$ -th diamond is  $V_i$  for  $i$  from 1 to  $N$ . Players make moves alternatively. In each move player proceeds as follows. If there is only one diamond left the player takes it otherwise he/she tosses the unbiased coin and if it lands on heads he/she takes the **first** diamond among the diamonds that are currently present in a line, otherwise he/she takes the **last** diamond. What is the expected total value **Alice** will get, if she plays **first**? Note that the unbiased coin lands on heads with probability 0.5 (and hence lands on tail with the same probability). Also note that the total value is the sum of values of all diamonds that Alice will get.

**Input**

The first line of the input contains an integer  $T$ , the number of test cases.  $T$  test cases follow. Each test case consists of exactly 2 lines. The first line of each test case contains a single integer  $N$ , the number of diamonds. The second line contains  $N$  space separated integers  $V_1, V_2, \dots, V_N$ , the values of diamonds in the order they lie in a line.

**Output**

Output for each test case should contain a single real number in a separate line, the expected total value Alice will get for this particular test. Your answer will be considered as correct if it has an absolute error less than **10<sup>-2</sup>**.

---

## Constraints

1	$\leq T \leq$	500
1	$\leq N \leq$	2000
$0 \leq V_i \leq 999$		

---

## Example

### Input

```
4
1
100
2
420
3
1 4 9
4
5 5 5
```

### Output

```
100.000
21.000
9.500
10.000
```

---

## Explanation

**Case 1:** In the first move Alice has to take **100**, and this is the only move. Hence, the expected value Alice will get is **100**.

**Case 2:** It is quite clear that Alice will make only 1 move. As she can get **42** or **0** with equal probability, the expected value she will get is equal to  $0.5 \cdot 42 + 0.5 \cdot 0 = 21$ .

**Case 3:** Let's consider all possible scenarios how game can proceed:

1st move	2nd move	3rd move	Alice total value
Alice gets 1	Bob gets 4	Alice gets 9	10
Alice gets 1	Bob gets 9	Alice gets 4	5
Alice gets 9	Bob gets 1	Alice gets 4	13
Alice gets 9	Bob gets 4	Alice gets 1	10

Clearly each of these scenarios will happen with probability  $0.5 \cdot 0.5 = 0.25$ . Hence the expected value Alice will get is equal to  $0.25 \cdot 10 + 0.25 \cdot 5 + 0.25 \cdot 13 + 0.25 \cdot 10 = 9.5$ .

**Case 4:** Out of the 4 diamonds, Alice will get some 2 of them. As all of the diamonds have the same value **5**, the expected value she will get is equal to **5 + 5 = 10**.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Blade Master

## Blade Master Problem Code: BMASTER

Loda and Maelk are legendary ChefCraft players. They have played so many games that this number doesn't fit in a standard 32-bit integer type. Today Loda and Maelk are going to sort the things out and find out who is the greatest ChefCraft player ever. So the great fight is coming. There are a lot of different heroes you may choose to play ChefCraft. Obviously every hero has his pros and cons. Loda perfectly plays Blade Master. His main skill is to make mirror images of himself to spoof the enemy.

As every other popular game ChefCraft is played on a rectangular grid which consists of **N** rows and **M** columns. Rows of the grid are numbered by integers from **1** to **N**. So the upper row of the grid has number **1** and the lower row has number **N**. The same holds for columns. They are numbered by integers from **1** to **M** such that the most left column has number **1** while the most right column has number **M**.

At the beginning of the game the only Blade Master's image stands on some starting cell  $(S_x, S_y)$  where  $1 \leq S_x \leq N$  and  $1 \leq S_y \leq M$ . Then Loda makes  $T$  moves. Maelk knows how the distribution of images on the grid changes after each Loda's move. This happens according to the following rules.

1. If there is an image standing on the cell  $(i, j)$  then the new images appear by the next rules:

- Let  $F(i, j)$  be the total number of images in the "cross" of the cell  $(i, j)$ . The "cross" of the cell  $(i, j)$  is union of all cells in the  $i$ -th row of the grid and in the  $j$ -th column of the grid. So  $N + M - 1$  cells belongs to the "cross".
- Let  $X = F(i, j) \bmod 6$ , that is  $X$  is the remainder of the division of  $F(i, j)$  by 6.
- For every possible value of  $X$  we have following values:  $D_1, D_2, P_1$  and  $P_2$ .
- $D_1$  and  $D_2$  may be equal to one of the 4 values  $['U', 'R', 'D', 'L']$  and mean some two directions.  
Here ' $U$ ' means **up**, ' $R$ ' means **right**, ' $D$ ' means **down** and ' $L$ ' means **left**.
- $P_1$  and  $P_2$  are integer numbers.
- New mirror images will appear at every cell in the direction  $D_1$  with the period  $P_1$  starting from cell  $(i, j)$  and in the direction  $D_2$  with the period  $P_2$  also starting from the cell  $(i, j)$ . Of course, no images will appear out of the grid. For example, if  $D_1 = 'U'$  and  $P_1 = 2$  then images appear at the cells  $(i - 2, j), (i - 4, j), (i - 6, j)$ , and so on.
- Loda always use the same values for  $D_1$  and  $D_2$ . Namely,  
 $D_1 = 'U', D_2 = 'D'$  for  $X = 0$ ,  
 $D_1 = 'L', D_2 = 'R'$  for  $X = 1$ ,  
 $D_1 = 'U', D_2 = 'R'$  for  $X = 2$ ,  
 $D_1 = 'R', D_2 = 'D'$  for  $X = 3$ ,  
 $D_1 = 'D', D_2 = 'L'$  for  $X = 4$ ,  
 $D_1 = 'L', D_2 = 'U'$  for  $X = 5$ .
- But values  $P_1$  and  $P_2$  may vary for different games. But once chosen they will be the same for all moves.

2. Appearing of new mirror images happens immediately.

3. Whenever there is more than one image at the cell they start one on one fights. In each fight two images participate and both die. So if the number of images in the cell was even than all images will disappear in the end, otherwise exactly one image will remain at this cell.

Now Maelk wants to choose his hero in order to win the fight. The most important thing he needs to know for this is how the number of images changes during Loda's moves. So he

asks you for help. Denote by  $C(t)$  the number of images on the grid after the  $t$ -th Loda's move for  $t$  from 1 to  $T$ . For convenience we denote  $C(0) = 1$  with meaning that 0-th move is the putting of the only Blade Master's image at the starting cell. Maelk wants you to calculate the sum  $C(0) + C(1) + \dots + C(T)$ . Since Maelk doesn't know what to expect from Loda he would like to know the answer for several values of  $T$ . As you remember the total number of games played by Maelk and Loda at ChefCraft doesn't fit in a standard 32-bit integer type. Of course, the same can hold for the number of moves in their final fight. Since Maelk plays ChefCraft the whole life he is not strong in math and can't calculate such large sums. So let's help him to win the final fight and become the only ChefCraft master ever.

## Input

The first line of the input contains three space separated integers  $N$ ,  $M$  and  $Q$ . Here  $N$  and  $M$  are sizes of the grid described above and  $Q$  is the number of Maelk's queries. The second line contains two space separated integers  $Sx$  and  $Sy$ , row index and column index of the starting position of the first image. Each of the following six lines contains two space separated integers, the values  $P1$  and  $P2$  for the corresponding  $X$ , that is,  $i$ -th line among these six lines contains values  $P1$  and  $P2$  for  $X = i - 1$ . Each of the following  $Q$  lines contains a single integer  $T$ , the number of Loda's moves for the corresponding Maelk's query.

## Output

For every Maelk's query output on a separate line the numbers of images Maelk will see during Loda's move.

## Constraints

$N$ and $M$ are	positive
$N \cdot M \leq 34$	
$1 \leq Sx \leq N$	
$1 \leq Sy \leq M$	
$1 \leq P1, P2 \leq \max\{N, M\}$	
$1 \leq Q \leq 100$	
$1 \leq T \leq 10^{16}$	

## Example

Input:

3 3 3

1 1

1 1

1 1

1 1

1 1

1 1

1

2

3

**Output:**

4

12

17

## Explanation

The initial grid looks as follows:

```
100
000
000
```

Here '1' represents a cell with an image and '0' represents a free cell.

After the first move grid is

```
111
000
000
```

After the second move grid is

```
101
111
111
```

Finally, after the third move we have

```
011
```

101  
010

So after the first move Maelk will see **3** images, after the second move – **8** images and after the third move – **5** images. Hence the answers for  $T = 1, 2, 3$  are  $1 + 3 = 4$ ,  $1 + 3 + 8 = 12$  and  $1 + 3 + 8 + 5 = 17$  respectively.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Team Formation

## Team Formation Problem Code: TEAMSIZE

Battle of the Chefs is coming up! It's a contest where different teams of different Chefs compete against each other. Our Chef wants to send one of the best teams from his entire staff. Selecting a good team is not about having the best people, but it is about having members who can cooperate and work well with each other. Chef is very aware of this fact. So he chooses the following strategy to decide a team. He lines up all his **N** cooks in a line. Each of these cooks has a skill level associated with him. Denote the level of the  $i$ -th cook as  $\text{skill}[i]$  for  $1 \leq i \leq N$ . Let us make a few points on a team formation.

- A team can have any number of cooks.
- Each team should consist of consecutive cooks. So, for example, cooks with positions 3, 4, 5 can form a team, but cooks with positions 4, 5, 7, 8 can not.
- A team is a *candidate team* if the difference between the maximum skill and the minimum skill of the cooks in the team does not exceed the threshold **C**.
- A cook can be in 0 or more *candidate teams*. Which means that *candidate teams* can intersect.
- Two *candidate teams* are considered to be different if there exists at least one cook who is in one team but not in the other.

Chef has not yet decided the team of what size he is going to send to the battle. To be precise we note, that the team size is the total number of cooks in it. For the given team size Chef wants to consider all candidate teams of this size according to the rules described above, evaluate them for himself and choose the best one. Denote the total number of candidate teams for the given size **K** as **cand[K]**. To increase the chances of choosing better team he wants the number of candidate teams to be as large as possible. But Chef is a human and can't evaluate too many teams. The maximal number of teams he can evaluate depends on many factors such as his mood or fatigue. We call it the Chef's limit. Denote the current Chef's limit as **M**. Since Chef still wants the total number of candidate teams to be as large as possible you need to find the team size **K** such that  $\text{cand}[K] \leq M$ , among all such **K** find the value for which **cand[K]** is maximal and if such **K** is not unique find the minimal **K** among them. Since Chef's limit always changes according to his mood or fatigue you should find the required team size for several values of **M**. Chef also wants to know the total number of candidate teams for the size you will find. There will be **Q** values of **M** in all.

In order to keep the input small the skills of the cooks will be given in the following way.

- Skills of the first **X** cooks are given in the input, where  $X = \min(10000, N)$ .
- If  $N > 10000$  skills of the remaining cooks for  $10000 < i \leq N$  are calculated as  

$$\text{skill}[i] = (A * \text{skill}[i-1] + B * \text{skill}[i-2] + D) \bmod 2^{30},$$

where **A**, **B** and **D** are given in the input. Here **mod** denotes the modulo operation, that is, **Y mod Z** is the remainder of division of **Y** by **Z**.

---

## Input

The first line of the input contains a single integer **T**, the number of test cases to follow. Each test case consists of 3 lines. The first line contains 6 space-separated integers **N**, **C**, **Q**, **A**, **B** and **D**. The second line contains  $X = \min(N, 10000)$  space-separated integers, *i*-th integer among them represents the skill of the *i*-th cook, that is, the number **skill[i]**. The third line contains **Q** space-separated integers, the values of **M** for which you need to find the optimal team size and the number of candidate teams of this size.

---

## Output

For each test case, output **Q** lines containing the optimal team size for the corresponding value of **M** and the number of candidate teams corresponding to this team size separated by a space.

---

## Constraints

$1 \leq T \leq 5$   
 $1 \leq N \leq 500000$   
 $0 \leq C < 2^{30}$   
 $1 \leq Q \leq 1000$   
 $0 \leq A, B, D < 2^{30}$   
 $0 \leq \text{skill}[i] < 2^{30}$  for  $1 \leq i \leq X$   
 $1 \leq M \leq N$

---

## Example

Input:

2

6 3 3 1 1 1

6 2 4 5 6 3

4 3 6

6 10 1 2 2 2

1 1000 1 1000 1 1000

4

**Output:**

2 4

3 3

1 6

2 0

## Explanation

**Case 1.**

With team size 2, we have 4 candidate teams:  $\{2, 4\}$ ,  $\{4, 5\}$ ,  $\{5, 6\}$ , and  $\{6, 3\}$ . Here numbers denote the cooks' skills. With any other team size, the number of candidate teams is either less than 4 or greater than 4. Similarly, with team size 3, we have 3 candidate teams, which are  $\{2, 4, 5\}$ ,  $\{4, 5, 6\}$  and  $\{5, 6, 3\}$ .

With team size 1, we have 6 candidate teams with one member in each team.

**Case 2.**

With team size 1, we have 6 possible candidate teams. But  $6 > 4$ . For all other team sizes, the total number of candidate teams is 0. Thus, the answer is the smallest size among them, i.e., 2.

[Home](#) » [Compete](#) » [October Challenge 2012](#) » Max Circumference

## Max Circumference Problem Code: MAXCIR

In the magic land there is a triangle **ABC**.

You have **N** spell cards. Each card has two parameters, **i**-th card has parameters **x[i]** and **y[i]**. Using of the **i**-th spell card add **x[i]** to the **x**-coordinate of the point **A** and add **y[i]** to the **y**-coordinate of the point **A**. So if **A** has coordinates **(X, Y)** it will be moving to **(X + x[i], Y + y[i])** by using this card.

You can use at most **K** spell cards. Applying of several spell cards performs consecutively. So if **A** was moved to some point **A'** by the first spell card then second spell card is applied to the new position of **A**, that is, to the point **A'**, and so on for further cards.

Your task is to find the maximal possible circumference of the triangle **ABC** after using of at most **K** spell cards (note that you can use no spell cards). The circumference of a

triangle **ABC** is the sum of its side lengths:  $|AB| + |BC| + |CA|$ . And in our problem by triangle we mean any three points **A**, **B**, **C** of the plane. So, in particular, the points **A**, **B**, **C** can lie on the same line, or even some of them can coincide.

---

## Input

The first line of the input contains two integers **N** and **K**. The second line contains two integers **Ax** and **Ay**, the coordinates of the point **A**. The third line contains two integers **Bx** and **By**, the coordinates of the point **B**. The fourth line contains two integers **Cx** and **Cy**, the coordinates of the point **C**. Each of the following **N** lines contains two integers **x[i]** and **y[i]**, the parameters of the **i**-th spell card.

---

## Output

Output should contain a single real number rounded to exactly **13** digits after the decimal point, the maximal possible circumference of the triangle **ABC** that can be achieved by using at most **K** spell cards. Your answer will be considered as correct if it has an absolute error less than  $10^{-12}$ .

---

## Constraints

$$1 \leq N \leq 500$$

$$0 \leq K \leq N$$

$$|Ax|, |Ay|, |Bx|, |By|, |Cx|, |Cy| \leq 10^9$$

$$|x[i]|, |y[i]| \leq 10^6$$


---

## Example

Input 1:

3 2

0 0

1 0

-1 0

0 1

1 0

1 1

**Output 1:**

6.8284271247462

**Input 2:**

3 3

0 0

1 0

-1 0

0 1

1 0

1 1

**Output 2:**

7.8416192529638

**Explanation:**

**Case 1.** Here the optimal way is to use the 1st and the 3rd spell cards. The new coordinates of the point **A** will be **(1, 2)**. So the circumference of the new triangle **ABC** is:  $|(1, 2) - (1, 0)| + |(1, 0) - (-1, 0)| + |(-1, 0) - (1, 2)| = 4 + 2 * \sqrt{2}$ .

**Case 2.** Here the optimal way is to use all of the spell cards. The new coordinates of the point **A** will be **(2, 2)**. So the circumference of the new triangle **ABC** is:  $|(2, 2) - (1, 0)| + |(1, 0) - (-1, 0)| + |(-1, 0) - (2, 2)| = 2 + \sqrt{5} + \sqrt{13}$ .

## Ciel and password cracking Problem Code: CIELHACK

Chef Ciel now faces a crisis situation. It is a really complicated situation, please read carefully. Here, your task is to calculate the minimum expected time for slipping out of the crisis situation.

As you know, Ciel uses a safe having *Box and Ball System* as the combination of the safe (See **NOTE 1**). And, of course, Ciel now knows that the number of the valid combinations for the safe is really large, and it is hard to open it if she forgets her combination. Hence Ciel wrote the combination in a text file and split the text file into **K** pieces, where each piece is compressed with some password. That is, she needs passwords for all **K** pieces for checking the combination of her safe.

Now Ciel has forgotten the combination for her safe and passwords for all **K** pieces! However she remembers that the password for the **i**-th piece is a positive integer not greater than **N<sub>i</sub>**, that is, there are only **N<sub>i</sub>** possibilities for the **i**-th piece's password. So she'd like to try brute-force.

Since her computer is too slow, Ciel decides to use computer centers in her town. There are **C** computer centers in all. Since Ciel is afraid of leaking the combination for her safe, Ciel will use exactly **K** distinct computer centers, one center for one piece.

Currently Ciel is in her restaurant. Apparently her restaurant and all computer centers are located along one street. Hence we will consider Ciel's restaurant and computer centers as points on a coordinate line. Let's assume that Ciel's restaurant has coordinate **0** while the coordinate of the **j**-th computer center is **X<sub>j</sub>**. So the distance between Ciel's restaurant and the **j**-th computer center is  $|X_j|$  meters while the sign of **X<sub>j</sub>** indicates in which direction of the street (positive or negative) Ciel should walk to this center.

After Ciel will choose **K** different centers for cracking passwords of her **K** pieces she will visit them in some order, crack all the pieces (one piece at one computer center) and return back to her restaurant. At each computer center she should wait until the password for the current piece will be cracked and only then she can proceed to the next center. Ciel's walking speed is **V** meters per nanosecond, so she needs exactly  $|A - B| / V$  nanoseconds to walk from the point with coordinate **A** to the point with coordinate **B**.

In the **j**-th computer center, Ciel can use grid computing with at most **P<sub>j</sub>** computers. Each computer is really faster than Ciel's one, it needs exactly **T<sub>j</sub>** nanoseconds for checking one password. However computers are connected by really strange and bad network. So the step-by-step Ciel's plan for cracking the piece is the following:

- Choose some set of computers (among available **P<sub>j</sub>** computers) that will be involved in the cracking process. This takes no time.
- Copy the compressed piece (which she decided to crack in this center) and the program that will be used for cracking directly into one of the chosen computers. This also takes no time.
- Copy the piece and the program to all of the remaining computers by one of the scenarios that she may choose. The copying process will be made by network

connection between computers and may take some time. However, if she chooses only one computer than this step should be omitted.

- Simultaneously run the cracking program on all of the chosen computers.
- Once one of the computers will crack the piece, save the password, terminate all the programs and delete all her information from all the computers. This takes no time since she will simply reboot and clear all the involved computers by the evil red button :)
- Now she can proceed to the next center or to the restaurant if this was the last piece.

Now we describe in detail this plan.

**The cracking program description.** The program that will be used for cracking proceeds as follows. At each step it chooses uniformly at random one of the valid passwords that was not chosen before and check this password (this takes exactly  $T_j$  nanoseconds). If the password is correct the program returns this password to the Ciel and terminates. Note that the computers can't connect to each other while this program is running at each of them. So it is possible that some passwords will be checked for several times at different computers.

**The network connection.** As was mentioned, the computers will distribute the piece and the program through the network connection between them. Bluntly speaking each time each computer that has the piece and the program try to connect to some other computer that has not yet this information and copy it to this computer. This process finishes when all required by Ciel computers have the piece and the program. Since the network connection is weird the time needed for connecting some computer to another computer is random! Namely it has the exponential distribution with mean  $S_j$  (see [wikipedia](#) for details). So the average time for connection between two computers is  $S_j$  nanoseconds. But once connection between two computers is established some files or some data can be copied with no time. Note that computer can try to connect only to one other computer at the same time. However several computers can try to connect to the same computer and once one of them is connected all other computers will be immediately notified about this and stop the connection process with this computer.

**The connection scenario details.** Ciel can control how the computers will try to connect to each other. Namely, for each computer she can provide a preferences list, that is, the order in which this particular computer will try to connect to other computers after it receives the piece and the program. Initially the computer that receives the piece and the program from Ciel will start the process. Consider some computer **A** that has already received the piece and the program. It considers computers in its list one by one. If the current computer **B** in its list has already the piece and the program then it will be skipped. This check requires no time since it can be made by instant ping from **A** to **B**. Otherwise the computer **A** will try to connect to the computer **B** and once connected copy required information to it. After that the computer **A** will proceed to the next computer in its list. It is possible that some other computers are trying to connect to the computer **B** at the same time when **A** is doing this. In this case the first computer that establishes the connection will copy the piece and the program to **B** while other computers stop the connection process with **B** and proceed to the next computers in their lists. Only when all required computers will have the piece and the program Ciel simultaneously execute the program in all of them. Some examples for cracking are in **NOTE 2**.

So Ciel should choose the using computer center for each piece, choose the number of using computers and connection scenario in each computer center in order to minimize the expected needed total time to crack all the pieces and return back to her restaurant.

**NOTE 1.** If you've never seen Box and Ball System, you may find it in [BBSYSTEM](#). But it is not needed to understand Box and Ball System for solving this problem.

**NOTE 2.** Consider the process of cracking the password with  $N_i = 3$  by using 4 computers labeled by integers from 1 to 4. Let the correct password be 1, and wrong passwords be 2 and 3. Also since all computers are the same we may assume that Ciel will copy the piece and the program to the first computer and always choose the first several computers. Some of Ciel's choices are the following:

**1)** Ciel uses only 1 computer. In this case the time for connections is not needed. The running time is  $T_j$  nanoseconds if the computer checks 1 at first (probability 1/3). The running time is  $2 * T_j$  nanoseconds if the computer checks passwords in order (2, 1) or (3, 1) (probability 1/3). The running time is  $3 * T_j$  nanoseconds if the computer checks passwords in order (2, 3, 1) or (3, 2, 1) (probability 1/3). The expected needed time is  $T_j * (1/3) + 2 * T_j * (1/3) + 3 * T_j * (1/3) = 2 * T_j$  nanoseconds.

**2)** Ciel uses 2 computers. Then there is only one scenario of how the piece and the program will be distributed among the computers: computer 1 will send the program and the piece to the computer 2. The expected time for this is  $S_j$  nanoseconds. The expected running time of the cracking program is more complicated to calculate. For example, the running time will be  $T_j$  nanoseconds if the pair of the first checked passwords by the computers is one of (1, 1), (1, 2), (1, 3), (2, 1), (3, 1), where the first elements is checked by the computer 1, and second is checked the computer 2. The running time may be  $2 * T_j$  or  $3 * T_j$  nanoseconds in many cases.

**3)** Ciel uses 3 computers. In this case several preferences lists choices are available. For example, one of such choices can be represented as  $\{(2, 3), (1, 3), (2, 1)\}$ . Here (2, 3) is the preference list of computer 1, (1, 3) is the preference list of computer 2 and (2, 1) is the preference list of computer 3. For this scenario at first computer 1 will send the piece and the program to computer 2. And then both computers 1 and 2 will try to connect to computer 3. If, for example, computer 2 will be connected to computer 3 at first then it copy the required information while the computer 1's trial will be canceled immediately. Then 3 computers may run the brute-force program. Here the expected time is secret. Please calculate by yourself.

