A - Total Beans
Input File: TotalBeansIn.txt

Farmer Ryan sells beanbags of various sizes at his farm stand. His friend Nora, a bean counter, has filled six bags with beans after carefully counting the amount to be placed into each beanbag. Ryan wants to know the total number of beans Nora placed into the six beanbags in order to replenish his inventory of beans.

Inputs
There will be one line of input that contains the number of beans placed into each of the six bean bags.

Outputs
There will be one line of output containing the string "**`Ryan has to plant`**" followed by the number of beans in each of the six bags, followed by the string "**`beans`**". The strings and the integer must be separated by a space.

Sample inputs
241 675 897 12 4354 7625

Sample output
Ryan has to plant 13804 beans

B- Mister Chips
Input File: MisterChipsIn.txt

Mister Chips distributes chips to patrons of a poker casino. In order to cover his expenses, he charges a surcharge on the chips each patron purchases as shown below.

Chip Denomination	Surcharge
$1	2.000%
$5	1.000%
$10	0.220%
$100	0.112%

Your task is to write a program to determine how much to charge each patron for their chips, to the nearest penny.

Inputs
The first line will contain the number of patrons, P, purchasing chips. This will be followed by one line of input per patron that contains the number of $1, $5, $10, and $100 chips they wish to purchase.

Outputs
There will be one line of output for each patron that contains the cost of the chips, which includes Mister Chip's surcharge rounded to the nearest penny. The output lines should be formatted exactly as shown below.

Sample inputs
2
1 2 5 4
700 401 0 10003

Sample output
$461.68
$1,004,159.39

C - Whoops
Input File: WhoopsIn.txt

The auto correction feature of a word processor can be annoying when it changes a misspelled word to a word we did not intend to type. (Whoops.) To evaluate the merits of several auto correction algorithms, you are to write a program to compare an intended paragraph text to the corrected text and determine the number of incorrectly changed words. Your program should ignore extra spaces between words but not case sensitivity.

Inputs

The first line will contain the number of paragraphs, P, to compare. This will be followed by P sets of paragraphs. The first paragraph in a set will be the intended paragraph and the second paragraph in a set will be the auto corrected paragraph. All paragraphs will be separated by a blank line. Paragraphs will consist of no more than 10 lines.

Outputs

There will be one line of output for each set of paragraphs. The line will contain the number of words incorrectly changed by the auto correction algorithm.

Sample inputs

2
A stitch in time saves nine.

A *twitch of mine* saves nine.

Once upon a midnight dreary, while I pondered weak and weary,
Over many a quaint and curious volume of forgotten lore,
While I nodded, nearly napping, suddenly there came a tapping,
As of some one gently rapping, rapping at my chamber door.
Tis some visitor, I muttered, tapping at my chamber door,
Only this, and nothing more.

Months upon a midnight dreary, while I pondered weak and weary,
Over many a quaint and curious volume of forgotten *law*,
While I nodded, nearly *rapping*, suddenly there came a tapping,
As of some one gently rapping, rapping at *me* chamber door.
It's some visitor, I muttered, tapping at my chamber door,
Only this, and nothing *sore*.

Sample output

3
6

D - Circular Tour
Input File: CircularTourIn.txt

N cars are positioned on a circular touring road, each with a different amount of fuel. The cars are numbered 1 to N in the order they are encountered in a clockwise direction, with car number 1 being an arbitrarily chosen car. At most there is one car that can tour the entire road. Write a program to determine the number of the car that can make the tour in a clockwise direction, provided it can transfer the fuel from each car it reaches into its own tank.

Inputs
The first line will contain the number of tours to be processed. This will be followed by one data set per tour consisting of two lines per data set. The first line will contain the length of the circular road (an integer) and the number of cars on this road. The second line will contain two integer data items per car, the first being the range of the car (distance it can travel) and the second being the car's location (the clockwise circular arc distance from car 1 to the car). The first of these pairs will be car 1's range and location, the second integer pair will be car 2's range and location, etc.