**4)** Ciel uses 4 computers. Here we consider two examples of how preferences lists can be chosen.

**4.1)** The lists are  $\{(2, 3, 4), (1, 3, 4), (1, 2, 4), (1, 2, 3)\}$ . Here computer 1 sends the piece and the program to computer 2 at first. Then both computers 1 and 2 will try to connect to computer 3 and first connected computer will copy the required information, while other computer's trial will be canceled. Finally all 3 computers 1, 2 and 3 will try to connect to computer 4 and first connected computer will copy the required information, while other computer's trials will be canceled.

**4.2)** The lists are  $\{(2, 3, 4), (1, 4, 3), (1, 2, 4), (1, 2, 3)\}$ . Here computer 1 sends the piece and the program to computer 2 at first. Then computer 1

will try to connect to computer 3, while computer 2 will try to connect to computer 4. Assume that computer 1 will establish the connection earlier than computer 2. Then computer 1 will copy the required information to computer 3 and both computers 1 and 3 will try to connect to computer 4. Note that computer 2 is still trying to connect to computer 4. So we will have all 3 computers trying to connect to computer 4 at the same time. Though computer 2 started this process earlier, any computer among 1, 2, 3 can be the first connected to computer 4 with some probability. As before the first connected computer will copy the required information, while other computer's trials will be canceled.

---

## Input

The first line of the input contains a single integer  $Q$ , the number of test cases. Each test case is described as follows:

- The first line is empty.
  - The second line contains 3 numbers  $C, K$  and  $V$ .
  - The third line has  $K$  integers, where the  $i$ -th integer denotes  $N_i$ .
  - The next  $C$  lines describe the information about computer centers. The  $j$ -th of them contains 4 numbers  $P_j, S_j, T_j$  and  $X_j$ .
- 

## Output

Output for each test case should contain a single real number in a separate line, the minimum expected time for slipping out of the crisis situation. Your answer will be considered as correct if it has a relative error less than  $10^{-6}$ .

---

## Constraints

1  $\leq Q \leq$  5000 ( $Q$  is an integer)  
 1  $\leq C \leq$  1000 ( $C$  is an integer)  
 The sum of  $C$  in one test file is at most 5000.  
 1  $\leq K \leq$   $\min(5, C)$  ( $K$  is an integer)  
 1  $\leq V \leq$   $10^{20}$  ( $V$  can be a non-integer)  
 1  $\leq N_i \leq$   $10^{18}$  ( $N_i$  are integers)  
 1  $\leq P_j \leq$   $10^{18}$  ( $P_j$  are integers)  
 1  $\leq S_j, T_j \leq$   $10^{20}$  ( $S_j$  and  $T_j$  can be non-integers)  
 $-10^{20} \leq X_j \leq 10^{20}$  ( $X_j$  can be non-integers)

Every real value has at most 6 digits after the decimal point.

---

## Example

Input:

1 1 1.0

100

1 1.0 1.0 1.0

2 1 1.0

2

1 1.0 100.0 1.0

2 1.0 100.0 -2.0

1 1 1.0

3

3 34.0 97.0 1.0

104 4.1873

1278 9877 7383 8173

7612 731.371 35.0 726.123

9171 461.487 62.0 -39.340

1253 836.177 12.0 34.354

9134 888.228 56.0 384.172

1937 585.858 34.0 222.058

9938 992.124 50.0 333.058

1230 342.314 58.0 444.058

982 453.124 31.0 -555.058

7846 274.126 58.0 812.384

4526 553.412 44.0 58.58

**Output:**

52.5

130

182.3333333333333333333333

12938.4586699

## Explanation

**Case 1.** Here Ciel has no choice other than she goes to the first computer center and cracks the password for the only piece by using 1 computer, then Ciel back to her restaurant. The answer is 1 (going to the computer center) + 50.5 (cracking) + 1 (back to her restaurant) = 52.5. In the cracking phase since Ciel uses only one computer the time for connection is not needed. The program will check passwords one by one and 1 nanosecond is needed for checking one password. So the running time will be  $k$  nanoseconds with probability 1% for  $1 \leq k \leq 100$ . Therefore the expected time for the cracking phase is  $1 * 0.01 + 2 * 0.01 + \dots + 100 * 0.01 = 50.5$ .

**Case 2.** Here Ciel has 3 choices.

- 1) The first computer center is used with 1 computer (expected time =  $1+150+1=152$ )
- 2) The second computer center is used with 1 computer (expected time =  $2+150+2=154$ )
- 3) The second computer center is used with 2 computers (expected time =  $2+126+2=130$ )

Therefore the minimum expected time is 130. Here the expected time for cracking phase in the last choice is calculated as follows:

At first, the program and piece are sent by computer 1 to computer 2, and the needed average time is 1 nanoseconds. Then 2 computers run the brute-force program. Let the correct password be 1, and the wrong one be 2. When (Password checked by first computer at first, Password checked by second computer at first) = (1, 1), (1, 2), (2, 1), Ciel will leave the computer center after this with program's running time 100. The probability of this case is 75%. Otherwise Ciel will leave with program's running time 200. Therefore the expected time for this computer center is  $1 + 100*0.75 + 200*0.25 = 126$ .

**Case 3.** Here Ciel also has 3 choices.

- 1) Ciel uses 1 computer: The expected time for connections = 0, the expected time for the program = 194.

- 2) Ciel uses 2 computers: The expected time for connections = 34, the expected time for the program = 150.889.  
 3) Ciel uses 3 computers: The expected time for connections = 51, the expected time for the program = 129.333.  
 So the optimal way is using 3 computers.

COOK27	<a href="#">October Cook-Off 2012</a>	21 Oct 2012 21:30:00	2 hours 30 minutes	1472
--------	---------------------------------------	-------------------------	--------------------	------

[Home](#) » [Compete](#) » [October Cook-Off 2012](#) » Buying Sweets

## Buying Sweets Problem Code: BUYING2

Banknotes in the state of Strangeland don't have any regulated values like 1, 5, 10, 20, 50, 100, etc. In fact, it's possible to see any positive integer on a banknote of Strangeland. Indeed, this isn't the most convenient thing.

Ann is working as a sweet seller at a shop in Strangeland. Every kind of sweets in this shop has its own cost, and sweets of the same kind have the same cost.

Customers in Strangeland are strange. A customer points at some kind of sweets and gives several banknotes to the seller. This means that he wants to buy a positive number of sweets of that kind. He doesn't tell the exact number of sweets he wants to buy. The only thing Ann knows is: **an 'adequate' customer won't give any extra banknotes**. It means that if you throw away any banknote, the resulting amount of money won't be enough to buy the wanted number of sweets.

Ann has to determine the number of sweets the customer wants. Help Ann write a program which determines this number or tells that it's impossible.

---

### Input

The first line of the input contains a single integer **T**, the number of test cases (no more than 20). **T** test cases follow. Each test case consists of two lines. The first of these lines contains two integers **N** and **X** ( $1 \leq N, X \leq 100$ ) separated by a single space. **N** is the number of banknotes given by the customer. **X** is the cost of a single sweet of the chosen kind. The second of these lines contains **N** space-separated integers **A<sub>i</sub>** ( $1 \leq A_i \leq 100$ ), the values of the banknotes.

---

### Output

For each test case output exactly one line containing a single integer:

- **-1** if the customer is inadequate and has given extra banknotes, or
- **K**, the number of sweets the customer wants to buy. If there are several possible answers, output the **largest** of them.

## Example

Input:

3

4 7

10 4 8 5

1 10

12

2 10

20 50

Output:

-1

1

7

---

## Explanation

In the first test case, the total amount of money received from the customer is 27. He can't buy more than 3 sweets with this amount. If you throw away the banknote of value 5 (or of value 4), it will still be possible to buy 3 sweets. Hence the customer is inadequate.

In the second test case, it's clear that he wants to buy just one sweet (note that this number should be positive).

In the third test case, the total amount of money received is 70, which is enough for 7 sweets. The customer might want to buy only 6 sweets, but we are interested in the largest possible answer.

[Home](#) » [Compete](#) » [October Cook-Off 2012](#) » Tennis Tournament

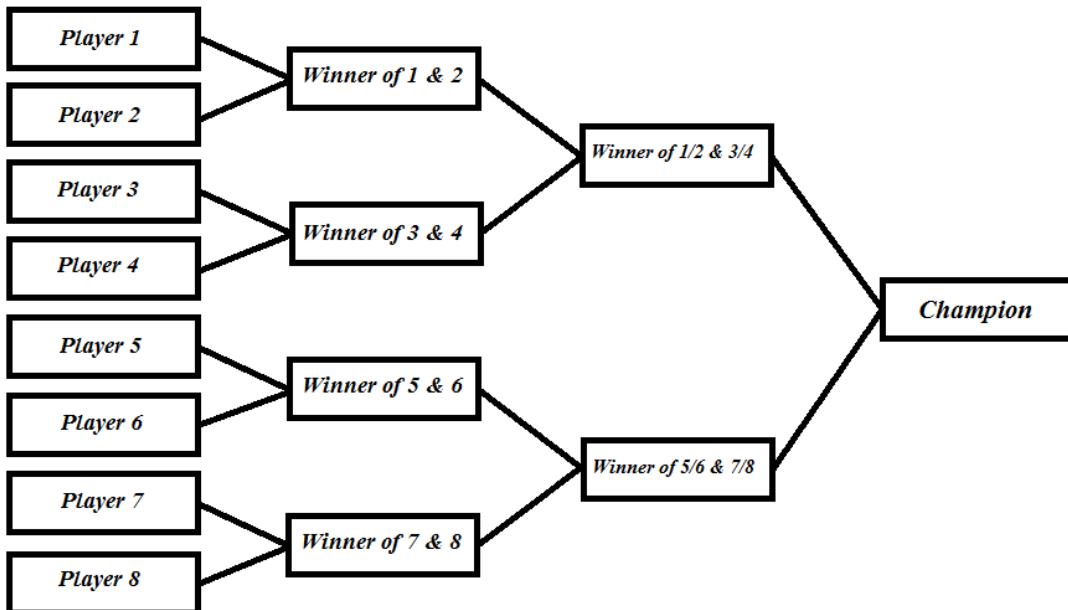
# Tennis Tournament

Problem Code: TOURNAM

Caroline is a huge tennis fan. She loves watching tennis and especially rooting for players of Strangeland, her native country.

Generally, tennis tournaments are played using the Olympic system. Let's consider such a tournament with  $N = 2^K$  players. This tournament is held in  $K$  rounds. In each round, every player plays exactly one match against some other player. The loser of each match is eliminated, and the winner advances to the next round. After the  $K^{\text{th}}$  round, there's only one non-eliminated player who is declared the **champion** of the tournament.

An example of such a tournament for  $N = 8$  can be seen in the picture below.



We'll number the players from **1 to  $N$**  from **top to bottom** in the tournament draw for a given  $N$ . In the first round players 1 and 2, 3 and 4, ...,  $N-1$  and  $N$  play a match against each other. In the second round the winners of the match between players 1 and 2 and the match between players 3 and 4 play together, the winners of the match between players 5 and 6 and the match between players 7 and 8 play together, ..., the winners of the match between players  $N-3$  and  $N-2$  and the match between players  $N-1$  and  $N$  play together. The matches are played in the same way till the  $K^{\text{th}}$  (final) round.

A big tennis tournament is going to start soon.  $M$  Strangelandian players are taking part in this tournament, and their positions in the tournament draw are known in advance. Caroline wants to know the probability that a Strangelandian player will be the champion of this tournament. She collected the information from the previous ten years of tennis history and knows that a Strangelandian player wins a match over a non-Strangelandian player with probability  $P$  percents. All non-Strangelandian players look the same to Caroline, so she considers the chance of winning a match between two non-Strangelandian players to be 50 percents for both players. The same applies to all Strangelandian players. For Caroline, all of

them are equally strong, and she thinks that both of them can win a Strangelandian derby with 50-percent probability.

Help Caroline write a program which calculates the probability that a Strangelandian player will win the tournament.

---

## Input

The first line of the input contains a single integer  $T$ , the number of test cases (no more than 20). Each test case is described by exactly two lines. The first of these lines contains three space-separated integers  $N$ ,  $M$  and  $P$  ( $2 \leq N \leq 2^{30}$ ,  $N = 2^K$  for some integer  $K$ ,  $1 \leq M \leq 10000$ ,  $M \leq N$ ,  $0 \leq P \leq 100$ ) -- the total number of players in the tournament, the number of competing Strangelandian players and the probability that a Strangelandian player beats a non-Strangelandian player in percents, respectively. The second of these lines contains  $M$  space-separated distinct integers  $A_i$  ( $1 \leq A_i \leq N$ ), the 1-based positions of Strangelandian players in the tournament draw.

---

## Output

For each test case output exactly one line containing the required probability in percents. Your answer will be considered correct if its absolute error doesn't exceed  $10^{-6}$ .

---

## Example

Input:

4

2 1 4 2

1

4 2 6 6

1 2

4 2 6 6

1 3

8 3 7 1

3 5 2

**Output:**

42.00000000000000

66.00000000000000

73.18080000000000

75.47784648840000

[Home](#) » [Compete](#) » [October Cook-Off 2012](#) » Equalization

## Equalization Problem Code: EQUALIZ

Bob has an array consisting of **N** integers. He wants to perform a sequence of operations after which all the numbers in the array will be equal.

Each operation consists of choosing any **prime** integer **P**. Bob then chooses any **P** numbers from the array and replaces each of them by the **arithmetic mean** of these **P** numbers (leaving them at the same places in the array as before this operation). Note that the resulting numbers may not be integer.

Help Bob, find a sequence of operations which achieves his goal. It is guaranteed that this goal is always achievable. The sequence is **not** required to be the shortest, but it should consist of **no more than 1000** operations.

### Input

The first line of the input contains a single integer **T**, the number of test cases (no more than 20). Each test case is described by two lines. The first line contains a single integer **N** ( $2 \leq N \leq 30$ ). The second line contains **N** integers **A<sub>i</sub>** ( $1 \leq A_i \leq 1000$ ) separated by single spaces.

### Output

For each test case output a line containing a single integer **K** ( $0 \leq K \leq 1000$ ), the number of operations in your sequence, followed by **K** lines describing the operations in the order of execution. In the **i<sup>th</sup>** of these lines, print a prime integer **P<sub>i</sub>** ( $2 \leq P_i \leq N$ ) followed by **P<sub>i</sub>** distinct space-separated integers **B<sub>i,j</sub>** ( $1 \leq B_{i,j} \leq N$ ), the **1-based indices** of the elements in the array which are chosen for this operation. See the Example Section below for clarity.

### Example

**Input:**

6

4

5 6 7 8

7

9 5 4 9 4 2 11

6

42 42 42 42 42 42

6

9 1 8 2 7 3

12

7 2 6 9 3 9 11 2 5 1 4 13

9

12 10 5 14 16 3 2 1 9

**Output:**

2

2 2 3

2 1 4

1

7 3 1 7 4 6 2 5

0

5

3 1 2 3

3 4 5 6

2 1 4

2 2 5

2 3 6

1 6

3 1 2 3

3 4 5 6

2 1 4

2 2 5

2 3 6

3 7 8 9

3 1 0 1 1 1 2

2 7 1 0

2 8 1 1

2 9 1 2

2 1 7

2 2 8

2 3 9

2 4 1 0

2 5 1 1

2 6 1 2

6

3 1 2 3

3 4 5 6

3 7 8 9

3 1 4 7

3 2 5 8

3 3 6 9

## Explanation

In the first test case, the initial array is (5, 6, 7, 8). The array after the first operation is (5, 6.5, 6.5, 8). The array after the second operation is (6.5, 6.5, 6.5, 6.5).

In the second test case, all the elements of the array become equal to 44/7 after the only operation.

In the third test case, all the elements of the array are already equal, so no operations are needed.

In the last test case, after the first 3 operations the array becomes (9, 9, 9, 11, 11, 11, 4, 4, 4). After all the operations are complete, the array looks like (8, 8, 8, 8, 8, 8, 8, 8, 8).

[Home](#) » [Compete](#) » [October Cook-Off 2012](#) » Tree Fun

## Tree Fun Problem Code: TREEFUN

David is a computer science student who loves his future profession. He is one of those who think that trees grow down, not up.

David likes connected undirected acyclic graphs, also known as trees. He especially likes solving problems about trees. Recently David found a piece of paper with a tree with **N** vertices drawn on it. He also found **M** queries on the same piece of paper, where each query was a non-empty list of some vertices of this tree. For each query, it was asked to find the number of vertices in the tree (which didn't belong to the list) such that if you removed this vertex and all edges incident to it from the tree, then all vertices from the list would belong to pairwise different connected components.

David spent seven unhappy days and six sleepless nights trying to solve this problem efficiently. He is still trying. As you're known to be a good problem solver, David's friend asked for your help. David won't sleep until he knows the answer to each query. Write a program which answers all the queries correctly.

## Input

The first line of the input contains a single integer  $T$ , the number of test cases (no more than 5). Each test case is described as follows. The first line contains two integers  $N$  and  $M$  ( $2 \leq N \leq 50000$ ,  $1 \leq M \leq 50000$ ), the number of vertices in the tree and the number of queries, respectively.  $N-1$  lines follow, describing the edges of the tree. The  $i^{\text{th}}$  line contains two integers  $X_i$  and  $Y_i$  ( $1 \leq X_i, Y_i \leq N$ ,  $X_i \neq Y_i$ ), the indices of the vertices connected by the  $i^{\text{th}}$  edge. Then  $M$  lines follow, describing the queries. The  $i^{\text{th}}$  line contains an integer  $K_i$  ( $2 \leq K_i \leq N$ ) followed by  $K_i$  distinct integers  $A_{i,j}$  ( $1 \leq A_{i,j} \leq N$ ), the indices of the vertices in the  $i^{\text{th}}$  query.

It is guaranteed that the given graph is a tree in all the test cases. The sum of all  $K_i$  in each test case doesn't exceed 100000.

---

## Output

For each test case output  $M$  lines containing a single integer each -- the answer to the corresponding query from the input.

---

## Example

### Input:

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

### Output:

```
2
```

1  
0

---

## Explanation

In the first query, vertex 1 is such a vertex. If you remove vertex 1 and all the edges that connect 1 to other vertices, then vertices 2 and 5 will belong to separate connected components. Similarly, vertex 4 is also valid.

In the second query, vertex 1 is the only valid vertex. If you remove vertex 1 and all the edges that connect 1 to other vertices, then the connected components will be {2}, {3} and {4, 5}. It's easy to see that vertices 2 and 3 will belong to separate components, vertices 3 and 4 will belong to separate components, and vertices 2 and 4 will belong to separate components.

In the third query, you cannot remove a single vertex from the set {2, 4} (and the corresponding edges) so that any pair of vertices from the set {1, 3, 5} will be in separate connected components.

[Home](#) » [Compete](#) » [October Cook-Off 2012](#) » Rearrangement

## Rearrangement Problem Code: REARRAN

Emily has an array of **N** integers. She wants to rearrange its elements in such a way that the sum of any **K consecutive** elements of the rearranged array is divisible by **M**. She was able to accomplish this for several small cases. Help her with the larger ones.

---

### Input

The first line of the input contains a single integer **T**, the number of test cases (no more than 5). Each test case is described by exactly two lines. The first of these lines contains three space-separated integers **N**, **M** and **K**, respectively ( $1 \leq K \leq 70$ ,  $2*K \leq N \leq 10000$ ,  $1 \leq M \leq 10^9$ ). The second of these lines contains **N** space-separated integers **A<sub>i</sub>** ( $0 \leq A_i \leq 10^9$ ).

---

### Output

For each test case output exactly one line containing either a single integer -1 if there is no solution or a sequence of **N** space-separated integers which has the required property and is a permutation of the array **A** from the input. If there are several ways to do this, any of them will be accepted.

---

### Example

Input:

3

6 4 3

6 1 1 9 2 5

8 2 3

2 5 0 9 1 9 9 4

8 5 3

1 9 0 1 1 9 9 1

**Output:**

2 9 1 6 5 1

2 5 9 0 9 9 4 1

-1

## Explanation

There are 6 sequences of 3 consecutive numbers in (2 5 9 0 9 9 4 1). Formally:

- (2 5 9), sum = 16, which is divisible by 2
- (5 9 0), sum = 14, which is divisible by 2
- (9 0 9), sum = 18, which is divisible by 2
- (0 9 9), sum = 18, which is divisible by 2
- (9 9 4), sum = 22, which is divisible by 2
- (9 4 1), sum = 14, which is divisible by 2

Another possible answer is (4 5 9 0 1 9 2 9).

NOV12

[November Challenge 2012](#)01 Nov 2012  
15:00:00

11 days

2572

[Home](#) » [Compete](#) » [November Challenge 2012](#) » [Coin Flip](#)

## Coin Flip Problem Code: CONFLIP

Little Elephant was fond of inventing new games. After a lot of research, Little Elephant came to know that most of the animals in the forest were showing less interest to play the

multi-player games. Little Elephant had started to invent single player games, and succeeded in inventing the new single player game named COIN FLIP.

In this game the player will use  $NN$  coins numbered from  $11$  to  $NN$ , and all the coins will be facing in "Same direction" (Either Head or Tail), which will be decided by the player before starting of the game.

The player needs to play  $NN$  rounds. In the  $kk$ -th round the player will flip the face of the all coins whose number is less than or equal to  $kk$ . That is, the face of coin  $ii$  will be reversed, from Head to Tail, or, from Tail to Head, for  $i \leq k \leq k$ .

Elephant needs to guess the total number of coins showing a particular face after playing  $NN$  rounds. Elephant really becomes quite fond of this game COIN FLIP so Elephant plays  $GG$  times. Please help the Elephant to find out the answer.

---

### Input:

- The first line of input contains an integer  $TT$ , denoting the number of test cases. Then  $TT$  test cases follow.
- The first line of each test contains an integer  $GG$ , denoting the number of games played by Elephant. Each of the following  $GG$  lines denotes a single game, and contains  $33$  space-separated integers  $II$ ,  $NN$ ,  $QQ$ , where  $II$  denotes the initial state of the coins,  $NN$  denotes the number of coins and rounds, and  $QQ$ , which is either  $11$ , or  $22$  as explained below.

Here  $I=1I=1$  means all coins are showing Head in the start of the game, and  $I=2I=2$  means all coins are showing Tail in the start of the game.  $Q=1Q=1$  means Elephant needs to guess the total number of coins showing Head in the end of the game, and  $Q=2Q=2$  means Elephant needs to guess the total number of coins showing Tail in the end of the game.

---

### Output:

For each game, output one integer denoting the total number of coins showing the particular face in the end of the game.