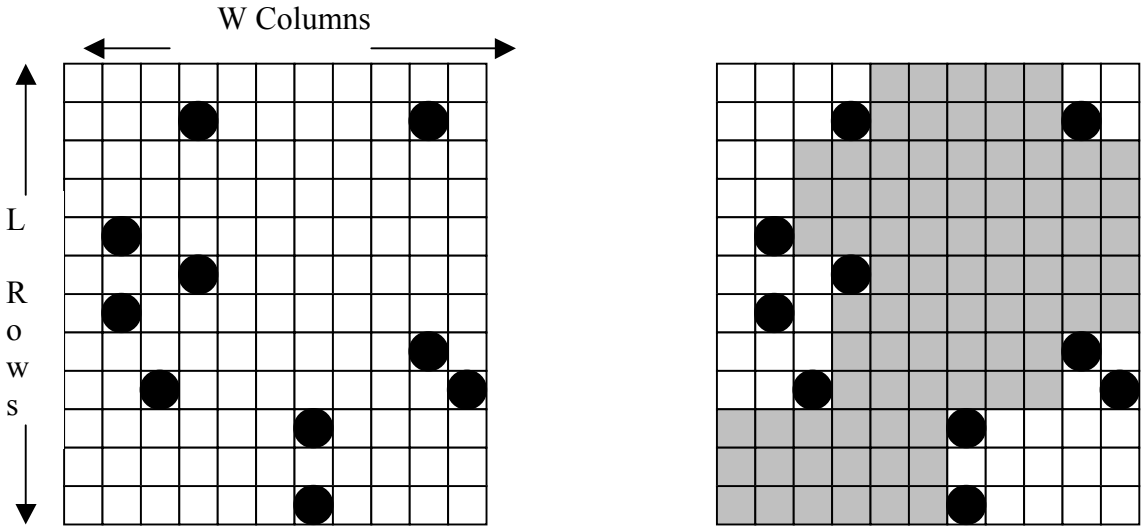
Outputs
There will be one line of output for each circular touring road. The line will contain the car number that can complete the tour, or in the event that no car can complete the tour, it will contain the string "no car can make the tour".

Sample inputs
```
3
20 4
1 0 1 1 13 3 5 12
314 4
150 0 50 25 100 200 10 250
60 6
5 0 7 6 5 14 6 17 20 38 17 59
```

Sample output
```
3
no car can make the tour
6
```

E - Mowing
Input File: MowingIn.txt

Uncle Mo mows rectangular lawns that are L feet long and W feet wide. The lawns are divided into 1 foot by 1 foot squares, and some of these squares have trees growing within them (shown as black circles in the figure below).



Uncut Lawn

Cut Lawn (shaded portion)

Mo has a 3 x 3 foot square lawn mower that can only move parallel to the edges of the lawn (vertical or horizontal). It must remain entirely within the rectangular lawn at all times and can never overlap a square containing a tree. The lawn mower cannot be rotated and cannot be picked up and moved over a tree.

You are to write a program to determine which squares of the lawn can be cut ("reached") by the lawn mower (the shaded region in the right portion of the above figure assuming the mower' intial position was row 5, column 6).

Inputs

The first line of the input contains the number of lawns to be cut. This is followed by one data set for each lawn. The first line of a data set contains four positive integers, the length and width, L and W, followed by row and column number of the initial position of the center of the lawn mower (row 0, column 0 is the upper left square of the lawn). Each of the next L lines contains W marker characters, each of which corresponds to a square on the lawn. The marker character **T** means that there is a tree on the square and the marker character **.** means no tree. To aid readability, each marker character is preceded by a single space. (You may assume that length and width are at most 40.)

Outputs

For each lawn, there should be L lines of output. Each output line will contain W marker characters, each preceded by a single space. This grid of marker characters should be the same as the input lawn description except that the `.` markers shouild be replaced with a C marker character wherever a square of the lawn can be cut by the mower. The output lawns should be separated by one blank line.