---

### Constraints:

$1 \leq T \leq 10$

$1 \leq G \leq 2000$

$1 \leq N \leq 10$

$1 \leq I \leq 2$

$1 \leq Q \leq 2$

---

**Sample Input:**

```

1
2
1 5 1
1 5 2

```

---

**Sample Output:**

```

2
3

```

---

**Explanation:**

In the 1st game in Example,  $I=1$ , so initial arrangement of coins are H H H H H, and now Elephant will play 5 rounds and coin faces will be changed as follows: After the 1st Round: T H H H H After the 2nd Round: H T H H H After the 3rd Round: T H T H H After the 4th Round: H T H T H After the 5th Round: T H T H T Finally  $Q=1$ , so we need to find the total number of coins showing Head, which is 2

In the 2nd game in Example: This is similar to the 1st game, except Elephant needs to find the total number of coins showing Tail. So the Answer is 33. (Please see the final state of the coins in the 1st game)

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Sridhar Likes Travelling

## Sridhar Likes Travelling Problem Code: TOURMAP

---

Sridhar was a seasoned traveler. He liked to visit new places. More than all he was a meticulous planner. This time he was planning to visit Europe. He wrote down his travel itinerary like as follows:

If he wanted to visit Madrid, Paris, Munich, Warsaw and Kiev in this order, he would write it down like as:

```

Madrid Paris 100ParisMunich200ParisMunich200
Munich Warsaw 150WarsawKiev120WarsawKiev120

```

More formally, if he wanted to go from **A** to **B** directly and the price is **C** dollars, then he would write

A B C\$

on a card. Each move was written on a different card. Sridhar was a great planner, so he would never visit the same place twice. Just before starting his journey, the cards got shuffled. Help Sridhar figure out the actual order of the cards and the total cost of his journey.

---

## Input

The first line of the input contains an integer **T**, the number of test cases. **T** test cases follow. Each case contains an integer **N**, the number of cities Sridhar is planning to visit. **N-1** lines follow. Each line is of the form

$A_i \ B_i \ C_i \$$

where the **i-th** line refers to the **i-th** card after getting shuffled.

---

## Output

For each case the output contains **N** lines, the first **N-1** lines should contain the **N-1** cards in their proper original order, the **N-th** line should contain the total cost of the travel. See Example for detailed format.

---

## Constraints

1	$\leq T \leq$	10
1	$\leq N \leq$	5000
1	$\leq \text{length}$	of
1	$\leq \text{length}$	$A_i \leq$
1	$\leq C_i \leq$	50
		50
		1000

$A_i, B_i$  will contain only **lowercase and uppercase latin characters**, no two cities will have **same names**.

The names of cities are case-sensitive. So "warsaw" and "Warsaw" should be considered as different cities.

---

## Example

### Input

1

5

Warsaw Kiev 120MadridParis100MadridParis100

Munich Warsaw 150ParisMunich200ParisMunich200

#### Output

Madrid Paris 100ParisMunich200ParisMunich200

Munich Warsaw 150WarsawKiev120WarsawKiev120

570\$

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Delicious Dishes

## Delicious Dishes Problem Code: DDISH

Chef is the owner of the loveliest restaurant in ChefTown. To make his work easier, all dishes are numbered by distinct positive integers, and you can order with this positive integer in Chef's restaurant. Amazingly Chef has many kinds of recipes, so every positive integer denotes exactly one dish. To taste dish numbered with some integer C, you have to pay exactly C dollars.

Unfortunately not all the dishes are delicious. After years of working in his restaurant, Chef discovered that a dish is considered to be delicious if the number that denotes this dish consists of different digits. Let's call such numbers "delicious numbers". For example numbers 123, 1, 91, 102 are delicious but 122, 11, 900 are not.

Ira is a little girl from the University of Lublin and she's fond of traveling. Tonight she stops in ChefTown and wants to have a meal at Chef's restaurant. On the one hand she has R dollars in her pocket and she is ready to spend it, on the other hand she wants to seem rich, so she won't spend less than L dollars. Of course, since she is a little girl, she wants to order just one dish. It is difficult to make a choice for Ira because of variety of dishes. Now she asks you to find out the number of delicious dishes Ira may order, considering her conditions.

---

#### Input

First line contains single integer T ( $1 \leq T \leq 200000$ ) – the number of test cases. Each test case consists of two positive integers in a line - L and R ( $1 \leq L \leq R \leq 10^{18}$ ).

---

#### Output

For every test case, output the answer on a separate line.

---

#### Example

Input:

2

5 13

1 100

**Output:**

8

90

## Explanation

For the first test Ira can order every dish from 5 to 13 except 11.

[Home](#) » [Compete](#) » [November Challenge 2012](#) » A Wonderful Chocolate

## A Wonderful Chocolate Problem Code: CBARS

A few days ago Chef decided to cook a new dish – chocolate. This must be something amazing. The idea is that chocolate bar will be divided into cells. It must be long, but narrow. To interest customers every bar must be unique. Bar will consist of cells of black or white chocolate. In addition every bar must be good looking. It means that the bar must not contain any totally white or totally black rectangle, whose width **and** length are more than 1 (Note that a bar is good if **(width > 1 and length = 1)** or **(length > 1 and width = 1)**). Now, Chef wants to know how many bars can he cook? He's not good in computer programming, so this task is for you. By the way, it's not permitted to rotate bars. It means that WBB and BBW are different bars.

## Input

Input contains two integers: width  $a$  ( $1 \leq a \leq 6$ ) and length  $b$  ( $1 \leq b < 2^{63}$ ).

## Output

Print in output a single integer which is the answer. Answer can be a very big number, so print it modulo  $10^9+7$  (1000000007).

## Example

**Input:**

2 2

Output:

14

Input:

3 3

Output:

322

---

## Explanation

In the first sample, there are  $2^{(2*2)} = 16$  ways coloring the chocolate in total, and the only following 2 chocolates are not good

WW

WW

The bar contains a totally white rectangle of length = 2 and width = 2.

BB

BB

The bar contains a totally black rectangle of length = 2 and width = 2.

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Candy Collecting Game

## Candy Collecting Game Problem Code: CANDYGAM

Alice and Bob have come to spend their day in a FunFair which is close to their house. They have played almost all games and seen all the attractions. But there's one game left which has caught their attention. The winner of this game could get a LOT of candies! The game is as follows:

- The game is a two-player game which is played on an **NxM** grid.

- Each cell of the grid has a certain number of candies.  $A[i][j]$  denotes the number of candies in the  $j^{\text{th}}$  column of the  $i^{\text{th}}$  row.
- The players make alternate turns.
- In each turn, a player *must* remove one *complete row* or one *complete column* from the grid. He will add all the candies in that row/column into his account.
- The only possible rows that a player can remove in his/her turn are the top row or the bottom row. Similarly, only leftmost column or the rightmost column can be removed in one turn.
- After removing the row or column, the game is played on the new reduced grid with one less row or column respectively.
- When the entire matrix is emptied, the player with higher number of candies wins the game and can take all those candies with him/her. The loser has to give back the candies collected so far and return empty-handed.
- If both players have equal number of candies, both of them are declared winners.

Alice and Bob want to take as many candies home as possible. They don't have much time with themselves. So they come up with a strategy using which they want to increase Bob's share of candies and decrease Alice's share. This way, they plan to make Bob win and take his entire share of candies! The strategy is as follows:

- Alice decides to pick the row or column from the present grid which has the *least* number of candies in every turn of hers.
- If there are more than one such options, then her order of preference will be:
  - 1) first row
  - 2) last row
  - 3) first column
  - 4) last column.
 For example, if all these 4 rows/columns have the same number of candies, she will remove the first row in this turn.

- Bob chooses an optimal strategy through which he maximizes his share of candies in the end of the game. Note that Bob's strategy maximizes the number of his candies, not winner's candies. Also note that Bob knows Alice's strategy, of course, and he will take into account her strategy when he decides his move.

It's quite evident that their strategy is not optimal. It might happen that Bob loses a game. And sometimes, both might win (with equal number of candies). Given the grid and the number of candies in each cell, your task is to tell them how many candies will the winner get if they play with this combined strategy. Alice will always start the game (Ladies first!).

---

### **Input:**

First line of the test data contains a single integer **T**, the number of test cases. Each test case starts with two space separated integers **N** and **M**, the number of rows and columns of the grid respectively. Then follow **N** lines containing **M** space separated integers **A[i][j]**. These lines describe the grid.

---

### **Output:**

For each test case, output a single line containing the number of candies the winner(s) gets.

---

### **Constraints:**

$1 \leq T \leq 25$

$1 \leq N, M \leq 50$

$0 \leq A[i][j] \leq 1000000000$

---

### **Example:**

#### **Input:**

3

3 3

0 9 9

0 6 6

0 9 6

2 2

1 1

1 1

1 4

1 2 3 4

### Output:

39

4

9

### Explanation:

1<sup>st</sup> case: Alice picks the first column (with all 0's). We're left with a 3x2 grid now. Bob picks the first column (9+6+9=24). We're left with a 3x1 grid. Alice picks the last row (6). A 2x1 grid is left. Bob picks the entire column (9+6=15). Thus, Bob has 24+15=39 candies and Alice has 6. So clearly, Bob is the winner.

2<sup>nd</sup> case: Here, both end up with two candies each. Hence both are the winners. Thus the winner(s) have 2+2=4 candies in all.

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Lucky Balance

## Lucky Balance Problem Code: LUCKY9

---

Chef has the string **s** of length **n** consisted of digits **4** and **7**. The string **s** is called balanced if there exists such integer **x** ( $1 \leq x \leq n$ ) that the number of digits **4** in substring **s[1; x]** is equal to the number of digits **7** in substring **s(x; n]**, where **s[1; x]** is the substring from the **1<sup>st</sup>** digit to **(x-1)**<sup>th</sup> digit of **s**, and **s(x; n]** is the substring from the **(x+1)**<sup>th</sup> digit to **n<sup>th</sup>** digit of **s**. For example, **s = 747474** is a balanced string, because **s[1; 4] = 747** has one **4** and **s(4; 6] = 74** has one **7**. Note that **x** can be **1** or **n** and **s[1; 1]** and **s(n; n]** denote an empty string.

In one turn Chef can choose any pair of consecutive digits and swap them. Find for Chef the total number of different balanced string that can be obtained from string **s** using any (even 0) number of turns. Print the result modulo **1000000007**.

---

## Input

The first line of the input contains one integer **T**, the number of test cases. Then **T** lines follow, each of which contains string **s** for the corresponding test.

---

## Output

**T** lines, each of which contains single integer - the answer for the corresponding test modulo **10<sup>9</sup>+7**.

---

## Constraints

$1 \leq T \leq 10$

$1 \leq n \leq 5000$

---

## Example

**Input:**

2

47

4477

**Output:**

1

4

## Jam Board Problem Code: JABO

Jimmy is a very studious and obedient boy. He has recently joined Digital Circuits Course in his school. He is absolutely fascinated by digital circuits, and stays back in lab after school hours to make circuits. What interests him is plugging wires and voltage sources on Jam Board and then lighting up an LED.

A Jam Board is a large plastic board, with lots of holes in it. The holes are arranged as following-

- Holes in a group of 5 share a common metallic plate, i.e if one of them gets a high voltage, then all 5 of them are at "High". The group is aligned vertically.
- There are several groups placed horizontally as well as vertically, depending on the dimensions of the Jam Board.

For illustration, consider a Jam Board Having 6 Columns and 2 Rows -

A B C D E F

A B C D E F

A B C D E F

A B **1** D E F

A B C D **2** F

G H I J K L

G H I J K L

G H I J K L

G **3** I J K L

G H I J K L

All A's are in a group, and share same voltage. Notice the **3** above, it is basically in H group. How can we refer to the point **3** on Jam Board, using coordinate system? Candy for taking a guess! It is (2, 9), from Top Left.

Now if Jimmy plugs one end of copper wire in **1** and the other in **2**, then C and E will become a single group and start sharing voltage. Jimmy can apply a high voltage to any hole, and that group becomes "high". The Jam Board is grounded, thus each group is initially at low voltage. When an LED is connected across two points, it glows up if there is a potential difference between the two points, i.e. when one end is "high" and the other is "low".

The lab assistant must leave, but he allows Jimmy to stay with one Jam Board, one LED, one Voltage sources (with several output points) and lots of wires. So Jimmy gets down to work, he connects wires, applies (high) voltage, and occasionally connects the LED between two points. Sometimes he removes a voltage source as well. At times he connects a wire(or voltage source) to the point, which already has a wire(or voltage source) connected to it. This is not a good practice, but Jimmy is excused because he is new to electrical stuff. And when he removes voltage source from a point, he removes only one of it at a time.

Since the number of connections is rather enormous and confusing, he requires your help to tell him whether the LED will light up each time he connects it.

However Jimmy's cryptography classes have made him rather irksome. He is telling you the row or column not as a number, but in a Base-52 notation of his own. In this notation, each number is represented as 2 characters, where each character can be either A-Z or a-z. Here A = 0, B = 1, ... Z = 25, a = 26, ... z = 51. So 1 is represented as AB, 5 is AF, 97 is Bt. And to top that he doesn't give any gap between row number and column number, or between pair of coordinates (column and row), used to indicate the endpoints of wire.

## Input

The first line contains 3 space separated integers, **N** ( $\leq 1000000$ ), **R** ( $\leq 500$ ), **C** ( $\leq 2500$ ). R is the number of rows and C is the number of columns in the Jam Board. Then N lines follow. Each line starts with one of the following characters-

- W - It means a wire is attached in the Board. W is followed by the pair of coordinates, for the two endpoints of wire.
- V - It means a voltage is applied on the board. V is followed by the coordinate where the Voltage is applied.
- R - It means a voltage source is removed from the board. R is followed by the coordinate from where the source is removed.
- L - It means an LED is attached on the Board. L is followed by the pair of coordinates, for the two endpoints of LED.

Note that there is no space between coordinates, and between column and row, for any type of query. See example for clarity.

## Output

For each input line starting with 'L' output "ON" if the LED is on, and "OFF" otherwise, on separate line.

## Example

Input:

9 2 10

WADAEAFAG

VAGAD

LAFAHAGAE

VAKAK

LAJAKAKAJ

LAKAIAGAB

LABABACAB

RAKAK

LAJAKAKAJ

**Output:**

ON

ON

OFF

OFF

OFF

---

## Explanation

The input is to be interpreted as

9 2 10

W 3 4 5 6

V 6 3

L 5 7 6 4

V 10 10

L 9 10 10 9

L 10 8 6 1

L 1 1 2 1

R 10 10

L 9 10 10 9

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Many Left

## Many Left Problem Code: MANYLEFT

The classical game of OneLeft is played as follows. Some pegs are placed on an  $N \times N$  grid. Initially, at least one cell is empty and at least one contains a peg (each cell contains at most one peg). A move consists of jumping one peg over an adjacent peg to an empty cell, and removing the peg that was jumped over. Formally, if there is a peg in cell  $(x_1, y_1)$ , and cell  $(x_2, y_2)$  is empty, and  $(x_1 - x_2, y_1 - y_2)$  is one of  $(0, 2)$ ,  $(0, -2)$ ,  $(2, 0)$ , or  $(-2, 0)$ , and there is a peg in cell  $((x_1 + x_2)/2, (y_1 + y_2)/2)$ , then the peg in cell  $(x_1, y_1)$  may be moved to cell  $(x_2, y_2)$  and the peg in cell  $((x_1 + x_2)/2, (y_1 + y_2)/2)$  removed. The coordinate  $(0, 0)$  indicates the top-left corner,  $(N-1, 0)$  indicates the top-right corner,  $(0, N-1)$  indicates the bottom-left corner, and  $(N-1, N-1)$  indicates the bottom-right corner.

The game continues until no more moves are possible. Normally the goal of OneLeft is to leave a single peg on the grid. However, in this problem the goal is to leave as many pegs as possible. Optimal solutions are not required, but solutions that leave more pegs will score more points.

---

### Input

Input begins with an integer  $N$ , the size of the grid.  $N$  lines follow with  $N$  characters each, representing the grid. A '.' character indicates an empty cell, and a '\*' character indicates a peg.

---

### Output

For each test case, first output the number of moves in your solution. Then output each move in the form " $x_1 \ y_1 \ x_2 \ y_2$ ", which indicates a peg moving from  $(x_1, y_1)$  to  $(x_2, y_2)$ . Any whitespace in your solution will be ignored.

---

### Scoring

Your score for each test case is the fraction of cells containing pegs after performing the moves in your solution. Your overall score is the average of your scores on the individual test cases. Invalid solutions will be judged as "wrong answer". In particular, if any legal moves exist after the moves in your solution have been performed, your solution will be considered invalid.

---

### Sample Input 1

```
6
..*...
*..*.*
***.**
.****..
****..
**.*.*
```

---

### Sample Output 1

```
13
1 3 1 1
3 3 1 3
0 5 0 3
0 2 0 0
3 5 3 3
5 2 3 2
5 0 5 2
3 2 3 0
2 4 0 4
0 3 0 5
```

3 0 1 0

0 0 2 0

0 5 2 5

---

### Sample Input 2

5

. \* . \* .

. . \* . .

\* . . . \*

. . . \* .

. \* . . .

---

### Sample Output 2

0

The first sample output scores  $8/36 = 0.2222$ . The second sample output scores  $7/25 = 0.28$ . Recall that the goal is to maximize your score.

---

### Test Case Generation

For each official test file,  $N$  is chosen randomly and uniformly between 10 and 30, inclusive. A real number  $D$  is chosen randomly and uniformly between 0.5 and 0.95, then each cell is independently chosen to contain a peg with probability  $D$ .

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Arithmetic Progressions

## Arithmetic Progressions Problem Code: COUNTARI

Given  $N$  integers  $A_1, A_2, \dots, A_N$ , Dexter wants to know how many ways he can choose three numbers such that they are three consecutive terms of an arithmetic progression.

Meaning that, how many triplets  $(i, j, k)$  are there such that  $1 \leq i < j < k \leq N$  and  $A_j - A_i = A_k - A_j$ .

So the triplets  $(2, 5, 8)$ ,  $(10, 8, 6)$ ,  $(3, 3, 3)$  are valid as they are three consecutive terms of an arithmetic progression. But the triplets  $(2, 5, 7)$ ,  $(10, 6, 8)$  are not.

---

## Input

First line of the input contains an integer  $N$  ( $3 \leq N \leq 100000$ ). Then the following line contains  $N$  space separated integers  $A_1, A_2, \dots, A_N$  and they have values between  $1$  and  $30000$  (inclusive).

---

## Output

Output the number of ways to choose a triplet such that they are three consecutive terms of an arithmetic progression.

---

## Example

Input:

10

3 5 3 6 3 4 10 4 5 2

Output:

9

---

## Explanation

The followings are all 9 ways to choose a triplet

1 :  $(i, j, k) = (1, 3, 5)$ ,  $(A_i, A_j, A_k) = (3, 3, 3)$

2 :  $(i, j, k) = (1, 6, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

3 :  $(i, j, k) = (1, 8, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

4 :  $(i, j, k) = (3, 6, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

5 :  $(i, j, k) = (3, 8, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

6 :  $(i, j, k) = (4, 6, 10)$ ,  $(A_i, A_j, A_k) = (6, 4, 2)$

7 :  $(i, j, k) = (4, 8, 10)$ ,  $(A_i, A_j, A_k) = (6, 4, 2)$

8 :  $(i, j, k) = (5, 6, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

9 :  $(i, j, k) = (5, 8, 9)$ ,  $(A_i, A_j, A_k) = (3, 4, 5)$

[Home](#) » [Compete](#) » [November Challenge 2012](#) » Martial Arts

## Martial Arts Problem Code: MARTARTS

---

Chef somehow strayed away from kitchen and got involved in martial arts. He isn't in the right shape to get into fights but he is qualified enough to be a coach in the local gym. He's just been put in charge of organizing an upcoming tournament.

There will be two teams of  $N$  fighters participating in the tournament. We will refer to the Chef's team as home team. The other one will be guest team. As the organizer, Chef has to assign all fighters to  $N$  fights. One contestant from each team will take part in a fight and the judges will award certain number of points to each fighter's team according to their performance. Team's final score is equivalent to the sum of points which the contestants earned in their fights. Chef has to assign every fighter to exactly one fight.

Chef knows all participants very well and wants to assign pairs in such way that will maximize the difference between home and guest team's score. Let's say that home team will score  $H$  and guest team  $G$  points. Chef wants to maximize  $H-G$ . If he can do that in multiple ways, he would then like to maximize  $H$ .

However, he is not the only one with a hidden agenda. After the pairs are announced, guest coach can make up an excuse that one of his contestants got injured. This means that this contestant's fight won't take place and therefore won't contribute to team scores. When deciding whether to cancel some fight and which fight should that be, the guest coach has a similar goal as Chef. His primary goal is to maximize  $G-H$  and secondary to maximize  $G$ .

Chef is stunned by guest coach's tactic and asks for your help to organize fights optimally.

---

### Input

The first line contains the number of contestants  $N$  in each team. The following  $N$  lines describe all possible pairings of contestants.  $j$ -th element in  $i$ -th line is of the form " $A_{i,j}:B_{i,j}$ " which means that home team would get  $A_{i,j}$  points and guest team  $B_{i,j}$  points if  $i$ -th contestant from home team fights against the  $j$ -th from guest team.

---

### Output

Output the number of points H scored by home team and the number of points G scored by guest team in an optimally conducted tournament. Separate them by a single space.

---

## Constraints

- $1 \leq N \leq 100$
  - $0 \leq A_{i,j}, B_{i,j} < 10^{12}$
- 

## Example

Input:

3

10:7 0:20 6:5

5:5 0:10 8:10

0:0 50:0 100:0

Output:

18 17

---

## Explanation

Chef will pair contestants in the following way: (1,1), (2,3), (3,2). Coach of the guest team will cancel the last fight (3,2) which would earn home team 50 points. This way the final score is 18 points for home team and 17 points for guest team.

COOK28

[November Cook-Off  
2012](#)

18 Nov 2012  
21:30:00

2 hours 30 minutes 736

[Home](#) » [Compete](#) » [November Cook-Off 2012](#) » Little Elephant and Permutations

## Little Elephant and Permutations Problem Code: LEPERMUT

The Little Elephant likes permutations. This time he has a permutation  $A[1], A[2], \dots, A[N]$  of numbers  $1, 2, \dots, N$ .

He calls a permutation  $A$  good, if the number of its inversions is equal to the number of its local inversions. The number of inversions is equal to the number of pairs of integers  $(i, j)$

such that  $1 \leq i < j \leq N$  and  $A[i] > A[j]$ , and the number of local inversions is the number of integers  $i$  such that  $1 \leq i < N$  and  $A[i] > A[i+1]$ .

The Little Elephant has several such permutations. Help him to find for each permutation whether it is good or not. Print **YES** for a corresponding test case if it is good and **NO** otherwise.

---

## Input

The first line of the input contains a single integer  $T$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains a single integer  $N$ , the size of a permutation. The next line contains  $N$  space separated integers  $A[1], A[2], \dots, A[N]$ .

---

## Output

For each test case output a single line containing the answer for the corresponding test case. It should be **YES** if the corresponding permutation is good and **NO** otherwise.

---

## Constraints

$1 \leq T \leq 474$

$1 \leq N \leq 100$

It is guaranteed that the sequence  $A[1], A[2], \dots, A[N]$  is a permutation of numbers  $1, 2, \dots, N$ .

---

## Example

Input:

4

1

1

2

2 1

3

3 2 1

4

1 3 2 4

**Output:**

YES

YES

NO

YES

## Explanation

**Case 1.** Here  $N = 1$ , so we have no pairs  $(i; j)$  with  $1 \leq i < j \leq N$ . So the number of inversions is equal to zero. The number of local inversion is also equal to zero. Hence this permutation is good.

**Case 2.** Here  $N = 2$ , and we have one pair  $(i; j)$  with  $1 \leq i < j \leq N$ , the pair  $(1; 2)$ . Since  $A[1] = 2$  and  $A[2] = 1$  then  $A[1] > A[2]$  and the number of inversions is equal to 1. The number of local inversion is also equal to 1 since we have one value of  $i$  for which  $1 \leq i < N$  (the value  $i = 1$ ) and  $A[i] > A[i+1]$  for this value of  $i$  since  $A[1] > A[2]$ . Hence this permutation is also good.

**Case 3.** Here  $N = 3$ , and we have three pairs  $(i; j)$  with  $1 \leq i < j \leq N$ . We have  $A[1] = 3$ ,  $A[2] = 2$ ,  $A[3] = 1$ . Hence  $A[1] > A[2]$ ,  $A[1] > A[3]$  and  $A[2] > A[3]$ . So the number of inversions is equal to 3. To count the number of local inversion we should examine inequalities  $A[1] > A[2]$  and  $A[2] > A[3]$ . They both are satisfied in our case, so we have 2 local inversions. Since  $2 \neq 3$  this permutations is not good.

**Case 4.** Here we have only one inversion and it comes from the pair  $(2; 3)$  since  $A[2] = 3 > 2 = A[3]$ . This pair gives also the only local inversion in this permutation. Hence the number of inversions equals to the number of local inversions and equals to one. So this permutation is good.

[Home](#) » [Compete](#) » [November Cook-Off 2012](#) » Little Elephant and Divisors

## Little Elephant and Divisors Problem Code: LEDIV

The Little Elephant from the Zoo of Lviv has an array  $A$  that consists of  $N$  positive integers. Let  $A[i]$  be the  $i$ -th number in this array ( $i = 1, 2, \dots, N$ ).

Find the minimal number  $x > 1$  such that  $x$  is a divisor of all integers from array  $A$ . More formally, this  $x$  should satisfy the following relations:

$A[1] \bmod x = 0, A[2] \bmod x = 0, \dots, A[N] \bmod x = 0$ ,

where **mod** stands for the modulo operation. For example, **8 mod 3 = 2**, **2 mod 2 = 0**, **100 mod 5 = 0** and so on. If such number does not exist, output **-1**.

---

## Input

The first line of the input contains a single integer  $T$ , the number of test cases.  $T$  test cases follow. The first line of each test case contains a single integer  $N$ , the size of the array  $A$  for the corresponding test case. The second line contains  $N$  space separated integers  $A[1], A[2], \dots, A[N]$ .

---

## Output

For each test case output a single line containing the answer for the corresponding test case.

---

## Constraints

$1 \leq T \leq 100000$

$1 \leq N \leq 100000$

The sum of values of  $N$  in each test file does not exceed **100000**

$1 \leq A[i] \leq 100000$

---

## Example

Input:

2

3

2 4 8

3

4 7 5

Output:

2

-1

## Explanation

**Case 1.** Clearly 2 is a divisor of each of the numbers 2, 4 and 8. Since 2 is the least number greater than 1 then it is the answer.

**Case 2.** Let's perform check for several first values of  $x$ .

$x$	$4 \bmod x$	$7 \bmod x$	$5 \bmod x$
2	0	1	1
3	1	1	2
4	0	3	1
5	4	2	0
6	4	1	5
7	4	0	5
8	4	7	5
9	4	7	5

As we see each number up to 9 does not divide all of the numbers in the array. Clearly all larger numbers also will fail to do this. So there is no such number  $x > 1$  and the answer is -1.

[Home](#) » [Compete](#) » [November Cook-Off 2012](#) » Little Elephant and Magic

## Little Elephant and Magic Problem Code: LEMAGIC

The Little Elephant from the Zoo of Lviv believes in magic.

He has a magic board  $A$  that consists of  $N$  rows and  $M$  columns. Each cell of the board contains an integer from 0 to 9 inclusive. The cell at the intersection of the  $i$ -th row and the  $j$ -th column is denoted as  $(i; j)$ , where  $1 \leq i \leq N$  and  $1 \leq j \leq M$ . The number in the cell  $(i; j)$  is denoted as  $A[i][j]$ .

The Little Elephant owns the only magic operation which can be described as follows. He chooses some integer  $P$  and some row or column, after that for each cell in the chosen row or column he adds number  $P$  to the number in this cell and take the result modulo 10 in order to keep numbers in the range  $\{0, 1, \dots, 9\}$ . Our Little Magician wants to perform series of such operations to achieve some board for which certain characteristic called *level of the board* is maximal possible.

So what is the level of the board? Bluntly speaking it is the length of the longest non-increasing subsequence of cells of the board. Formally, the level of the board is the maximal integer  $K$  such that there exists such sequence of **different** cells  $(i_1; j_1), (i_2; j_2), \dots, (i_K; j_K)$  for which

$$\begin{aligned} 1 \leq i_1 \leq \dots \leq i_K \leq N, \\ 1 \leq j_1 \leq \dots \leq j_K \leq M, \end{aligned}$$

and

$$A[i_1][j_1] \geq A[i_2][j_2] \geq \dots \geq A[i_K][j_K].$$

Though, the magic operation, the Little Elephant owns, is quite powerful, there are some restrictions dictated by the Association of Cursed Magicians (ACM):

- The number  $P$  should be chosen in advance and should be the same for all operations.
- For each row the magic operation can be applied at most once.
- The same, of course, is true for columns.

Without these stupid restrictions of this stupid Association our hero could always achieve the maximal possible level  $M + N - 1$ . But now he is confused and asks you for help. Find the maximal level of the board  $A$  after making arbitrary number of magic operations according to the restrictions of ACM.

## Input

The first line of the input contains a single integer  $T$ , the number of test cases. Then  $T$  test cases follow. The first line of each test case contains two space separated integers  $N$  and  $M$ , the sizes of the board. Each of the following  $N$  lines contains  $M$  one-digit numbers without spaces. The  $i$ -th line among them contains the numbers  $A[i][1], \dots, A[i][M]$ .

## Output

For each test case output a single line containing a single integer, the maximal possible level of the board that Little Elephant can achieve under the restrictions of ACM.

## Constraints

**$1 \leq T \leq 10$**   
 **$1 \leq N \leq 100$**   
 **$1 \leq M \leq 100$**   
 **$0 \leq A[i][j] \leq 9$**

---

## Example

**Input:**

2

2 2

11

10

3 4

3478

4268

7173

**Output:**

3

5

---

## Explanation

**Case 1.** The board already has a sequence of **3** cells that satisfies all required constraints (without applying any operation). For example, one can choose, the sequence **(1; 1), (1; 2), (2; 2)**. It is also shown in the figure below (chosen cells are made bold):

11

10

Let's formally validate this sequence of cells. Inequality  $1 \leq i_1 \leq \dots \leq i_k \leq N$  takes the form  **$1 \leq 1 \leq 2$** . Inequality  $1 \leq j_1 \leq \dots \leq j_k \leq M$  takes the form  **$1 \leq 2 \leq 2$** . Finally,

inequality  $A[i_1][j_1] \geq A[i_2][j_2] \geq \dots \geq A[i_k][j_k]$  takes the form  $1 \geq 1 \geq 0$ . So all of them is satisfied, which means that the level of this board is at least 3. But clearly, we can't have the required sequence of cells of length more than 3. So 3 is the actual level of this board.

**Case 2.** The desired sequence of length 5 can be achieved by several values of  $P$ . Consider, for example,  $P = 3$ . At first let's apply the magic operation to the 1st row. We get the following transformation:

3478  $\rightarrow$  6701

4268  $\rightarrow$  4268

7173  $\rightarrow$  7173

Now let's transform the 1st column by the magic operation. We get:

6701  $\rightarrow$  9701

4268  $\rightarrow$  7268

7173  $\rightarrow$  0173

Finally we modify the 2nd column:

9701  $\rightarrow$  9001

7268  $\rightarrow$  7568

0173  $\rightarrow$  0473

Now we can take the following sequence of 5 cells to satisfy all needed constraints: (1; 1), (2; 1), (2; 2), (3; 2), (3; 4) (see the figure below):

9001

7568

0473

Just to reiterate we note that the inequality  $A[i_1][j_1] \geq A[i_2][j_2] \geq \dots \geq A[i_k][j_k]$  takes the form  $9 \geq 7 \geq 5 \geq 4 \geq 3$  for this sequence. One can check (for example, by brute force), that sequences of length more than 5 can't be achieved. So 5 is the answer.

# Little Elephant and Lucky Segment Problem

Code: LELUCKYN

The Little Elephant from the Zoo of Lviv likes lucky digits. Everybody knows that the lucky digits are digits **4** and **7**.

This time he has an array **A** that consists of **N** integers: **A[1], A[2], ..., A[N]**. Let **F4(x)** be the number of digits **4** in the decimal representation of **x**, and **F7(x)** be the number of digits **7** in the decimal representation of **x**. For example, **F4(5) = 0**, **F4(4467) = 2** and **F7(457747) = 3**.

Consider some pair of integers **L** and **R** such that  $1 \leq L \leq R \leq N$ . Let **C4** be the total number of digits **4** in decimal representation of integers **A[L], A[L + 1] ..., A[R]**, i. e.,

$$C4 = F4(A[L]) + F4(A[L + 1]) + \dots + F4(A[R]).$$

Similarly, let **C7** be the the total number of digits **7** in decimal representation of integers **A[L], A[L + 1] ..., A[R]**, i. e.,

$$C7 = F7(A[L]) + F7(A[L + 1]) + \dots + F7(A[R]).$$

The Little Elephant wants to know the number of such pairs **(L; R)** for which  $C4^{C7} \leq R - L + 1$ . But he believes that the number **2** is unlucky. Hence he discards all pairs where **C4 = 2** or **C7 = 2**.

Help the Little Elephant to find the answer for the problem.

**Remark.**  $0^0 = 1$ . It is a standard mathematical definition.

---

## Input

The first line of the input contains a single integer **T**, the number of test cases. Then **T** test cases follow. The first line of each test case contains a single integer **N**, the size of the array **A** for the corresponding test case. The second line contains **N** space separated integers **A[1], A[2], ..., A[N]**.

---

## Output

For each test case output a single line containing the answer for the corresponding test case.

---

## Constraints

$1 \leq T \leq 3$   
 $1 \leq N \leq 10^5$   
 $1 \leq A[i] \leq 10^9$

---

## Example

Input:

3  
3  
1 2 1  
4  
472477548  
1  
777

Output:

6  
5  
1

---

## Explanation

**Case 1.** Here  $C4 = C7 = 0$  for all pairs  $(L; R)$ . So  $0^0 = 1 \leq R - L + 1$  for each pair. There are 6 pairs  $(L; R)$  in all:  $(1; 1), (1; 2), (1; 3), (2; 2), (2; 3), (3; 3)$ . So the answer is 6.

**Case 2.** Here we have 10 pairs of  $(L; R)$  in all. Consider them all.

- $L = 1, R = 1: C4 = 1, C7 = 1, C4^{C7} = 1^1 = 1 \leq 1 = R - L + 1$ . The pair is counted.
- $L = 1, R = 2: C4 = 1, C7 = 1, C4^{C7} = 1^1 = 1 \leq 2 = R - L + 1$ . The pair is counted.
- $L = 1, R = 3: C4 = 2, C7 = 1, C4^{C7} = 2^1 = 2 \leq 3 = R - L + 1$ . But the pair is not counted since  $C4 = 2$ .

- $L = 1, R = 4: C4 = 3, C7 = 3, C4^{C7} = 3^3 = 27 > 4 = R - L + 1$ . The pair is not counted.
- $L = 2, R = 2: C4 = 0, C7 = 0, C4^{C7} = 0^0 = 1 \leq 1 = R - L + 1$ . The pair is counted.
- $L = 2, R = 3: C4 = 1, C7 = 0, C4^{C7} = 1^0 = 1 \leq 2 = R - L + 1$ . The pair is counted.
- $L = 2, R = 4: C4 = 2, C7 = 2, C4^{C7} = 2^2 = 4 > 3 = R - L + 1$ . The pair is not counted. It is also not counted since  $C4 = 2$ , and also since  $C7 = 2$ .
- $L = 3, R = 3: C4 = 1, C7 = 0, C4^{C7} = 1^0 = 1 \leq 1 = R - L + 1$ . The pair is counted.
- $L = 3, R = 4: C4 = 2, C7 = 2, C4^{C7} = 2^2 = 4 > 2 = R - L + 1$ . The pair is not counted. It is also not counted since  $C4 = 2$ , and also since  $C7 = 2$ .
- $L = 4, R = 4: C4 = 1, C7 = 2, C4^{C7} = 1^2 = 1 \leq 1 = R - L + 1$ . But the pair is not counted since  $C7 = 2$ .

As we see there are exactly 5 pairs  $(L; R)$  that satisfy required constraints.

**Case 3.** Here we have the only pair  $(L; R) = (1; 1)$  for which  $C4 = 0, C7 = 3$  and  $C4^{C7} = 0^3 = 0 \leq 1 = R - L + 1$ . So it is counted, and the answer is 1.

[Home](#) » [Compete](#) » [November Cook-Off 2012](#) » Little Elephant and Sheet

## Little Elephant and Sheet Problem Code: LEPAPER

The Little Elephant from the Zoo of Lviv wants to paint the sheet of paper. The sheet has the size  $1 \times Z$  and consists of  $Z$  consecutive cells numbered from 1 to  $Z$  inclusive (we use letter  $Z$  because  $N$  will denote another value below and the words "size" and "zoo" both contain letter 'Z' :))

There are also  $C$  available colors numbered from 1 to  $C$  inclusive. Current color of the  $i$ -th cell is  $Col[i]$  ( $1 \leq Col[i] \leq C$ ).

The Little Elephant can repaint some cells of the sheet. In a single move he chooses arbitrary pair of integers  $L$  and  $R$  such that  $1 \leq L \leq R \leq Z$  and some color  $Ci$  from 1 to  $C$  inclusive, after that he repaints all the cells on the segment  $[L, R]$  to the color  $Ci$ , that is, he assigns  $Col[j] = Ci$  for  $j$  from  $L$  to  $R$  inclusive. **IMPORTANT: it is allowed to repaint each cell at most once during the whole sequence of moves.**

Help the Little Elephant to find the total number of different sheets he can get in no more than  $K$  moves. Since the answer can be large, print it modulo  $1000000007$  ( $10^9 + 7$ ).

Two sheets are considered different if there exists at least one index  $i$  from 1 to  $Z$  inclusive, such that that  $A[i]$  is not equal to  $B[i]$ , where  $A[i]$  is the color of the  $i$ -th cell of the first sheet and  $B[i]$  is the color of the  $i$ -th cell of the second sheet.

Little Elephant, in fact, has some enormous sheet of paper, possibly having billions of cells. But it has very specific structure. Namely, it consists of several blocks of consecutive cells, where all cells in each particular block have the same color (possibly different for different blocks), and the number of these blocks is relatively small. Consider the following example. Let our sheet of paper be  $\{1, 1, 1, 2, 4, 4, 4, 1, 1\}$  (numbers represent colors of the cells). Then it consists of 4 blocks, where the first block has 3 cells of color 1, the second block has 1 cell of color 2, the third block has 3 cells of color 4 and the last fourth block has 2 cells

of color **1**. It seems natural for consecutive blocks to be of different color but we allow opposite situation in the input too. So please DO NOT assume that two consecutive blocks in the input are always of different color.

---

## Input

The first line of the input contains a single integer **T**, the number of test cases. Then **T** test cases follow. The first line of each test case contains three space separated integers **N**, **C** and **K**. Here **N** is the number of blocks in the sheet, **C** is the number of available colors and **K** is the upper bound on the number of moves Little Elephant can make. Each of following **N** lines contains two space separated integers **S[i]** and **M[i]**, where **S[i]** is the color of the **i**-th block and **M[i]** is the number of cells in the **i**-th block. So the above value **Z** equals to **M[1] + M[2] + ... + M[N]**.

---

## Output

For each test case output a single line containing the number of different sheets the Little Elephant can get using at most **K** moves. As was mentioned above you should output this number modulo **10<sup>9</sup> + 7**.

---

## Constraints

$1 \leq T \leq 7$   
 $1 \leq N \leq 7$   
 $1 \leq C \leq 10^9$   
 $1 \leq K \leq 7$   
 $1 \leq S[i] \leq C$   
 $1 \leq M[i] \leq 10^9$

---

## Example

Input:

3

1 4 7 1

1 1

2 2 1

1 1

2 1

2 3 2

1 2

2 2

**Output:**

47

3

57

## Explanation

**Case 1.** Here the sheet consists of one cell of color 1. We have **47** available colors. Using one move we can color this cell in any of available colors. So we can get **47** different sheets.

**Case 2.** Here the sheet of paper is **{1, 2}** (numbers represent colors of the cells). We have **2** available colors. So there exist **4** different sheets of **2** cells. Using at most one move we can achieve every such sheet except **{2, 1}**. Indeed,

- Sheet **{1, 1}** is achieved by repainting the second cell at color **1**.
- Sheet **{1, 2}** is achieved using no repainting.
- Sheet **{2, 2}** is achieved by repainting the first cell at color **2**.

To get **{2, 1}** we need at least two moves. So the answer is **3**.

DEC12

[December Challenge 2012](#)01 Dec 2012  
15:00:00

10 days

2558

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Granama Recipes

## Granama Recipes Problem Code: GRANAMA

Chef has learned a new technique for comparing two recipes. A recipe contains a list of ingredients in increasing order of the times they will be processed. An ingredient is represented by a letter 'a'-'z'. The **i**-th letter in a recipe denotes the **i**-th ingredient. An ingredient can be used multiple times in a recipe.

The technique is as follows. Compare two recipes by comparing their respective lists. If the sets of ingredients used in both recipes are equal and each ingredient is used the same number of times in both of them (processing order does not matter), they are declared as **granama** recipes. ("granama" is the Chef-ian word for "similar".)

Chef took two recipes he invented yesterday. He wanted to compare them using the technique. Unfortunately, Chef forgot to keep track of the number of times each ingredient has been used in a recipe. He only compared the ingredients but NOT their frequencies. More precisely, Chef considers two recipes as granama if there are no ingredients which are used in one recipe and not used in the other recipe.

Your task is to report whether Chef has correctly classified the two recipes (as granama or not granama) although he forgot to keep track of the frequencies.

## Input

The first line of the input contains a single integer **T** denoting the number of test cases. The description for **T** test cases follows. Each test case consists of a single line containing two space-separated strings **R** and **S** denoting the two recipes.

## Output

For each test case, output a single line containing "YES" (quotes for clarity) if Chef correctly classified the two recipes as granama or not granama. Otherwise, output a single line containing "NO" (quotes for clarity) if Chef declared two recipes as granama when they actually are not.

## Constraints

$1 \leq T \leq 100$   
 $1 \leq |R|, |S| \leq 1000$

## Example

### Input:

3

alex axle

paradise diapers

alice bob

### Output:

YES

NO

YES

**Explanation:**

Example case 1: Chef declared them as granama recipes. They are actually granama because the sets of ingredients and the number of times each ingredient has been used are equal. The Chef got it right!

Example case 2: Chef declared them as granama recipes because both sets of ingredients are equal. But they are NOT granama since ingredient 'a' has been used twice in the first recipe but only once in the second. The Chef was incorrect!

Example case 3: Chef declare them as not granama. They are not granama as the sets of ingredients are different. Hence, the Chef was right!

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Magic Rankings

## Magic Rankings Problem Code: MGCRNK

Everybody loves magic, especially magicians who compete for glory on the Byteland Magic Tournament. Magician Cyael is one such magician.

Cyael has been having some issues with her last performances and today she'll have to perform for an audience of some judges, who will change her tournament ranking, possibly increasing it. As she is a great magician she managed to gather a description of the fixed judges' disposition on the room (which is represented as an  $N \times N$  square matrix), such that she knows in advance the fixed points each judge will provide. She also knows that the room is divided into several parallel corridors, such that we will denote the  $j$ -th cell on corridor  $i$ , as  $[i][j]$ . Note that some judges can award Cyael, zero points or negative points, as they are never pleased with her performance.

There is just one judge at each cell of the matrix, except the cells  $[1][1]$  and  $[N][N]$ .

To complete her evaluation, she must start on the top leftmost corner of the room (cell  $[1][1]$ ), and finish on the bottom right corner (cell  $[N][N]$ ), moving either to the cell directly in front of her on the same corridor (that is, moving from cell  $[r][c]$  to cell  $[r][c+1]$ , where  $c+1 \leq N$ ) or to the cell in the next corridor directly in front of where she is (that is, moving from cell  $[r][c]$  to cell  $[r+1][c]$ , where  $r+1 \leq N$ ). She will keep doing this until she reaches the end point of the room, i.e. last cell  $[N][N]$  on the last corridor. Cyael will be judged at all visited cells with a judge.

Cyael wants to maximize her average score at end of her performance. More specifically, if she passes  $K$  judges, each being on cell  $[i_1][j_1]$ , cell  $[i_2][j_2]$ , ..., cell  $[i_K][j_K]$  respectively, then

she wants to maximize  $(S[i_1][j_1] + S[i_2][j_2] + \dots + S[i_K][j_K]) / K$ , where  $S[i][j]$  denotes the points that the judge will give her on the cell  $[i][j]$ .

Help her determine the best path she has to follow in order to maximize her average points.

## Input

The first line contains a single integer  $T$  denoting the number of test cases. The description for  $T$  test cases follows. For each test case, the first line contains a single integer  $N$ . Each of the next  $N$  lines contains  $N$  space-separated integers. The  $j$ -th integer  $S[i][j]$  in  $i$ -th line denotes the points awarded by the judge at cell  $[i][j]$ . Note that the cells  $[1][1]$  and  $[N][N]$  have no judges, so  $S[1][1]$  and  $S[N][N]$  will be 0.

## Output

For each test case, if the maximum possible average points Cyael can obtain is negative, output a single line containing "Bad Judges" (quotes for clarity). Otherwise, output the maximum possible average points. The answer will be considered correct if it has an absolute error no more than  $10^{-6}$ .

## Constraints

$$1 \leq T \leq 20$$

$$2 \leq N \leq 100$$

$$-2500 \leq S[i][j] \leq 2500$$

$$S[1][1] = S[N][N] = 0$$

Your code will be judged against several input files.

## Example

Input:

2

2

0 -4

8 0

2

0 -45

-3 0

**Output:**

8.000000

Bad Judges

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Pizza Delivery

## Pizza Delivery Problem Code: DBOY

**Chef Po** had recently started home delivery service for pizzas. Po has only a single delivery boy that delivers the orders by riding his motorcycle. The motorcycle has an unlimited capacity of fuel tank. However, it is too old and can only ride 1 km for each 1 liter of fuel.

There are  $N$  fuel stations near the restaurant. The  $i$ th fuel station can fill a fuel tank exactly  $K$  litres; not more and not less. Filling a fuel tank with any amount of fuel in those stations tends to take a long time. Since the fuel stations are placed near the restaurant, no fuel is needed to go to a fuel station.

Today Chef Po received  $N$  pizza orders, which is the same number of fuel stations fortuitously. The house of the person that ordered the  $i$ th order is  $H$  km away from the restaurant. The delivery boy cannot deliver more than one order at a time. Therefore, after delivering an order, he must return back to the restaurant to take the next order.

The delivery boy is an efficient person and thus he wants to fill the fuel tank with the exact amount of fuel required to deliver an order and return back. He also does not want to spend much time filling the tank, so he wants to minimize the number of times he fills the tank. Help him determine the minimum number of times he must fill the tank to deliver all orders.