Sample Input

```
2
12 11 5 6
. . . . . . . . . . . .
. . . T . . . . . T . .
. . . . . . . . . . . .
. . . . . . . . . . . .
. T . . . . . . . . . .
. . . T . . . . . . . .
. T . . . . . . . . . .
. . . . . . . . . T . .
. . T . . . . . . . T
. . . . . . T . . . .
. . . . . . . . . . . .
. . . . . . T . . . .
11 13 9 1
. . . T . . . . . . . T
. . . . T . . . . . . .
. . . . . . . . . . . .
. . . . . . . . T . . .
. . . . T . . T . . . .
T . . . . . . . . . . .
. . . . . . . . . . . .
. . . . . . . . . . . T
. . . . . . . . T . . .
. . . . . . . T . . . .
. . . T . . T . . . . .
```

Sample Output

```
. . . . C C C C C . .
. . . T C C C C C T .
. . C C C C C C C C C
. . C C C C C C C C C
. T C C C C C C C C C
. . . T C C C C C C C
. T . C C C C C C C C
. . . C C C C C C T .
. . T C C C C C C . T
C C C C C C T . . . .
C C C C C C . . . . .
C C C C C C T . . . .
C C C T . C C C C C C C T
C C C C T C C C C C C C C
C C C C . C C C C C C C C
C C C C . C C C T C C C C
C C C C T . . T C C C C C
T C C C C C C C C C C C C
C C C C C C C C C C C C C
C C C C C C C C C C C C T
C C C C C C C C T C C C C
C C C C C C C T . C C C C
C C C T . . T . . C C C C
```

F - Voting
Input File: VotingIn.txt

When Australians vote in their general elections, they must rank all of the candidates in order of preference. To determine the winner of the election, initially only the first choices are counted. If one candidate received more than 50% of the vote the candidate is elected. If no candidate receives more than 50% of the first place votes, the candidates receiving the lowest number of first place votes are eliminated. Ballots that rank these candidates first are then recounted in favor of their next highest-ranked non-eliminated candidate. This process of eliminating the lowest first place vote-getters and recounting their ballots in favor of their highest-ranked non-eliminated candidate continues until one candidate receives more than 50% of the vote, or until all remaining candidates are tied.

Inputs

The first line will contain the number of elections to consider. This will be followed by one data set per election. The first line of each data set contains the number of candidates in the election, N, followed by the number of votes cast, C. The next N lines contain the names of the candidates, one per line, with the name of candidate 1 on the first line, the name of candidate 2 is on the second line, etc. The next C lines contain the ballots, one per line, each containing the integers between 1 and N in some order. The first integer is the number of the candidate that is a voter's first choice; the second integer is the number of the candidate that is a voter's second choice; etc.

Outputs

There will be one line of output for each election. The line will either contain the name of the candidate who won the election, or the names of the candidates who tied, in candidate number order separated by a space.

Sample inputs and outputs

(See next page)

Sample Inputs

```
3
3 5
Joe Down Under
Mary Poppins
Babe Ruth
1 2 3
2 1 3
2 3 1
1 2 3
3 1 2
4 6
Joe Down Under
Mary Poppins
Alexander the Great
Babe Ruth
1 2 3 4
4 1 3 4
3 1 2 4
2 3 4 1
1 2 4 3
3 4 1 2
5 7
Joe Down Under
Mary Poppins
Babe Ruth
Alexander the Great
Billy Boy
4 2 3 1 5
2 4 3 1 5
2 3 1 5 4
4 2 3 1 5
3 5 4 1 2
1 3 2 4 5
1 3 2 4 5
```

Sample output

```
Joe Down Under
Joe Down Under Alexander the Great
Alexander the Great
```

G - Rats
Input File: RatsIn.txt

A man has purchased N bottles of wine numbered 1 through N, which he plans to serve at his wedding. He has discovered that one of the bottles has been poisoned with a fatal poison that takes 24 hours to act. The poison is fatal to humans and rats. He intends to serve the wine in dishes to several rats, one dish per rat, 24 hours before the wedding to determine which bottle contains the poison. The dishes are labeled 1, 2, 3, etc., and every bottle of wine will be poured into *at least* one dish.

Your task is to determine the *minimum* number of dishes (rats) required to identify the poisoned bottle, and which bottles of wine will be poured into a given dish, D. Assume that the wine from bottle 1 will be poured first into all of its dishes, then wine from bottle 2 will be poured into all of its dishes, etc. In addition, assume that the first dish considered to add wine to is dish 1, then dish 2, then dish 3, etc. (e.g., skip dish 1, add wine to dishes 2 and 3, skip dishes 4 and 5, add wine to dish 6).

Inputs
The first line of