### Input

The first line contains a single integer  $T$  denoting the number test cases. The description of  $T$  test cases follows. For each test case, the first line contains a single integer  $N$ . The second line contains  $N$  space-separated integers  $H$ . The third line contains  $N$  space-separated integers  $K$ .

### Output

For each test case, output a single line containing the minimum number of times the delivery boy must fill the tank.

---

## Constraints

$1 \leq T \leq 500$   $1 \leq T \leq 500$

$1 \leq N \leq 500$   $1 \leq N \leq 500$

$1 \leq H_i \leq 500$   $1 \leq H_i \leq 500$

$1 \leq K_i \leq 500$   $1 \leq K_i \leq 500$

There is at least one way for completing the deliveries.

That is, the delivery boy always can fill a fuel tank exactly  $2 \cdot H_i \cdot K_i$  litres for  $1 \leq i \leq N$ .  $1 \leq i \leq N$ .

---

## Example

Input:

1

4

1 2 3 4

1 4 5 3

Output:

7

---

## Explanation:

Here is one possible solution.

For the first order, the delivery boy must ride  $1 + 1 = 2$  km long. Fill the tank twice in the first fuel station.

For the second order, the delivery boy must ride  $2 + 2 = 4$  km long. Fill the tank once in the second fuel station.

For the third order, the delivery boy must ride  $3 + 3 = 6$  km long. Fill the tank twice in the fourth fuel station.

For the fourth order, the delivery boy must ride  $4 + 4 = 8$  km long. Fill the tank in the third and fourth fuel stations.

In total, the delivery boy must fill the tank 7 times. There is no way to fill the tank less than 7 times.

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Book Exercises

## Book Exercises Problem Code: BEX

Harry is a bright student. To prepare thoroughly for exams, he completes all the exercises in his book! Now that the exams are approaching fast, he is doing book exercises day and night. He writes down and keeps updating the remaining number of exercises on the back cover of each book.

Harry has a lot of books messed on the floor. Therefore, he wants to pile up the books **that still have some remaining exercises** into a single pile. He will grab the books one-by-one and add the books that still have remaining exercises to the top of the pile.

Whenever he wants to do a book exercise, he will pick the book with the minimum number of remaining exercises from the pile. In order to pick the book, he has to remove all the books above it. Therefore, if there are more than one books with the minimum number of remaining exercises, he will take the one which requires the least number of books to remove. The removed books are returned to the messy floor. After he picks the book, he will do all the remaining exercises and trash the book.

Since number of books is rather large, he needs your help to tell him the number of books he must remove, for picking the book with the minimum number of exercises.

Note that more than one book can have the same name.

### Input

The first line contains a single integer **N** denoting the number of actions. Then **N** lines follow. Each line starts with an integer. If the integer is -1, that means Harry wants to do a book exercise. Otherwise, the integer is number of the remaining exercises in the book he grabs next. This is followed by a string denoting the name of the book.

### Output

For each  $-1$  in the input, output a single line containing the number of books Harry must remove, followed by the name of the book that Harry must pick.

---

## Constraints

$1 < N \leq 1,000,000$

$0 \leq (\text{the number of remaining exercises of each book}) < 100,000$

The name of each book consists of between 1 and 15 characters 'a' - 'z'.

Whenever he wants to do a book exercise, there is at least one book in the pile.

---

## Example

### Input:

6

9 english

6 mathematics

8 geography

-1

3 graphics

-1

### Output:

1 mathematics

0 graphics

[Home](#) » [Compete](#) » [December Challenge 2012](#) » The Uncountable Ways

## The Uncountable Ways Problem Code: CNTWAYS

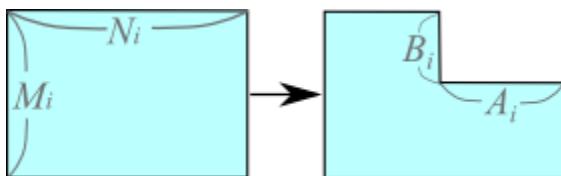
---

Little Chef loves mathematics. Every day, he solves some mathematical problems to improve his skill.

A few days ago, he found a popular problem about turtles. The problem is as follows. Little Chef is given  $R$  rectangles, numbered 1 through  $R$ . The width and height of the  $i$ -th

rectangle are  $N_i$  and  $M_i$  unit respectively. There is a turtle located on the top-left corner of each rectangle. For each rectangle, count the number of ways the turtle can reach the bottom-right corner, if each turtle can only move right or down 1 unit at any time. The turtle is not allowed to move outside the rectangle, but, of course, the turtle can move on the boundary of the rectangle.

In less than one second, this problem was solved for all rectangles. He felt that the problem was too easy. This morning, Little Chef wanted more challenges. Thus, for each rectangle  $i$ , he cut and removed a rectangle of  $A_i \times B_i$  unit from the top-right corner. See the following figure for detail.



He could not solve this new version of the problem easily. Help him count the number of ways each turtle can reach the bottom-right corner using the same rule as before.

## Input

The first line of the input contains a single integer  $R$ . The description of  $R$  rectangles follows. Each description consists of a single line containing four space-separated integers  $N_i$ ,  $M_i$ ,  $A_i$ , and  $B_i$ .

## Output

For each rectangle, output a single line containing the number of ways, modulo 1,000,000,007.

## Constraints

- 1  $\leq R \leq 10$
- 2  $\leq N_i, M_i \leq 400,000$
- 1  $\leq A_i < N_i$
- 1  $\leq B_i < M_i$

## Example

Input:

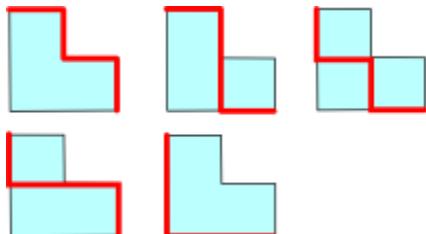
1

2 2 1 1

Output:

5

**Explanations:** In the sample case, there are 5 ways the turtle can reach the bottom right corner as follows:



[Home](#) » [Compete](#) » [December Challenge 2012](#) » WordNinjas

## WordNinjas Problem Code: WORDNINJ

**Attention!** This is challenge problem with the longest problem statement ever (it is even longer than for the legendary problem [CIELHACK](#)). You should invest some time in order to get all necessary points. But in the end there is a hint on how to get **Accepted**. So you will be awarded for your patience :)

Let's play [WordNinjas!](#) (**Note!** CodeChef is not affiliated with iTunes/Apple in any ways :))

In short, you should create words from letters that your ninja will get by attacking tiles. Some special tiles contain bonuses.

Here we consider some simplified version of the game where all the tiles are arranged in a line and we know their values in advance. Tiles are divided into blocks. At each move you can either skip the current block or take exactly one tile from it. Thus, you have a sequence of tile blocks as an input. Each block will be a string of characters where each character represents one tile. Each tile can be either a letter tile or some bonus tile. On each letter tile there will be written some letter of English alphabet and such tiles will be represented by the corresponding uppercase letters of English alphabet in the input. There exist 5 types of bonus tiles. The first one is a blank tile that behaves like an ordinary letter tile but you can write any letter on this tile when you create words (see below). The detailed description of all types of bonus tiles will be provided later.

The main aim of the game is to score as many points as possible. Most of the points can be gained by creating words. For this you have an array of length 7 that can be filled by letter or blank tiles that you get from the tile blocks. Each cell of the array can be occupied by at most one tile. **If at some move you take some letter or blank tile but all 7 cells in the array are occupied, this tile will be thrown away, otherwise this tile will occupy the first unoccupied cell of the array.** Thus, tiles in array always occupy several first cells. We number the cells in the array by numbers from 1 to 7. So if you have 5 tiles in the array they

will occupy cells 1, 2, 3, 4, 5. You can also throw away tiles from the array at any time if you need this. Remaining tiles will be moved preserving their relative order and still will occupy several first positions. For example, if you have 5 tiles in the array and you throw away tiles from the 2nd and the 4th cells then the tile from the 3rd cell will occupy the 2nd cell and the tile from the 5th cell will occupy the 3rd cell after this operation.

Finally, you can arbitrarily rearrange tiles in the array. Once you write some letters on all your blank tiles and get the rearrangement of tiles such that the corresponding sequence of letters forms a correct word you can play this word to get some points that depends on letters you have in this word and some other factors (see below). After this all the tiles except the last one will be thrown away from the array and the last tile will occupy the first position of the array. If it was the blank tile you can erase the letter you have written on it before and write another letter. Of course, you will be given a dictionary of all correct words that you can play. Note, that in all official input files this dictionary will be the same (see **Test Case Generation** section for details). But in the example we will use some dummy dictionary that consists of only two words :)

### Types of bonus tiles:

- **Blank tile.** As mentioned before, it occupies one cell in the array and any letter can be written on it. You will get 5 points by taking a blank tile since the magic panda keeps this tile. It will be denoted by the character ? (question mark) in the input. You can carry several blank tiles in the array at the same time. **When points for the played word are calculated it scores 0 points regardless of the letter that was written on it.**
- **Double letter tile.** It doubles points scored by the letter in the particular cell of the array. The tile that corresponds to the  $i$ -th cell will be represented by the digit  $i$  in the input.
- **Triple letter tile.** It triples points scored by the letter in the particular cell of the array. Since there are no uppercase digits in the English alphabet we will represent the triple letter tiles by lowercase English letters. That is, the character **a** in the input means the tile that triples score of the letter written on the tile that occupies the first cell of the array, character **b** do the same for the second cell and so on. Note that the letter tiles are represented by uppercase letters so there will be no ambiguities.
- **Double word tile.** It doubles points scored by all letters in the array. This tile will be represented by **+** character.
- **Triple word tile.** It triples points scored by all letters in the array. This tile will be represented by **\*** character.

You will get 2 points by taking tiles of the last four types since each such tile is kept by baby dragon. To describe in detail how these tiles behave let us introduce a concept of multipliers. Namely, each array cell (there are 7 of them) has its multiplier. Initially all these multipliers are equal to 1. A double letter tile assigns the multiplier for the corresponding cell to 2 and a triple letter tile assigns this multiplier to 3. So if multiplier for some position was 3 and you take the double letter tile for this cell it will be assigned to 2 and you will possibly get lower score for the word. The whole array also has a multiplier that initially equals to 1 and changes by double and triple word tiles in the same way as multipliers for particular cells. **After you play a word all multipliers are reset to 1.**

**How many points will you get for a particular word?** Each letter has some point value which is based on the letter's frequency in standard English writing: commonly used letters such as **A** or **S** are worth one point, while less common letters score higher, with **Q** and **Z** each worth 10 points. The full distribution of points by letters coincides with the standard Scrabble's one and can be found [here](#). For the sake of completeness we provide it here:

- 1 point: **A, E, I, L, N, O, R, S, T, U**
- 2 points: **D, G**
- 3 points: **B, C, M, P**
- 4 points: **F, H, V, W, Y**
- 5 points: **K**
- 8 points: **J, X**
- 10 points: **Q, Z**

Now assume that you play a word of length **L**, where  $2 \leq L \leq 7$  (there are no words of length 1 in SOWPODS). Denote by **C[i]** the score of the *i*-th letter in this word according to the distribution above. As mentioned above, if some letter is created from the blank tile then the corresponding score is 0. Further denote by **M[i]** the multiplier of the *i*-th cell of the array and by **M** the multiplier of the whole array. Then the number of points you will get for this word equals to

$$M * (M[1] * C[1] + \dots + M[L] * C[L]) + 3 * \max(L - 3, 0) + 30 * \max(L - 6, 0).$$

The second summand means bonus points for long words and equals to 3 for 4-letter words, 6 for 5-letter words, 9 for 6-letter words and 12 for 7-letter words. The last summand is additional large bonus of 30 points for 7-letter words. (In the actual WordNinjas game the whole score of the word is 10 times higher as well as the points for taking bonus tiles, but we get rid of this 10 in this problem.)

Thus, the total score you will get during the game is the sum of two sums, the first one is the sum of scores of all the played words during the game and the second one is the sum of points you got by taking bonus tiles.

Your goal is to maximize your score. But this is a challenge problem and you don't need to find the optimal solution.

## Input

The first line of the input contains an integer **N**, the number of words in the dictionary. Each of the following **N** lines contains one word composed of uppercase letters of English alphabet.

The next line contains a positive integer **M**, the number of tile blocks. Each of the following **M** lines contains one non-empty string of at most 10 characters that represents the set of tiles in the current block. Each character in this string can be one of the following:

- The uppercase letter of English alphabet, which represents the usual letter tile.
- The **?** character, which represents the blank tile.

- The **+** character, which represents the double word tile.
- The **\*** character, which represents the triple word tile.
- The digit from 1 to 7, which represents the double letter tile for the corresponding cell of the array.
- The lowercase letter from the set **{a, b, c, d, e, f, g}**, which represents the triple letter tile for the corresponding cell of the array.

There will be no white space characters in the input other than new lines (ASCII code 10).

---

## Output

For each tile block in the input, your output may consist of one, two or three lines. The first line should always contain one of the strings "**Take C**" or "**Skip**" depending on whether you take some tile from this block or skip it. Here **C** denotes the character from the block that you want to take. As was mentioned before if you take the letter or blank tile but all 7 cells in the array are occupied this tile will be thrown away. However you will get 5 points if you take a blank tile in any case. If you try to take the tile that is not present in this block your move will be ignored. But in any case **C** should be a character that represents some tile. Otherwise you will get **Wrong Answer**.

If you want to throw away some tiles, then start the next line with the string "**Throw**" (the last character is a space) followed by the non-empty list of tiles you want to throw. You should denote them as they were denoted in the input: letter tiles by uppercase letters of English alphabet, blank tiles by question marks. You can list them in any order. If some of the tiles you want to throw are not present in your array they will be ignored. But this list should contain no more than 7 characters in any case (we don't want to analyze millions of characters if you decide to output some enormous list). Also each character in this list should be either an uppercase English letter or a question mark and you should not separate characters in this list by white spaces. Otherwise you will get **Wrong Answer**.

If you want to play a word at this move, then start the next line with the string "**Play**" (the last character is a space) followed by a string composed of uppercase and lowercase letters of the English alphabet. The length of this word should be at least 2 and no more than 7. Here uppercase letters mean letters written on usual letter tiles and lowercase letters mean letters written on blank tiles. If you try to play incorrect word or the word that can not be created from available letter and blank tiles in your array your move will be ignored. If the word is correct and can be created from your tiles you will get the corresponding score for it. The word does not have to use all the tiles in the array. In this case the special judge will throw away all unnecessary tiles instead of you and then the word will be played. After you played the correct word only one tile remains in the array - the last tile of your word.

Any additional white spaces in your output will be ignored.

**Your output will be considered as correct if and only if it corresponds to the described format and you play at least one correct word during the game.** It is guaranteed that such output exists (see **Test Case Generation** section for details).

## Scoring

Your score for a particular test file is the score that your output will get for the whole game. The total score for a submission is the average score across all the test files. Your goal is to maximize the total score.

---

## Example

Input:

2

CODE

CHEF

13

\*+

CC

OA

D?

E

+

F

abcdefg

H

E

???

E

QZ321??

**Output 1:**

Take \*

Take C

Take O

Take D

Take E

Take +

Play CODE

Take F

Take a

Take H

Skip

Play CHEF

Take ?

Play slsEr

Take E

Play cHEF

Take ?

**Output 2:**

Take \*

Take C

Take O

Take D

Take E

Skip

Take F

Take d

Take H

Take E

Take ?

Take E

Take 2

Throw ED

Play CHEF

## Explanation

### Output 1:

Here we have two correct words **CODE** and **CHEF**. At first we take triple word tile, which bring us 2 points, then 4 letters **C, O, D, E** and double word tile which bring us 2 points. Then we play the word **CODE**, which bring us  $2 * (3 + 1 + 2 + 1) + 3 = 17$  points. After this we have letter **E** in the array. Note that if we would skip double word tile then the multiplier for the array would be 3 and we get **24** points for the word.

Next we want to play the word **CHEF**. At the next moves we take the letter **F**, then the triple letter tile **a** that corresponds to the first cell of the array and brings us 2 points, and the letter **H**, but skip **E** since we already have it. Then our little sister accidentally tries to play the word **CHEF**. Though this is a correct word we have only letters **E, F** and **H** in our possession and can't play this word. Luckily the special judge is kind and simply ignores this move.

At the next move we kick the magic panda and take the blank tile, which brings us 5 additional points. Now we can play the word **cHEF** by writing letter **C** on the blank tile but just before playing the word little sister again plays tricks on us and at first try to play word **sIstEr** (having 4 blank tiles!) which is incorrect according to our dictionary and then take the next letter **E**. So we try to play the word **cHEF** having letters **E, E, F, H** and one blank tile in our possession. Luckily special judge is kind and forgive us this mistake. It threw away unnecessary letter **E** and only then played the word which bring us  $(3 * 0 + 4 + 1 + 4) + 3 = 12$  points. Note

that the blank tile gives 0 points with any letter written on it. So we waste triple letter tile by playing this word.

At the last move we take the blank tile from the last block and receive additional 5 points. Game over. The total score for the game is  $2 + 2 + 17 + 2 + 5 + 12 + 5 = 45$ .

#### Output 2:

Here we take some tile from each block except the double word tile from the 6th block. In particular, we take the triple word tile from the first block, the triple letter tile corresponding to the 4th cell from the 8th block and the double letter tile corresponding to the second cell from the last block. Note that taking blank tile brings us 5 points but since we already have 7 tiles in the array we did not get this tile in our possession, the same is true for last letter **E** - taking this letter is simply ignored. After that we throw away letters **D** and **E** and are left with letters **C**, **E**, **F**, **H** and **O**. Finally we play the word **CHEF** which bring us  $3 * (3 + 2 * 4 + 1 + 3 * 4) + 3 = 75$  points since we have multiplier 3 for the whole array, multiplier 2 for the second cell and multiplier 3 for the 4th cell. As in the previous example unnecessary letter **O** was automatically thrown away before playing the word. Thus the total score for the game is  $2 + 2 + 5 + 2 + 75 = 86$ . As you can see playing just one word during the game can be better than playing two words even if one of them coincide with that only word :)

## Test Case Generation

The dictionary will be the same in all official test files and can be found [here](#). It consists of all **74414** words in [SOWPODS](#) list that have length from 2 to 7. Also, note that in each official input file words are sorted by their length and words of the same length are sorted in the alphabetical order. In generating tiles we have followed Scrabble standards. The total number of tiles in blocks of each official test file will be exactly 10000. There will be exactly 9000 letter and blank tiles. They together form 90 full English-language sets for Scrabble. Each such set consists of 98 letter tiles and 2 blank tiles. The distribution of letters in this set can be found [here](#). In particular, each input file will have exactly 180 blank tiles, exactly 90 letters **Z**, exactly 810 letters **A** and so on. The remaining tiles are bonus tiles that affect multipliers. There will be exactly 30 triple word tiles, 70 double word tiles, 300 triple letter tiles and 600 double letter tiles. Each double or triple letter tile will correspond to each cell of the array uniformly at random. To get the input sequence of block tiles some random permutation of mentioned 10000 tiles are taken and then it is divided by blocks, where each block has size from 1 to 10 uniformly at random.

## Hint

*In order to get **Accepted** you can simply output **M** times string **Take A** and then output **Play AA**.*

Why is this correct? Since special judge ignores tries to take incorrect tiles, all your moves for blocks that do not contain tile with letter **A** will be considered as skip moves. In the end you will definitely have all 7 positions in the array filled by letters **A** since there are **810** tiles

with letter **A** in the input. Since **AA** is a correct word and the special judge will throw away unnecessary tiles before playing the word, you will successfully play the word **AA** in the end.

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Sereja and Data Structures

## Sereja and Data Structures Problem Code: SEREJA

---

Sereja is a little student from the University of Kyiv. He studies computer science. He is rather experienced programmer and he knows a lot of advanced algorithms. The only thing he likes more than advanced algorithms is [segment tree data structure](#).

Every evening he goes to the loveliest place in Kyiv - The Chef's cafe. Chef is also interested in computer science, so they often discuss different problems. Last evening Sereja told Chef following problem: given an array  $A_1, A_2, \dots, A_N$  of  $N$  integers. The task is to find the product of ranges of the segments  $[i, j]$  over  $1 \leq i < j \leq N$ , where the range of the segment  $[i, j]$  means  $\max(A_i, A_{i+1}, \dots, A_j) - \min(A_i, A_{i+1}, \dots, A_j)$ .

Both Sereja and Chef couldn't find any efficient algorithm for this problem with segment tree. They gave up thinking of it, and Chef said we may calculate this value only for a **random array**, that means **all elements of the array are generated uniformly and independently at random**. Sereja were surprised to hear that, and he wanted to know about details. However Chef is very busy at the moment, so he asked you to solve this problem for a random array.

---

### Input

The first line contains a single integer  $N$ , then  $N$  integers  $A_1, A_2, \dots, A_N$  follow on the second line.

---

### Output

Output a single line containing the required product modulo  $1000000007$  ( $10^9+7$ ), since the answer can be very large.

---

### Constraints

$2 \leq N \leq 100000$  ( $10^5$ )

$0 \leq A_i < 1000000007$  ( $10^9+7$ )

All test cases, except Example, are created by the method described in Test Case Generation.

---

### Example

Input 1:

3

1 2 4

**Output 1:**

6

**Input 2:**

5

5 5 5 5 5

**Output 2:**

0

## Explanation

In **Input 1**, we should consider 3 segments  $[i, j] = [1, 2], [2, 3], [1, 3]$ .

The range of segment  $[1, 2]$  is  $\max(1,2) - \min(1,2) = 1$ .

The range of segment  $[2, 3]$  is  $\max(2,4) - \min(2,4) = 2$ .

The range of segment  $[1, 3]$  is  $\max(1,2,4) - \min(1,2,4) = 3$ .

Therefore, we obtain the answer  $1 * 2 * 3 = 6$ , and you should output  $6 \bmod 1000000007 = 6$ .

In **Input 2**, we should consider 10 segments, but all ranges of such segments are 0. And, of course, the answer is  $0 * 0 * 0 * 0 * 0 * 0 * 0 * 0 * 0 * 0 = 0$ .

Note that Example is created by manually, so these arrays are not random arrays. It is quite unlikely that the official test cases contain an array with equal elements such as **Input 2**.

## Test Case Generation

There are 15 official test files. For every test file,  $A_i$  are chosen from 0 to  $1000000006$  ( $10^9+6$ ) uniformly and independently at random.

The first 5 test files: **N** is chosen from 2 to 100 uniformly at random.

The next 5 test files: **N** is chosen from 2 to 100000 ( $10^5$ ) uniformly at random.

The last 5 test files: **N** = 100000 ( $10^5$ )

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Arigeom Beats

## Arigeom Beats Problem Code: ARIGEOM

Our amazing Chef recently met his best friend Joe at his place. Joe is a musician and owns a wide variety of musical instruments. He got introduced to the Arigeom beats in a music training session at Los Angeles. Arigeom beats, as the name suggests, is a combination of two series of beats. One of them has all its frequencies in [arithmetic progression](#) while the other series of beats has all its frequencies in [geometric progression](#).

A series of beats is represented by a beat notation, which is the sequence of the frequencies at which the beats are played.

To play Arigeom beats, two digital musical instruments are played simultaneously in front of an instrument known as Sono Phone. One instrument plays the beats in arithmetic progression while the other instrument plays the beats in geometric progression. Sono Phone will capture the beats played by each of the two instruments and process them into Arigeom beats. It will merge the beats, sort the beats in ascending order, and remove duplicate beats.

For example, suppose one instruments plays the following beats notation: (2, 5, 8, 11), an arithmetic progression, while the other plays (2, 4, 8, 16), a geometric progression. The resulting Arigeom beat notation recorded by Sono Phone will be (2, 4, 5, 8, 11, 16).

Because Joe knows how to play Arigeom beats, owns the instruments, and has his friend Chef with him as well, he can't wait anymore to play the beats. Joe has an Arigeom beat notation from his music book and decides that he would play the beats in arithmetic progression while Chef will play the beats in geometric progression. The beat notation consists of **N** beat frequencies  $F_1, F_2, \dots, F_N$ . However, he is now confused as to which subset of the beats has to be played by Chef and which one by himself.

Help Joe and Chef figure out the subsets of beats to be played by Joe and Chef.

### Input

The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows. For each test case, the first line contains an integer **N**. The second line contains **N** space-separated integers  $F_1, F_2, \dots, F_N$ .

### Output

For each test case, output two lines. The first line contain the beat notation to be played by Joe (in arithmetic progression). The second line contain the beat notation to be played by Chef (in geometric progression). Each beat notation must contain at least two beats. The beats in each beat notation must be sorted in ascending order.

If there are more than one possible pair of beat notations, output any one of them.

---

## Constraints

$$1 \leq T \leq 100$$

$$2 \leq N \leq 10,000$$

$$1 \leq F_i \leq 100,000$$

$$F_1 < F_2 < \dots < F_N$$

It is guaranteed that at least one pair of valid beat notations exists.

---

## Example

Input:

4

6

2 4 5 8 11 16

5

1 2 3 4 5

8

1 3 9 10 19 27 28 81

6

1 4 7 10 13 25

**Output:**

2 5 8 11

2 4 8 16

1 2 3 4 5

1 2 4

1 10 19 28

1 3 9 27 81

1 7 13

4 10 25

---

**Note**

In last example, the common ratio is 2.5 (**non-integer**) even when all the elements of geometric progression sequence are integers.

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Different Trips

## Different Trips Problem Code: DIFTRIP

---

The Chef is enjoying a wonderful vacation in Byteland. What makes the Chef impressed the most is the road system of the country. Byteland has **N** cities numbered 1 through **N**. City 1 is the capital of Byteland. The country also has **N-1** bidirectional roads connecting the cities.

The  $i$ -th road connects two different cities  $u_i$  and  $v_i$ . In this road system, people can travel between every pair of different cities by going through exactly one path of roads.

The roads are arranged in such a way that people can distinguish two cities only when both cities have different number of roads connected to it. Such two cities will be considered *similar*. For example, city **A** is similar to the capital if the number of roads connected to city **A** is equal to the number of roads connected to the capital.

On each day during the vacation, the Chef wants to have a trip as follows. He chooses two cities **A** and **B** such that the Chef will visit city **B** if he goes from **A** to the capital using the shortest path. Then, the Chef will visit the cities on the shortest path from **A** to **B** through this path. Please note that **A** may be equal to **B**; that means the Chef will enjoy the day in a single city.

The Chef does not want to have *similar* trips. Two trips are considered *similar* if and only if they both have the same number of visited cities and for each  $i$ , the  $i$ -th city visited in one trip is *similar* to the  $i$ -th city visited in the other trip.

The Chef wants to have as many different, namely not *similar*, trips as possible. Help him count the maximum number of possible trips such that no two of them are similar.

## Input

The first line of the input contains a single integer **N**. The  $i$ -th line of next **N-1** lines contains two space-separated integers  $u_i$  and  $v_i$ , denoting the  $i$ -th road.

## Output

Output a single line containing the maximum number of different trips.

## Constraints

$$1 \leq N \leq 100000 (10^5)$$

$$u_i \neq v_i$$

Every pair of different cities can be traveled by going through exactly one path of roads.

## Sample

Input:

3

2 1

3 1

**Output:**

3

## Explanation

In the sample, the country consists of three cities. There are two roads. The first road connects city 1 and city 2. The second road connects city 1 and city 3. Each day, the Chef can choose the following possible trips:

- $A = 1, B = 1$
- $A = 2, B = 2$
- $A = 3, B = 3$
- $A = 2, B = 1$
- $A = 3, B = 1$

However, since the trip ( $A = 2, B = 2$ ) is similar to the trip ( $A = 3, B = 3$ ), and the trip ( $A = 2, B = 1$ ) is similar to the trip ( $A = 3, B = 1$ ), there are only three possible different trips for the Chef.

[Home](#) » [Compete](#) » [December Challenge 2012](#) » Quasi-Polynomial Sum

## Quasi-Polynomial Sum Problem Code: QPOLYSUM

You are given a polynomial  $P(X) = C_D * X^D + \dots + C_1 * X + C_0$  with integer coefficients  $C_0, C_1, \dots, C_D$ . You are also given a non-negative integer  $Q$  and positive integers  $M$  and  $N$ . Your task is to find the following sum

$$(P(0) * Q^0 + P(1) * Q^1 + \dots + P(N - 1) * Q^{N - 1}) \bmod M.$$

Here  $A \bmod B$  means the remainder of the division of  $A$  by  $B$ .

Usually polynomials are given by the sequence of their coefficients. However, in this problem you will be given the sequence  $A_0, A_1, \dots, A_D$ , where  $A_i = P(i) \bmod M$ , as an input. One can prove that these values are enough to restore the value of  $P(K) \bmod M$  for any integer  $K$ . Therefore, the value of the above sum is uniquely determined by the values  $M, Q, N, D, A_0, A_1, \dots, A_D$ .

The function of the form  $P(X) * Q^X$  sometimes is called *quasi-polynomial*, hence the title of the problem.

## Input

The first line of the input contains an integer  $T$  denoting the number of test cases. The description of  $T$  test cases follows. The first line of each test case contains four space-separated integers  $M, Q, N, D$ . The second line contains  $D + 1$  space-separated integers  $A_0, A_1, \dots, A_D$ .

---

## Output

For each test case, output a single line containing the value of the corresponding sum.

---

## Constraints

- $1 \leq T \leq 5000$
  - $1 < M < 10^{18}$
  - $0 \leq Q < M$
  - $1 \leq N < 10^{100000}$
  - $0 \leq D < 20000$
  - $0 \leq A_i < M$  for  $i = 0, 1, \dots, D$
  - The sum of  $D + 1$  over each input file does not exceed **20000**.
  - The overall number of digits in all numbers  $N$  in each input file does not exceed  $10^5$ .
  - **M is not divisible by any number from 2 to D + 14, inclusive.**
  - It is guaranteed that there exists a polynomial  $P(X)$  of degree at most  $D$  with integer coefficients such that  $A_i = P(i) \bmod M$  for  $i = 0, 1, \dots, D$ .
- 

## Example

### Input:

```
2
101 2 5 0
1
289 1 6 2
1 4 9
```

### Output:

```
31
91
```

---

## Explanation

**Example case 1.** We have  $M = 101$ ,  $Q = 2$ ,  $N = 5$ ,  $D = 0$  and  $P(0) \bmod M = 1$ .

Therefore,  $P(X) = C_0$  and  $P(0) \bmod 101 = 1$ . Hence,  $C_0 \bmod 101 = 1$ . So we can take, for example,  $C_0 = 1$ . Then the corresponding sum takes the form of  $(P(0) * Q^0 + P(1) * Q^1 + \dots + P(N - 1) * Q^{N-1}) \bmod M = (1 * 2^0 + 1 * 2^1 + 1 * 2^2 + 1 * 2^3 + 1 * 2^4) \bmod 101 = (1 + 2 + 4 + 8 + 16) \bmod 101 = 31 \bmod 101 = 31$ . Note, that the polynomial  $P(X)$  is not unique. We can take, for example,  $C_0 = 102$  or  $C_0 = 2021$ . However, the same will be always the same.

**Example case 2.** We have  $M = 289$ ,  $Q = 1$ ,  $N = 6$ ,  $D = 2$  and  $P(0) \bmod M = 1$ ,  $P(1) \bmod M = 4$ ,  $P(2) \bmod M = 9$ . It is easy to see that we can take  $P(X) = (X + 1)^2$ . Then the corresponding sum takes the form of  $(1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2) \bmod 289 = (1 + 4 + 9 + 16 + 25 + 36) \bmod 289 = 91$ . Note that the polynomial  $P(X)$  again is not unique. In fact, the set of all polynomials of degree at most 2 that satisfy the conditions  $P(0) \bmod 289 = 1$ ,  $P(1) \bmod 289 = 4$ ,  $P(2) \bmod 289 = 9$  has the form of  $\{(X + 1)^2 + 289 * (C_2 * X^2 + C_1 * X + C_0) \mid C_0, C_1, C_2 \text{ are integers}\}$ .

COOK29

[December Cook-Off 2012](#)

23 Dec 2012  
21:30:00

2 hours 30 minutes

1113

[Home](#) » [Compete](#) » [December Cook-Off 2012](#) » [Packaging Cupcakes](#)

## Packaging Cupcakes Problem Code: MUFFINS3

Now that Chef has finished baking and frosting his cupcakes, it's time to package them. Chef has  $N$  cupcakes, and needs to decide how many cupcakes to place in each package. Each package must contain the same number of cupcakes. Chef will choose an integer  $A$  between 1 and  $N$ , inclusive, and place exactly  $A$  cupcakes into each package. Chef makes as many packages as possible. Chef then gets to eat the remaining cupcakes. Chef enjoys eating cupcakes very much. Help Chef choose the package size  $A$  that will let him eat as many cupcakes as possible.

---

### Input

Input begins with an integer  $T$ , the number of test cases. Each test case consists of a single integer  $N$ , the number of cupcakes.

---

### Output

For each test case, output the package size that will maximize the number of leftover cupcakes. If multiple package sizes will result in the same number of leftover cupcakes, print the largest such size.

---

### Constraints

- $1 \leq T \leq 1000$
  - $2 \leq N \leq 100000000 (10^8)$
- 

## Sample Input

2  
2  
5

---

## Sample Output

2  
3

---

## Explanation

In the first test case, there will be no leftover cupcakes regardless of the size Chef chooses, so he chooses the largest possible size. In the second test case, there will be 2 leftover cupcakes.

[Home](#) » [Compete](#) » [December Cook-Off 2012](#) » Reversing directions

## Reversing directions Problem Code: DIRECTI

Chef recently printed directions from his home to a hot new restaurant across the town, but forgot to print the directions to get back home. Help Chef to transform the directions to get home from the restaurant.

A set of directions consists of several instructions. The first instruction is of the form "Begin on XXX", indicating the street that the route begins on. Each subsequent instruction is of the form "Left on XXX" or "Right on XXX", indicating a turn onto the specified road.

When reversing directions, all left turns become right turns and vice versa, and the order of roads and turns is reversed. See the sample input for examples.

---

## Input

Input will begin with an integer **T**, the number of test cases that follow. Each test case begins with an integer **N**, the number of instructions in the route. **N** lines follow, each with exactly one instruction in the format described above.

---

## Output

For each test case, print the directions of the reversed route, one instruction per line. Print a blank line after each test case.

---

## Constraints

- $1 \leq T \leq 15$
  - $2 \leq N \leq 40$
  - Each line in the input will contain at most 50 characters, will contain only alphanumeric characters and spaces and will not contain consecutive spaces nor trailing spaces. By alphanumeric characters we mean digits and letters of the English alphabet (lowercase and uppercase).
- 

## Sample Input

2

4

Begin on Road A

Right on Road B

Right on Road C

Left on Road D

6

Begin on Old Madras Road

Left on Domlur Flyover

Left on 100 Feet Road

Right on Sarjapur Road

Right on Hosur Road

Right on Ganapathi Temple Road

---

## Sample Output

Begin on Road D

Right on Road C

Left on Road B

Left on Road A

Begin on Ganapathi Temple Road

Left on Hosur Road

Left on Sarjapur Road

Left on 100 Feet Road

Right on Domlur Flyover

Right on Old Madras Road

## Explanation

In the first test case, the destination lies on Road D, hence the reversed route begins on Road D. The final turn in the original route is turning left from Road C onto Road D. The reverse of this, turning right from Road D onto Road C, is the first turn in the reversed route.

[Home](#) » [Compete](#) » [December Cook-Off 2012](#) » Root of the Problem

## Root of the Problem Problem Code: TREEROOT

Chef has a binary tree. The binary tree consists of 1 or more nodes. Each node has a unique integer id. Each node has up to 2 children, which are identified by their ids, and each node is the child of at most 1 other node. A node **X** is considered to be an ancestor of node **Y** if node **Y** is a child of node **X** or if there is some node **Z** for which **X** is an ancestor of **Z** and **Y** is a child of **Z**. No node is an ancestor of itself. A special node called the root node is an ancestor of all other nodes.

Chef has forgotten which node of his tree is the root, and wants you to help him to figure it out. Unfortunately, Chef's knowledge of the tree is incomplete. He does not remember the ids of the children of each node, but only remembers the sum of the ids of the children of each node.

---

## Input

Input begins with an integer **T**, the number of test cases. Each test case begins with an integer **N**, the number of nodes in the tree. **N** lines follow with 2 integers each: the id of a node, and the sum of the ids of its children. The second number will be 0 if the node has no children.

---

## Output

For each test case, output on a line a space separated list of all possible values for the id of the root node in increasing order. **It is guaranteed that at least one such id exists for each test case.**

---

## Constraints

- $1 \leq T \leq 50$
  - $1 \leq N \leq 30$
  - All node ids are between 1 and 1000, inclusive
- 

## Sample Input

```
2
1
4 0
6
1 5
2 0
3 0
4 0
5 5
6 5
```

---

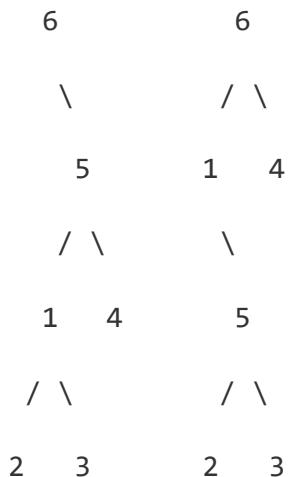
## Sample Output

4

6

## Explanation

In the first sample test case, there is only one node, which is clearly the root. In the second test case, there are two non-isomorphic trees that satisfy the constraints, as seen in the following picture:



[Home](#) » [Compete](#) » [December Cook-Off 2012](#) » Expected Greatest Common Divisor

# Expected Greatest Common Divisor Problem

Code: EXGCD

Several integers  $X_0, X_1, \dots, X_{K-1}$  are randomly chosen from some intervals. Namely,  $X_0$  is randomly chosen between  $A_0$  and  $B_0$ , inclusive,  $X_1$  is randomly chosen between  $A_1$  and  $B_1$ , inclusive, and so on. We are interested in the expected value of the greatest common divisor of the  $X_i$  which is the average of  $\text{GCD}(X_0, X_1, \dots, X_{K-1})$  over all possible choices of  $X_0, X_1, \dots, X_{K-1}$ . In order to avoid floating-point precision issues, we instead use the following output method. Suppose the expected value of the greatest common divisor of all the  $X_i$  is  $P/Q$ , where  $P$  and  $Q$  are relatively prime positive integers. Find an integer  $N$  between 0 and 1000000006, inclusive, for which  $(P + Q * N)$  is divisible by  $1000000007 = 10^9 + 7$ .

## Input

Input will begin with an integer  $T$ , the number of test cases. Each test case begins with an integer  $K$ , the number of integers to be chosen.  $K$  pairs of integers  $A_i, B_i$  follow.

## Output

For each test case, output the value **N** according to the problem statement. If multiple such values exist, print any one of them. If no such value exists, print -1.

---

## Constraints

- $1 \leq T \leq 50$
  - $2 \leq K \leq 5$
  - $1 \leq A_i \leq B_i \leq 200000 (2 * 10^5)$
- 

## Sample Input

```
4
2
2 4
3 5
3
1 2
1 2
1 2
3
1 12
1 12
1 12
2
2700 2701
2612 2724
```

---

## Sample Output

```
333333334
```

875000005

722222226

314159265

## Explanation

**Sample test 1.** Here  $X_0$  is randomly chosen between 2 and 4, inclusive, and  $X_1$  is randomly chosen between 3 and 5, inclusive. The distribution of the greatest common divisor of  $X_0$  and  $X_1$  can be seen from the following table:

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

The expected value is therefore  $(1 + 3 + 1 + 2 + 1 + 4 + 1 + 1 + 1) / 9 = 15 / 9 = 5 / 3$ . Since  $5 + 3 * 33333334 = 1000000007$ , the answer is 33333334.

**Sample test 2.** Here each of  $X_0$ ,  $X_1$ ,  $X_2$  is randomly chosen between 1 and 2, inclusive. So we have 8 possibilities in all. The greatest common divisor is 2 when  $X_0 = X_1 = X_2 = 2$  and it is 1 in the 7 remaining cases. The expected value is therefore  $(1 * 7 + 2 * 1) / 8 = 9 / 8$ . Since  $9 + 8 * 875000005 = 700000049 = 7 * 1000000007$ , the answer is 875000005.

**Sample test 3.** Here  $P$  is 23 and  $Q$  is 18.

**Sample test 4.** Just enjoy the answer :)

[Home](#) » [Compete](#) » [December Cook-Off 2012](#) » Game on a Grid

## Game on a Grid

Problem Code: GRIDGAME

Alice and Bob are playing a game. A single pawn is placed on a grid. The grid consists of cells  $(X, Y)$  for non-negative integers  $X$  and  $Y$ . Some cells of the grid have been marked as impassable. A move consists of choosing some positive integer  $D$  and moving the pawn from  $(X, Y)$  either to  $(X - D, Y)$  or to  $(X, Y - D)$ , of course, if this cell belongs to the grid. In other words, we move the pawn a positive number of steps along the row or column towards the origin. The pawn is not allowed to land on nor pass through an impassable cell. Alice and Bob alternate moves, with Alice going first. The first player unable to make a move loses.

Alice and Bob will play several games on the same grid, but with different starting positions of the pawn. Assuming both players play optimally, determine which player will win each game.

---

## Input

Input will begin with an integer **T**, the number of test cases that follow. Each test case will begin with an integer **N**, the number of impassable cells. **N** lines follow with 2 integers each: the X and Y coordinates of an impassable cell. Following this is a line with an integer **Q**, the number of games to be played on this grid. **Q** lines follow with 2 integers each: the X and Y coordinates of the pawn.

---

## Output

For each test case, print the winner of each of the **Q** games, one per line.

---

## Constraints

- $1 \leq T \leq 10000 (10^4)$
  - $1 \leq N \leq 100000 (10^5)$
  - $1 \leq Q \leq 100000 (10^5)$
  - The sum of **N** over all test cases will not exceed  $100000 (10^5)$
  - The sum of **Q** over all test cases will not exceed  $100000 (10^5)$
  - All coordinate values will be between 0 and  $1000000000 (10^9)$ , inclusive
  - All impassable cells will be distinct
  - No starting position of the pawn will be impassable
- 

## Sample Input

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

6 5

---

## Sample Output

Alice

Bob

Alice

Bob

---

## Explanation

In the first game, the pawn is at cell (2, 2). In order to win Alice can begin by moving the pawn to (1, 2). Bob's only available move is to (0, 2) since (1, 1) is impassable. Alice then moves the pawn to (0, 0), leaving Bob with no available moves.

Here is the small portion of the table of winning and losing positions. Here X-direction goes from the left to the right and Y-direction goes from the top to the bottom. Letter 'A' at the cell (X, Y) means that Alice will win when (X, Y) is the starting position of the pawn and 'B' means that Bob will win. We use different background color for Alice and Bob cells for convenience. Impassable cells have black background color.

	0	1	2	3	4	5	6
0	B	A	A	A	A	A	A
1	A		B	A	A	A	A
2	A	B	A	A		B	A
3	A	A	A	B	A	A	A
4	A	A	A	A	B	A	A
5	A	A	A	A	A	A	B

JAN13

[January Challenge 2013](#)01 Jan 2013  
15:00:00

14 days

3228

[Home](#) » [Compete](#) » [January Challenge 2013](#) » The Minimum Number Of Moves

# The Minimum Number Of Moves

Problem Code: SALARY

---

Little chief has his own restaurant in the city. There are  $N$  workers there. Each worker has his own salary. The salary of the  $i$ -th worker equals to  $W_i$  ( $i = 1, 2, \dots, N$ ). Once, chief decided to equalize all workers, that is, he wants to make salaries of all workers to be equal. But for this goal he can use only one operation: choose some worker and increase by 1 salary of each worker, except the salary of the chosen worker. In other words, the chosen worker is the loser, who will be the only worker, whose salary will be not increased during this particular operation. But loser-worker can be different for different operations, of course. Chief can use this operation as many times as he wants. But he is a busy man. That's why he wants to minimize the total number of operations needed to equalize all workers. Your task is to find this number.

---

## Input

The first line of the input contains an integer  $T$  denoting the number of test cases. The description of  $T$  test cases follows. The first line of each test case contains a single integer  $N$  denoting the number of workers. The second line contains  $N$  space-separated integers  $W_1, W_2, \dots, W_N$  denoting the salaries of the workers.

---

## Output

For each test case, output a single line containing the minimum number of operations needed to equalize all workers.

---

## Constraints

- $1 \leq T \leq 100$
  - $1 \leq N \leq 100$
  - $0 \leq W_i \leq 10000 (10^4)$
- 

## Example

Input:

2

3

1 2 3

2

42 42

Output:

3

0

## Explanation

**Example Case 1.** Chief can equalize all salaries in 3 turns:

Turn ID	IDs of involved workers	Salaries after the move
1	1 2	2 3 3
2	1 2	3 4 3
3	1 3	4 4 4

**Example Case 2.** All salaries are already equal. He doesn't need to do anything.

[Home](#) » [Compete](#) » [January Challenge 2013](#) » Chef of the Year

## Chef of the Year Problem Code: CVOTE

Chefs from all over the globe gather each year for an international convention. Each chef represents some country. Please, note that more than one chef can represent a country.

Each of them presents their best dish to the audience. The audience then sends emails to a secret and secure mail server, with the subject being the name of the chef whom they wish to elect as the "**Chef of the Year**".

You will be given the list of the subjects of all the emails. Find the country whose chefs got the most number of votes, and also the chef who got elected as the "**Chef of the Year**" (the chef who got the most number of votes).

### Note 1

If several countries got the maximal number of votes, consider the country with the lexicographically smaller name among them to be a winner. Similarly if several chefs got the maximal number of votes, consider the chef with the lexicographically smaller name among them to be a winner.

### Note 2

The string **A** =  $a_1a_2\dots a_n$  is called lexicographically smaller than the string **B** =  $b_1b_2\dots b_m$  in the following two cases:

- there exists index  $i \leq \min\{n, m\}$  such that  $a_j = b_j$  for  $1 \leq j < i$  and  $a_i < b_i$ ;
- **A** is a proper prefix of **B**, that is,  $n < m$  and  $a_j = b_j$  for  $1 \leq j \leq n$ .

The characters in strings are compared by their ASCII codes.

Refer to function **strcmp** in C or to standard comparator **<** for **string** data structure in C++ for details.

---

## Input

The first line of the input contains two space-separated integers **N** and **M** denoting the number of chefs and the number of emails respectively. Each of the following **N** lines contains two space-separated strings, denoting the name of the chef and his country respectively. Each of the following **M** lines contains one string denoting the subject of the email.

---

## Output

Output should consist of two lines. The first line should contain the name of the country whose chefs got the most number of votes. The second line should contain the name of the chef who is elected as the "**Chef of the Year**".

---

## Constraints

- $1 \leq N \leq 10000 (10^4)$
  - $1 \leq M \leq 100000 (10^5)$
  - Each string in the input contains only letters of English alphabets (uppercase or lowercase)
  - Each string in the input has length not exceeding 10
  - All chef names will be distinct
  - Subject of each email will coincide with the name of one of the chefs
- 

## Example 1

Input:

1 3

Leibniz Germany

Leibniz

Leibniz

Leibniz

**Output:**

Germany

Leibniz

---

## Example 2

**Input:**

4 5

Ramanujan India

Torricelli Italy

Gauss Germany

Lagrange Italy

Ramanujan

Torricelli

Torricelli

Ramanujan

Lagrange

**Output:**

Italy

Ramanujan

---

## Example 3

**Input:**

2 2

Newton England

Euclid Greece

Newton

Euclid

**Output:**

England

Euclid

## Explanation

**Example 1.** Here we have only one chef **Leibniz** and he is from **Germany**. Clearly, all votes are for him. So **Germany** is the country-winner and **Leibniz** is the "**Chef of the Year**".

**Example 2.** Here we have chefs **Torricelli** and **Lagrange** from **Italy**, chef **Ramanujan** from **India** and chef **Gauss** from **Germany**. **Torricelli** got 2 votes, while **Lagrange** got one vote. Hence the **Italy** got 3 votes in all. **Ramanujan** got also 2 votes. And so **India** got 2 votes in all. Finally **Gauss** got no votes leaving **Germany** without votes. So the country-winner is **Italy** without any ties. But we have two chefs with 2 votes: **Torricelli** and **Ramanujan**. But since the string "**Ramanujan**" is lexicographically smaller than "**Torricelli**", then **Ramanujan** is the "**Chef of the Year**".

**Example 3.** Here we have two countries with 1 vote: **England** and **Greece**. Since the string "**England**" is lexicographically smaller than "**Greece**", then **England** is the country-winner. Next, we have two chefs with 1 vote: **Newton** and **Euclid**. Since the string "**Euclid**" is lexicographically smaller than "**Newton**", then **Euclid** is the "**Chef of the Year**".

[Home](#) » [Compete](#) » [January Challenge 2013](#) » Three Different Numbers

# Three Different Numbers

Problem Code: THREEDIF

---

This is probably the simplest problem ever. You just need to count the number of ordered triples of different numbers ( $X_1, X_2, X_3$ ), where  $X_i$  could be any positive integer from 1 to  $N_i$ , inclusive ( $i = 1, 2, 3$ ).

No, wait. I forgot to mention that numbers  $N_1, N_2, N_3$  could be up to  $10^{18}$ . Well, in any case it is still quite simple :)

By the way, because of this the answer could be quite large. Hence you should output it modulo  $10^9 + 7$ . That is you need to find the remainder of the division of the number of required triples by  $10^9 + 7$ .

---

## Input

The first line of the input contains an integer  $T$  denoting the number of test cases. The description of  $T$  test cases follows. The only line of each test case contains three space-separated integers  $N_1, N_2, N_3$ .

---

## Output

For each test case, output a single line containing the number of required triples modulo  $10^9 + 7$ .

---

## Constraints

- $1 \leq T \leq 1000$
- $1 \leq N_i \leq 10^{18}$

---

## Example

Input:

5

3 3 3

2 4 2

1 2 3

25 12 2012

1 1 2013

Output:

```
6
4
1
578880
0
```

---

## Explanation

**Example case 1.** We have the following triples composed of different numbers up to 3:

```
(1, 2, 3)
(1, 3, 2)
(2, 1, 3)
(2, 3, 1)
(3, 1, 2)
(3, 2, 1)
```

**Example case 2.** Here the triples are:

```
(1, 3, 2)
(1, 4, 2)
(2, 3, 1)
(2, 4, 1)
```

**Example case 3.** Here the only triple is (1, 2, 3).

**Example case 4.** Merry Christmas!

**Example case 5.** ... and Happy New Year! By the way here the answer is zero since the only choice for  $X_1$  and for  $X_2$  is 1, so any such triple will have equal numbers.

[Home](#) » [Compete](#) » [January Challenge 2013](#) » End Of The World

## End Of The World Problem Code: CHEFHACK

**Important!!!** Human's are in Danger... It may be The End of the Universe!!!

**Aliens** had stolen the important data about the **Birth of the Human beings** and they kept this data in their secured **Data Center**. With the help of these data along with the advanced

technologies Aliens had initiated a project called **It's The End** to get rid of the human beings and occupy the Earth.

On the other side, **Chef Po** is a good and very well-known **Hacker**. He is a great human lover, so he decided to get the data back from the Aliens by hacking their Data Center.

The Aliens Data Center contains various servers arranged in **N** rows and **N** columns and the Master Server will be at the last row and the last column of the Data Center. Denote the server at the intersection of the **i**-th row and the **j**-th column of the Data Center as **Server[i][j]** ( $1 \leq i, j \leq N$ ). So in this notation Master Server is **Server[N][N]**. Two servers are connected to each other if they are located either at the same row and consecutive columns or at the same column and consecutive rows. In other words, **Server[i][j]** and **Server[i-1][j]** are connected to each other for all **i** and **j** such that  $2 \leq i \leq N$  and  $1 \leq j \leq N$ . Also **Server[i][j]** and **Server[i][j-1]** are connected to each other for all **i** and **j** such that  $1 \leq i \leq N$  and  $2 \leq j \leq N$ .

Aliens had divided the **Data File** with the data about the Birth of the Human beings into **N \* N** parts denoted as **F[i][j]** ( $1 \leq i, j \leq N$ ) and placed the part **F[i][j]** at the **Server[i][j]**. Each server has a password which is some non-negative integer. Chef Po needs to crack all the servers in order to get all parts of the **Data File**.

Aliens are really strange beings so they single out three types of passwords:

- **Prime Password**, which is the prime number. The list of prime numbers starts as **2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, ...**
- **Even Password**, which is the even non-prime number. This list starts as **0, 4, 6, 8, 10, 12, 14, ...** Note, that we exclude **2** since it is a prime.
- **Odd Password**, which is the odd non-prime number. This list starts as **1, 9, 15, 21, 25, 27, 33, 35, 39, 45, 49, ...** So it is the list of all odd positive numbers where primes are excluded.

For simplicity we call server secured by the Even Password as Even Server. Similarly we define notion of Odd Server and Prime Server.

Each Odd or Even server has a friendly screen with the color indicator. Assume that you enter the password trying to guess the actual password at this server. Then indicator will become green if you enter the number which is smaller than the actual password, it will become red if your number is greater than the actual password or you will get the access to the server (and, thus, to the needed part of the Data File) if the password is correct.

On the other hand, each Prime Server has no such screen and the only way to crack it is to try all prime numbers until you find out the correct password. So our Chef simply iterate over the list of primes in increasing order and try them until he finds the correct password.

Probably most of you think that to crack servers with friendly screen Chef Po will use binary search. But he is quite weird Hacker and doesn't know this concept. To crack server with the friendly screen he will use the following strategy. He will try all even numbers starting from 0 (even number 2) until he either finds the correct password or the indicator becomes

red which automatically would mean that the password is odd and equals to the **K-1** where **K** is the even number that was entered at last. So he enters **K-1** in the latter case and get the access to the server.

Let's consider some examples.

- Let the password be **4**. It is an Even Password so Chef will see the friendly screen which means for him that the server is either Even or Odd. Hence he will try passwords in the order **0, 2, 4** making 2 unsuccessful tries.
- Let the password be **9**. It is an Odd Password so again Chef will see the friendly screen which means for him that the server is either Even Server or Odd Server. Hence he will try passwords in the order **0, 2, 4, 6, 8, 10, 9** making 6 unsuccessful tries. Note that after he enters **8** the indicator becomes green indicating that the password is greater than **8**, on the other hand, after he enters **10** the indicator becomes red indicating that the password is less than **10**. Since the only integer number greater than **8** and less than **10** is **9** he will enter it at the last step getting the access to the server.
- Finally, let the password be **11**. It is a Prime Password so Chef will not see the friendly screen which means for him that the server is Prime Server. Hence he will try passwords in the order **2, 3, 5, 7, 11** making 4 unsuccessful tries.

The Alien Data Center has very interesting vulnerability, that Chef has noticed by analyzing Alien's secured correspondence. If Chef crack some Even Server **S** then all Even Servers connected to **S** become cracked as well, and all Even Servers connected to these servers also become cracked and so on. More formally, the Even Server **S'** becomes cracked if it is connected with **S** via possibly other Even Servers. That is, there exists a sequence of Even Servers such that each two consecutive servers in this sequence are connected, the first server is **S** and the last server is **S'**. Chef calls this vulnerability as "**Grid Hacking Mechanism**". The same is true for Odd Servers. By it is no longer true for Prime Servers. So each Prime Server should be cracked individually.

Now the final tactic for Chef is the following.

- He will consider the servers in row-major order.
- That is, at first he consider servers of the first row in the order **Server[1][1], Server[1][2], ..., Server[1][N]**. Then he considers servers of the second row in the order **Server[2][1], Server[2][2], ..., Server[2][N]** and so on until he reaches the last server in the last row (which is the Master Server).
- For each server he at first check whether it is already cracked by the "**Grid Hacking Mechanism**".
- If yes then he gathers the corresponding part of the Data File and move on.
- Otherwise he checks for the friendly screen at this server and apply one of the cracking tactics described above.
- That is, when he sees the friendly screen he will enter even numbers starting from zero until he finds the password in the way described above. After that he gathers the corresponding part of the Data File and also some other servers become cracked by the "**Grid Hacking Mechanism**".
- And when he doesn't see the screen he will enter prime numbers in increasing order until he finds the password. After that he again gathers the corresponding

part of the Data File but no other servers become cracked since "**Grid Hacking Mechanism**" does not work for Prime Servers.

- Finally he will reach the Master Server, crack it and gather the last part of the Data File.

But the only way to recombine the original file is to use special Alien utility called **winripzar** on the Master Server. It also has the password. Chef Po is a great hacker, so he found out using his special skills that the password for **winripzar** equals to the number of unsuccessful tries used by him to crack all the servers in the Data Center.

After cracking all the servers and gathering all the parts of the Data File Chef became so happy that forgot the number of unsuccessful tries he made during this cracking marathon. But he noted down passwords for all the servers and now asks you for help. Calculate the number of unsuccessful tries Chef did during cracking of all the servers and let him save the Earth!

---

## Input

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows. The first line of each test case contains a single integer **N**. Each of the following **N** lines contains **N** space separated integers. The **j**-th number at the **i**-th line denotes the password at the **Server[i][j]**.

---

## Output

For each test case, output a single line containing the number of unsuccessful tries used to crack all the servers in the Data Center.

---

## Constraints

- $1 \leq T \leq 8$
  - $1 \leq N \leq 350$
  - $0 \leq \text{each password} < 10^7$
- 

## Example

Input:

2

3

2 6 4

4 8 9

7 9 4

2

8 4

15 4

**Output:**

20

13

## Explanation

### Example case 1.

The type of each server can be found in the following table.

Prime	Even	Even
Even	Even	Odd
Prime	Odd	Even

Now the process of cracking of servers by the Chef will be the following.

- At first Chef cracks the **Server[1][1]** secured by the password **2** which is a prime number. Hence Chef does not see the friendly screen and tries prime numbers in increasing order. Since **2** is the smallest prime number he cracks this server without unsuccessful tries. Since it is a Prime Server than "**Grid Hacking Mechanism**" does not work here.
- Next Chef cracks the **Server[1][2]** secured by the password **6** which is an Even Password (even non-prime number). Hence Chef sees the friendly screen and tries even numbers in order **0, 2, 4, 6** making 3 unsuccessful tries to crack this server. Since it is an Even Server than by "**Grid Hacking Mechanism**" the following 3 servers also become cracked: **Server[1][3]**, **Server[2][1]** and **Server[2][2]** since they all Even Servers and connected to the **Server[1][2]** via other Even Servers (see the above table). Note, that Even **Server[3][3]** does not become cracked since it is not connected to **Server[1][2]** via other Even Servers.
- Next Chef consider the **Server[1][3]**. As was mentioned above it was already cracked so Chef simply gathers the corresponding part of the Data File without entering any numbers.

- The same is true for the next two servers **Server[2][1]** and **Server[2][2]**.
- But **Server[2][3]** is secured by the odd non-prime number **9** so it is an Odd Server and Chef tries numbers in order **0, 2, 4, 6, 8, 10, 9** making 6 unsuccessful tries to crack it. Applying "**Grid Hacking Mechanism**" for this server has no effect since it is not connected to any other Odd Server.
- The **Server[3][1]** is Prime and Chef tries numbers in order **2, 3, 5, 7** making 3 unsuccessful tries to crack it.
- The **Server[3][2]** is Odd and was not cracked earlier. Hence Chef tries numbers in order **0, 2, 4, 6, 8, 10, 9** making 6 unsuccessful tries to crack it.
- The **Server[3][3]** is Even and was not cracked earlier. Hence Chef tries numbers in order **0, 2, 4** making 2 unsuccessful tries to crack it.

Summarizing we see that Chef made 20 unsuccessful tries in all. Their distribution among servers can be found in the following table.

0	3	0
0	0	6
3	6	2

### Example case 2.

The type of each server can be found in the following table.

Even	Even
Odd	Even

Now the process of cracking of servers by the Chef will be the following.

- The **Server[1][1]** is Even and Chef tries numbers in order **0, 2, 4, 6, 8** making 4 unsuccessful tries to crack it.
- The **Server[1][2]** is Even and already cracked by the "**Grid Hacking Mechanism**".
- The **Server[2][1]** is Odd and Chef tries numbers in order **0, 2, 4, 6, 8, 10, 12, 14, 16, 15** making 9 unsuccessful tries to crack it.
- The **Server[2][2]** is Even and already cracked by the "**Grid Hacking Mechanism**".

Thus, Chefs made 13 unsuccessful tries and their distribution among servers can be found in the following table.

4	0
9	0

---

### Warning!

The input file size could reach almost 8MB so be sure you are using fast I/O method.

[Home](#) » [Compete](#) » [January Challenge 2013](#) » Little Elephant and Alcohol

## Little Elephant and Alcohol Problem Code: LEALCO

---

The Little Elephant and his friends from the Zoo of Lviv were returning from the party. But suddenly they were stopped by the policeman Big Hippo, who wanted to make an alcohol test for elephants.

There were **N** elephants ordered from the left to the right in a row and numbered from **0** to **N-1**. Let **R[i]** to be the result of breathalyzer test of **i**-th elephant.

Considering current laws in the Zoo, elephants would be arrested if there exists **K** consecutive elephants among them for which at least **M** of these **K** elephants have the maximal test result among these **K** elephants.

Using poor math notations we can alternatively define this as follows. The elephants would be arrested if there exists **i** from **0** to **N-K**, inclusive, such that for at least **M** different values of **j** from **i** to **i+K-1**, inclusive, we have  $R[j] = \max\{R[i], R[i+1], \dots, R[i+K-1]\}$ .

The Big Hippo is very old and the Little Elephant can change some of the results. In a single operation he can add **1** to the result of any elephant. But for each of the elephants he can apply this operation at most once.

What is the minimum number of operations that the Little Elephant needs to apply, such that the sequence of results, after all operations will be applied, let elephants to avoid the arrest? If it is impossible to avoid the arrest applying any number of operations, output **-1**.

---

### Input

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows. The first line of each test case contains three space-separated integers **N**, **K**, **M**. The second line contains **N** space-separated integers **R[0]**, **R[1]**, ..., **R[N-1]** denoting the test results of the elephants.

---

### Output

For each test case, output a single line containing the minimum number of operations needed to avoid the arrest.

---

### Constraints

- $1 \leq T \leq 10$
- $1 \leq M \leq K \leq N \leq 17$

- $1 \leq R[i] \leq 17$
- 

## Example

Input:

4

5 3 2

1 3 1 2 1

5 3 3

7 7 7 7 7

5 3 3

7 7 7 8 8

4 3 1

1 3 1 2

Output:

0

1

1

-1

---

## Explanation

**Example case 1.** Let's follow the poor math definition of arrest. We will consider all values of  $i$  from **0** to **N-K = 2**, inclusive, and should count the number of values of  $j$  described in the definition. If it less than **M = 2** then this value of  $i$  does not cause the arrest, otherwise causes.

i	$\{R[i], \dots, R[i+K-1]\}$	$\max\{R[i], \dots, R[i+K-1]\}$	For which $j = i, \dots, i+K-1$ we have $R[j] = \max$	Conclusion
i=0	{1, 3, 1}	max = 3	$R[j] = 3$ for $j = 1$	does not cause the arrest
i=1	{3, 1, 2}	max = 3	$R[j] = 3$ for $j = 1$	does not cause the arrest
i=2	{1, 2, 1}	max = 2	$R[j] = 2$ for $j = 3$	does not cause the arrest

So we see that initial test results of the elephants do not cause their arrest. Hence the Little Elephant does not need to apply any operations. Therefore, the answer is 0.

**Example case 2.** We have  $N = 5$ ,  $K = 3$ ,  $M = 3$ . Let's construct similar table as in example case 1. Here the value of  $i$  will cause the arrest if we have at least 3 values of  $j$  described in the definition.

i	$\{R[i], \dots, R[i+K-1]\}$	$\max\{R[i], \dots, R[i+K-1]\}$	For which $j = i, \dots, i+K-1$ we have $R[j] = \max$	Conclusion
i=0	{7, 7, 7}	max = 7	$R[j] = 7$ for $j = 0, 1, 2$	causes the arrest
i=1	{7, 7, 7}	max = 7	$R[j] = 7$ for $j = 1, 2, 3$	causes the arrest
i=2	{7, 7, 7}	max = 7	$R[j] = 7$ for $j = 2, 3, 4$	causes the arrest

So we see that for initial test results of the elephants each value of  $i$  causes their arrest. Hence the Little Elephant needs to apply some operations in order to avoid the arrest. He could achieve his goal by adding 1 to the result  $R[2]$ . Then results will be  $\{R[0], R[1], R[2], R[3], R[4]\} = \{7, 7, 8, 7, 7\}$ . Let's check that now elephants will be not arrested.

i	$\{R[i], \dots, R[i+K-1]\}$	$\max\{R[i], \dots, R[i+K-1]\}$	For which $j = i, \dots, i+K-1$ we have $R[j] = \max$	Conclusion
i=0	{7, 7, 8}	max = 8	$R[j] = 8$ for $j = 2$	does not cause the arrest
i=1	{7, 8, 7}	max = 8	$R[j] = 8$ for $j = 2$	does not cause the arrest
i=2	{8, 7, 7}	max = 8	$R[j] = 8$ for $j = 2$	does not cause the arrest

So we see that now test results of the elephants do not cause their arrest. Thus we see that using 0 operations we can't avoid the arrest but using 1 operation can. Hence the answer is 1.

**Example case 3.** We have  $N = 5$ ,  $K = 3$ ,  $M = 3$ . Let's construct similar table as in example case 1. Here the value of  $i$  will cause the arrest if we have at least 3 values of  $j$  described in the definition.

i	$\{R[i], \dots, R[i+K-1]\}$	$\max\{R[i], \dots, R[i+K-1]\}$	For which $j = i, \dots, i+K-1$ we have $R[j] = \max$	Conclusion
---	-----------------------------	---------------------------------	--	------------

i=0	{7, 7, 7}	max = 7	R[j] = 7 for j = 0, 1, 2	causes the arrest
i=1	{7, 7, 8}	max = 8	R[j] = 8 for j = 3	does not cause the arrest
i=2	{7, 8, 8}	max = 8	R[j] = 8 for j = 3, 4	does not cause the arrest

So we see that for initial test results of the elephants the value of  $i = 0$  causes their arrest. Hence the Little Elephant needs to apply some operations in order to avoid the arrest. He could achieve his goal by adding 1 to the result  $R[1]$ . Then results will be  $\{R[0], R[1], R[2], R[3], R[4]\} = \{7, 8, 7, 8, 8\}$ . Let's check that now elephants will be not arrested.

i	$\{R[i], \dots, R[i+K-1]\}$	max{R[i], ..., R[i+K-1]}	For which $j = i, \dots, i+K-1$ we have $R[j] = \text{max}$	Conclusion
i=0	{7, 8, 7}	max = 8	R[j] = 8 for j = 1	does not cause the arrest
i=1	{8, 7, 8}	max = 8	R[j] = 8 for j = 1, 3	does not cause the arrest
i=2	{7, 8, 8}	max = 8	R[j] = 8 for j = 3, 4	does not cause the arrest

So we see that now test results of the elephants do not cause their arrest. Thus we see that using 0 operations we can't avoid the arrest but using 1 operation can. Hence the answer is 1. Note that if we increase by 1 the result  $R[2]$  instead of  $R[1]$  then the value  $i = 2$  will cause the arrest since  $\{R[2], R[3], R[4]\}$  will be  $\{8, 8, 8\}$  after this operation and we will have 3 values of  $j$  from 2 to 4, inclusive, for which  $R[j] = \text{max}\{R[2], R[3], R[4]\}$ , namely,  $j = 2, 3, 4$ .

**Example case 4.** When  $M = 1$  the Little Elephant can't reach the goal since for each value of  $i$  from 0 to  $N-K$  we have at least one value of  $j$  for which  $R[j] = \text{max}\{R[i], R[i+1], \dots, R[i+K-1]\}$ .

[Home](#) » [Compete](#) » [January Challenge 2013](#) » A Fence for Byteland

## A Fence for Byteland Problem Code: KILOWHAT

Byteland is in turmoil! The citizens of Byteland cannot agree on how many bytes are in a kilobyte. Members of the Decocratic party claim there are 1000 bytes in a kilobyte, and members of the Rebytelican party insist there are 1024 bytes in a kilobyte. The country is on the verge of a civil war. These times call for drastic measures, and Chef has decided the only option is to build a fence separating the citizens based on their opinions. Once the fence is built, the Decocrats should be fully separated from the Rebytelicans. Note that the fence is allowed to pass through the location of a citizen.

Chef wants your help in designing the fence. Chef will be happy as long as the fence separates the citizens properly, but would prefer a shorter fence. You'll be given a map of the city, and need to produce a polygonal fence that separates the citizens. The polygon must be strictly non-self-intersecting (that is, no two edges may cross, and no vertex may belong to more than two edges). Additionally, one of the following must be true:

- All Decocrats are inside or on the border of the polygon, and all Rebytelicans are outside or on the border of the polygon, or
  - All Decocrats are outside or on the border of the polygon, and all Rebytelicans are inside or on the border of the polygon.
- 

## Input

Input will begin with an integer **N**, the number of citizens in Byteland. **N** lines follow with 3 integers each: the **x** and **y** coordinates of the citizen, and the number of bytes the citizen believes are in a kilobyte (this will always be either 1000 or 1024).

---

## Output

First, output the number of vertices in your polygon. It should be an integer from 3 to 1000, inclusive. Then output the vertices of your polygon in order, **x** coordinate then **y** coordinate. All coordinates must be integers with absolute value at most 2000. All vertices must be distinct.

Just to reiterate, your output will be accepted if and only if all of the following conditions are satisfied:

- Let **L** be the number of vertices in your polygon. Output must contain exactly **2 \* L + 1** integers separated by spaces or line breaks, with first integer in the output equal to **L**.
  - The number of vertices of your polygon (that is **L**) lies between 3 and 1000, inclusive.
  - Coordinates of all vertices of your polygon must be integers with absolute value at most 2000.
  - All vertices of your polygon are distinct (which actually follows from the next condition).
  - Every two non-consecutive sides of your polygon have no common points considered as closed segments on the plane (it is equivalent formulation of strictly non-self intersecting property of the polygon described above).
  - Your polygon separates Decocrats and Rebytelicans as described above (which is, of course, the most important condition).
- 

## Scoring

Let **D** be the greatest distance between any two citizens, and **P** the perimeter of your polygon (the sum of the lengths of all the sides). Your score is then calculated as **10 \* P / (D \* sqrt(N))**. The score of the submission is the average score over all test files. Smaller scores will earn more points.

---

## Example

**Input:**

16

1 1 1024

1 2 1024

1 3 1024

1 4 1000

2 1 1000

2 2 1000

2 3 1000

2 4 1024

3 1 1000

3 2 1024

3 3 1000

3 4 1000

4 1 1000

4 2 1024

4 3 1024

4 4 1024

**Output:**

5

4 1

3 2

3 4

1 4

2 1

## Explanation

For this sample output, the Decocrats at (2, 2) and (2, 3) are inside the fence. The Rebytelicans at (1, 1), (1, 2), (1, 3), (4, 2), (4, 3), (4, 4) are outside the fence. Everyone else is on the border of the fence. Here  $D = 3 * \sqrt{2} = 4.2426$ ,  $P = \sqrt{2} + 2 + 2 + \sqrt{10} + 2 = 10.5765$ , and  $N = 16$ . The sample output would therefore score  $10 * 10.5765 / (4.2426 * \sqrt{16}) = 6.2323$ . Note that this output is optimal for this case. But you can output any polygon satisfying constraints above in order to get Accepted.

## Test Case Generation

There are two test case generation methods, each making up half of the test cases.

In the first method,  $N$  is chosen randomly between 100 and 200, inclusive. A real value  $Q$  is chosen randomly between 0.2 and 0.8, inclusive. Then  $N$  distinct points are chosen randomly with  $x$  and  $y$  coordinates between 1 and 1000, inclusive. Each point will be a Decocrat with probability  $Q$ , and a Rebytelican with probability  $(1-Q)$ .

In the second method, two integers  $W$  and  $H$  are chosen randomly between 10 and 14, inclusive. Two integers  $dW$  and  $dH$  are chosen randomly between 20 and 70, inclusive. A real value  $Q$  is chosen randomly between 0.2 and 0.8, inclusive.  $N$  is set to  $W * H$ , and the  $N$  points are generated as  $(dW*i, dH*j)$  with  $i$  ranging from 1 to  $W$ , inclusive, and  $j$  ranging from 1 to  $H$ , inclusive. Each point will be a Decocrat with probability  $Q$ , and a Rebytelican with probability  $(1-Q)$ .

Note that example does not satisfy any of these schemes. But it is guaranteed that every official test file will satisfy one of these schemes.

## Hint

In order to get Accepted you could simply output a polygon having all citizens on its border. It means that your answer will be accepted if, for example, you output some strictly non-self-intersecting polygon such that each citizen lies on its border. Actually, it is even possible to output all citizens in some order to get AC. (All these details on the hint have already contained in the comments.)

# Dividing Products Problem Code: DIVPRO

---

Dr. Bobo, while working in his laboratory, thought of an interesting problem. He has spent considerable time trying to find a solution but in vain. So he turns to you to find the solution using your special powers of coding.

For a positive integer **N**, he defines a function **DIVPRO(N)** as follows.

- Let **N** be **L**-digit number and **n<sub>i</sub>** is the **i**-th digit in the decimal representation of **N** (**i** = 1, 2, ..., **L**). So we can write **N** = **n<sub>1</sub>n<sub>2</sub>...n<sub>L</sub>**.
- Then **DIVPRO(N)** = **n<sub>1</sub>** / **n<sub>2</sub>** \* **n<sub>3</sub>** / **n<sub>4</sub>** \* ... (i.e., we alternate division and multiplication of digits).
- The result is calculated from left to right.
- Division here is performed in standard mathematical way, so the result of the division can be non-integer number.
- If division by 0 occurs at any point for a given **N**, then **DIVPRO(N)** is undefined in such a case.

Let's consider some examples:

- DIVPRO(1)** = 1. In fact, **DIVPRO(N)** = **N** for any 1-digit number **N**.
- DIVPRO(42)** = 4 / 2 = 2 is an integer.
- DIVPRO(123)** = 1 / 2 \* 3 = 3 / 2 = 1.5 is non-integer.
- DIVPRO(370)** = 3 / 7 \* 0 = 0, while intermediate result was 3 / 7 which is non-integer.
- DIVPRO(3465009)** = 3 / 4 \* 6 / 5 \* 0 / 0 \* 9 is undefined since we have division by zero.

Now Dr. Bobo would like to know how many **L**-digit numbers have their **DIVPRO** value equal to **V** and he wants your help. Since this number can be quite large output it modulo **2<sup>32</sup>**, that is, you need to find the remainder of the division of the answer by **2<sup>32</sup>**.

---

## Input

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows. The only line of each test case contains two space-separated integers **L** and **V**.

---

## Output

For each test case, output a single line containing the number of **L**-digit positive integers, whose **DIVPRO** value is **V**. As was mentioned above you should print this number modulo **2<sup>32</sup>**.

---

## Constraints

- 1 ≤ **T** ≤ 320000 (320 thousands)

- $1 \leq L \leq 36$
  - $0 \leq V < 10^{18}$
- 

## Example

### Input:

4

2 0

3 27

5 24

10 45

### Output:

0

5

486

2349595

## Explanation

**Example case 1.** No 2-digit number has **DIVPRO** value of **0** (as leading zeros are not allowed). Hence the answer is zero.

**Example case 2.** The only 3-digit numbers having **DIVPRO** value of **27** are **319, 629, 913, 926** and **939**. So there are 5 such numbers in all.

---

## Warning!

The input file size could reach 7MB so be sure you are using fast I/O method.

## Hotel Balifornia Problem Code: HOB

Byteland has always been popular among tourists. Seeing a business opportunity, Chef has built a brand new Hotel Balifornia there. The hotel has  $R$  unique comfortable rooms for guests' purpose. The rooms are numbered by integers from  $0$  to  $R-1$ , inclusive. All rooms in the hotel are on the same floor and arranged in a circle. Namely,

- the room next to the room  $0$  is the room  $1$ ;
- the room next to the room  $1$  is the room  $2$ ;
- ...
- the room next to the room  $R-2$  is the room  $R-1$ ;
- finally, the room next to the room  $R-1$  is the room  $0$ .

When  $R = 1$  we have only one room and it is next to itself.

Below, we will be needing the notion of  $k$ -th next room to the given room. It should be quite clear from the notation what does it mean but let's define it formally to avoid any ambiguities. The  $1$ -st next room to the room  $r$  is simply the next room to the room  $r$ . Assume that the notion of  $(k-1)$ -th next room is already defined. Then the  $k$ -th next room to the room  $r$  is the room next to the  $(k-1)$ -th next room to the room  $r$ .

At the time of check-in, among other things, the guest is required to provide his first name and the number of hours he will be staying in the hotel. Chef has sent a special machine called "Room Number Generator" (RNG) for allocating the rooms to the guests. The RNG takes a string (the first name of the guest) as the input and returns the room id. This is how RNG does it:

- The string is composed of lowercase and uppercase letters of English alphabet.
- At first, uppercase letters (from 'A' to 'Z', inclusive) are replaced by the corresponding lowercase letters: 'A' by 'a', 'B' by 'b', ..., 'Z' by 'z'.
- Next, each letter is replaced by an integer value: 'a' by  $1$ , 'b' by  $2$ , ..., 'z' by  $26$ .
- Then we consider thus obtained list of positive integers as a representation of some integer in base  $33$ . Namely, if  $D_1, D_2, \dots, D_L$  is the list of integers obtained after encoding all the letters we will consider the following value:  $D_1 * 33^{L-1} + D_2 * 33^{L-2} + \dots + D_{L-1} * 33 + D_L$ .
- Finally, RNG takes the remainder of division of the number, obtained at the previous step, by the number of rooms in the hotel. So this will be  $(D_1 * 33^{L-1} + D_2 * 33^{L-2} + \dots + D_{L-1} * 33 + D_L) \bmod R$ . This value is the room id returned by RNG. Well, at least Chef deems that RNG returns this value...

Consider some examples:

- "Ab" will be at first replaced by "ab" and then will be represented as  $1 * 33 + 2 = 35$ . If the number of rooms in the hotel is  $40$  then the room will be  $35 \bmod 40 = 35$ .
- "Chef" will be at first replaced by "chef" and then will be represented as  $3 * 33^3 + 8 * 33^2 + 5 * 33 + 6 = 116694$ . If the number of rooms in the hotel is  $37$  then the room will be  $116694 \bmod 37 = 33$ .

Chef noted that for some unlucky values of  $R$  regardless of the name of the guest, RNG returns only rooms from some range that does not cover all rooms in the hotel. This, of

course, would be disaster for Chef. For example, for  $R = 33$  rooms **0, 27, 28, 29, 30, 31, 32** can not be the values returned by RNG. Chef is smart guy and quickly realizes that this happens if and only if  $R$  is divisible by **33**. Hence he ensures that the number of rooms in the hotel is not divisible by **33**.

The hotel manager was having fun time, 'RNGing' the names of his acquaintances, just when he discovered that his wife's RNG number came out the same as his friend's! Horrified he visited the VP and explained the problem that different guests may end up in the same room. The VP thought for a moment, and then suggested that the one who comes later be allotted a different room. How clever! But which room? The hotel management fought for several days over this. Ultimately the lift boy's scheme was taken up.

- The lift boy is fond of sequences and number theory. Hence he will use the following sequence in his scheme: **1, 2, 3, 6, 11, 20, 37, ...** Here each term of the sequence, starting from the 4-th one, equals to the sum of three previous terms. We denote the  $n$ -th term of this sequence as  $A_n$ . So  $A_1 = 1$ ,  $A_2 = 2$ ,  $A_3 = 3$ ,  $A_4 = 6$ , and so on.
- Assume that RNG number of the current guest is  $r_0$ .
- At first we check the room  $r_0$  and if it is free we settle guest in this room.
- Otherwise, we check the room  $r_1$  which is  $A_1$ -th next to the room  $r_0$ . If it is free we settle guest in this room. Note that since  $A_1 = 1$  then, actually,  $r_1$  is simply the next room to  $r_0$ .
- Otherwise, we check the room  $r_2$  which is  $A_2$ -th next to the room  $r_1$ . If it is free we settle guest in this room. Note that since  $A_2 = 2$  then,  $r_2$  is the room, next to the room that is next to the room  $r_1$  (ignore this, if it blows your mind).
- In general, at  $k$ -th step we have all rooms  $r_0, r_1, \dots, r_{k-1}$  occupied and we check the room  $r_k$  which is  $A_k$ -th next to the room  $r_{k-1}$ . If it is free we settle guest in this room and finish the process. Otherwise we continue.
- If we can't find the free room in any finite number of steps, we inform the guest that we unable to settle him. In this case we provide him the minimal number of minutes after which he could come again so that we could find the free room for him by the above scheme, **assuming that we start the process again from the room  $r_0$** .
- Please, be sure that you are using exactly this definition to find the required number of minutes. According to some weird facts about 'RNG' below, the room 'RNGing' to the guest later could differ from  $r_0$ , but we still use the old value to perform the lift boy's scheme.
- Actually, no guest will return back to the hotel if we ask him to come back later without settling him in some room. The reason is that only celebrities visit Chef's hotel (see the example ;)) and they take a huge offense to the Chef if they were not settled in any room. Hence they will never return to the hotel again in this case. So actually we inform them the minimal number of minutes just out of courtesy :)

Chef hears this and is a bit worried about guests' response, so he requests you to analyze the inconvenience that will be caused to the guests. If the guest was settled in some room, the inconvenience equals to the least integer  $k$  for which the room  $r_k$  is free, so, actually, it is the number of rooms that was found occupied while processing this guest. Otherwise it is

equal to the number of minutes he should wait to come again if no free room was found for him by lift boy's scheme.

After invention of the lift boy's scheme Chef also realizes that his old fears about unlucky values of **R** is useless but still he believes that values of **R** divisible by **33** could lead to larger average guests' inconvenience than other values of **R**. So, just in case, he still ensures that **R** is not divisible by **33**.

What Chef and hotel management does not realize is that RNG has a mind of its own. Don't get scared! It just remembers the inconvenience caused to each guest and it uses this information to adjust the returned value. Namely, RNG add to the initial RNG value described above the sum of inconveniences caused to all previous guests. Then this number is taken modulo **R** to get the correct room number and RNG returns it. See example for clarity.

You will be given the list of all visits to the Hotel Balifornia in increasing order of time and should output the inconvenience caused to each guest. To make the output more readable print the dash before the inconvenience of each guest that was not settled in any room.

Also note that Chef's hotel is very popular, so the time interval between two consecutive visits is strictly less than 24 hours. Hence we will be providing only hour and minute of the day for each visit.

We could have several visits at the same moment of time. In this case we process guests in order they appear in the input.

If we have a visiting guest and some other guest leaving the hotel at the same moment, then the room which will be vacated by the leaving guest will be considered as free when applying the lift boy's scheme to the visiting guest. The same is true when deciding the waiting minutes for not settled guests.

We assume that all guests are different. But they could have equal first names. Indeed, Daniel Craig and Daniel Radcliffe are different persons having equal first names :)

## Input

The first line of the input contains an integer **T**, denoting the number of test cases. The description of **T** test cases follows. The first line of each test case contains two space-separated integers **N** and **R**, denoting the number of visits to the hotel and the number of rooms in the hotel respectively. The description of **N** visits follows. Namely, each of the following **N** lines contains 3 space separated integers **H**, **M**, **G** followed by a space and the string **S**. Here **H** and **M** is the hour and the minute of the day at which guest comes. Recall that visits are given in increasing order of time but the time interval between two consecutive visits is strictly less than 24 hours. **G** is the number of hours for which the guest will stay and **S** is his/her first name.

---

## Output

For each visit of each test, output a single line containing the inconvenience caused to the guest who made this visit. Don't forget to output '-' (dash) before result for each guest that should wait (see above). Please do not consider this dash as minus sign and do not assume that inconvenience is negative in this case. This dash is used only to make output more readable and have no effect on the actual inconvenience value.

---

## Constraints

- $1 \leq T \leq 1000$
  - $1 \leq N \leq 200000 (2 * 10^5)$
  - $1 \leq R \leq 500$  ( $R$  is not divisible by 33)
  - $0 \leq H \leq 23$
  - $0 \leq M \leq 59$
  - $0 \leq G \leq 9999999 (10^7 - 1)$
  - The total number of visits over the input does not exceed  $200000 (2 * 10^5)$
  - The name of each guest contains only uppercase and lowercase letters of English alphabet and has length from 1 to 10, inclusive
- 

## Example

Input:

1

8 4

0 10 15 Brad

0 12 6 Kanye

1 10 24 Angelina

5 38 48 John

12 55 56 Bill

12 56 6 Tom

20 1 2 Nicole

3 0 24 Daniel

Output:

```
0
0
0
2
2
-134
1
1
```

## Explanation

Guest's name	Cur Time	Prev. inconv.	RNG output	Room	Status	Order of rooms	Output
Brad	00:10	0	91513 + 0	1	----	1	0
Kanye	00:12	0	13097144 + 0	0	-B--	0	0
Angelina	01:10	0	60979313407 + 0	3	KB--	3	0
John	05:38	0	375983 + 0	3	KB-A	3 → 0 → 2	2
Bill	12:55	2	82083 + 2	1	-BJA	1 → 2 → 0	2
Tom	12:56	4	22288 + 4	0	BBJA	0 → 1 → 3 → 2	-134
Nicole	20:01	138	558693338 + 138	0	B-JA	0 → 1	1
Daniel	03:00	139	158240589 + 139	0	B-J-	0 → 1	1

- The **Prev. inconv.** column contains the total inconvenience caused to all previous guests.
- By **RNG output** here we mean the value before taking it modulo **R**.

- The **Room** column contains the room that was initially assigned to the guest and it is the actual value returned by RNG. It is equal to the value in the previous column take modulo  $R = 4$ .
- The **Status** column contains the current status of each room: '-' indicates a free room, while uppercase letter indicates that the room is occupied by the person whose first name starts at this letter.
- The **Order of rooms** column contains the sequence of rooms produced by the lift boy's scheme for the current guest.
- **Note on the Tom's visit.** After 4 steps of the lift boy's scheme we found out that all rooms in the hotel are occupied. Hence we can't settle Tom in any room. The room that will become free next is the room 1 occupied currently by Brad. This will happen at 15:10 which is 134 minutes after the 12:56 (the time of Tom's visit).
- **Note on the Daniel's visit.** We see that his visit time is less than for the previous visit. So it seems to violate the following constraint: times of visits are given in increasing order. In fact, this only means that Daniel came next day! (Again recall that the time interval between two consecutive visits is strictly less than 24 hours.)

[Home](#) » [Compete](#) » [January Challenge 2013](#) » A New Door

## A New Door Problem Code: ANDOOR

---

Now Chef is very good in cooking; his dishes are the best in the city, so he wants to make his restaurant prettier. The first, what customer sees, when comes to his restaurant is the door. So he will start with it. He has found a new design last night. He will cut out some circles from the door and fill the holes by the glass to make windows of peculiar form.

Formally, we can model the process Chef will perform to create the windows as follows. Let the door be a rectangle at the coordinate plane with corners at the points  $(0, 0)$ ,  $(W, 0)$ ,  $(0, H)$ ,  $(W, H)$ . Initially the door is colored white, while the remaining part of the coordinate plane is colored black. At each step Chef colors in black some circle including all its inner points. After Chef will color all required circles, the black part of the door (that is, black regions that lie in the rectangle corresponding to the door) is cut out and replaced by the glass.

The main measure of peculiarity of the door is the perimeter of all the windows considering only those parts of windows borders lying inside the door (see example case 2 for clarity). Chef is very tired after coloring so many circles and asks you to find this number.

---

### Input

The first line of the input contains an integer  $T$  denoting the number of test cases. The description of  $T$  test cases follows. The first line of each test case contains 3 space-separated integers  $W$ ,  $H$ ,  $N$ , denoting the measurements of the door, and the number of circles. The description of  $N$  circles follows in the next  $N$  lines. Namely, the  $i$ -th line among

them contains **3** space-separated real numbers **X**, **Y**, **R** with exactly two digits after the decimal point, denoting the coordinates of the center and the radius of the circle that Chef will cut out at the **i**-th step.

---

## Output

For each test case, output a single line containing the sum of the perimeters of all windows as described above. Your answer will be considered as correct if it has an absolute error less than **10<sup>-6</sup>**.

---

## Constraints

- **1 ≤ T ≤ 1000**
  - **1 ≤ W ≤ 1000**
  - **1 ≤ H ≤ 1000**
  - **1 ≤ N ≤ 1000**
  - The sum of all values of **N** over the input does not exceed **5000**
  - **0 < R ≤ 1000**
  - **0 ≤ X, Y ≤ 1000**
  - **X, Y, R** have exactly two digits after the decimal point
  - All circles are different, that is, every two circles have either different radii or different centers (or both)
- 

## Example

Input:

2

8 10 8

2.00 2.00 1.00

4.00 2.00 0.50

6.00 2.00 1.00

4.00 5.00 1.50

3.00 6.00 0.50

3.00 7.00 0.75

5.00 7.00 1.00

4.00 8.00 1.00

7 7 3

1.00 1.00 2.00

3.00 3.00 1.50

6.00 5.00 1.50

**Output:**

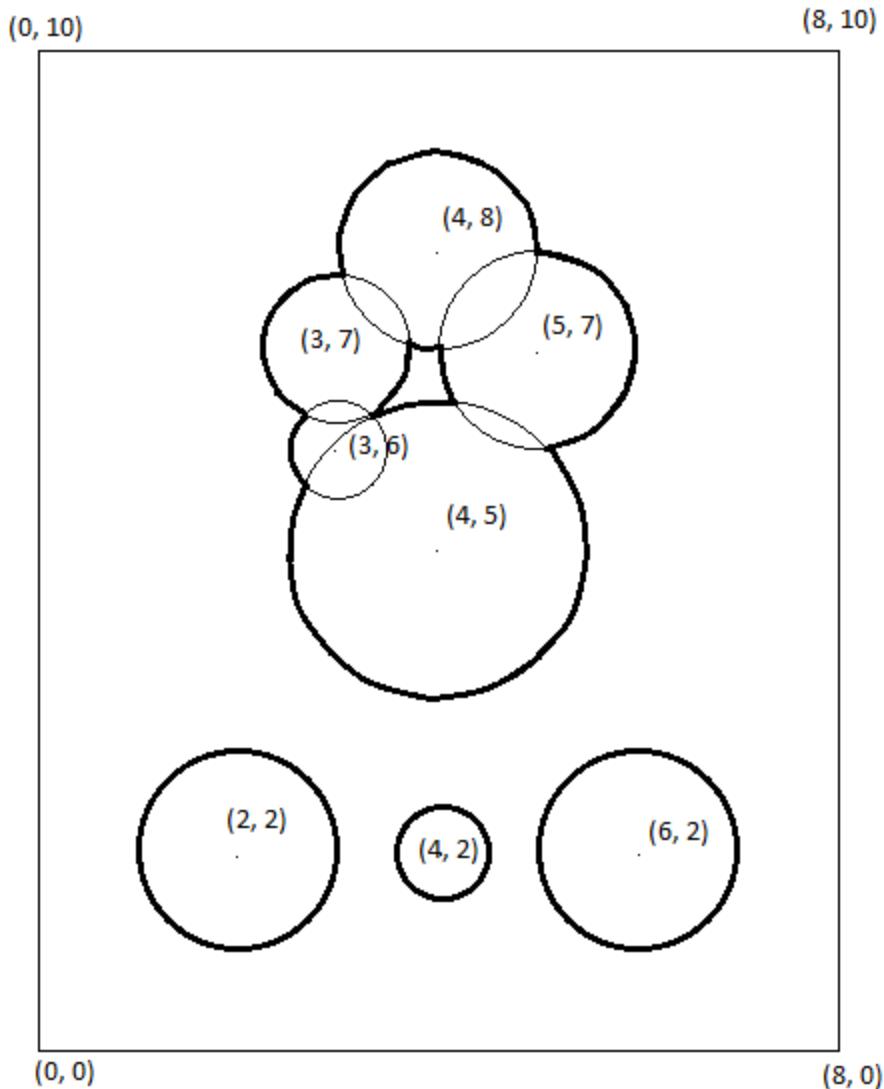
33.6404284

17.2058395

---

## **Explanation**

**Example case 1.** The answer is the length of the bold lines on the following picture:



**Example case 2.** The answer is the length of the bold lines on the following picture:

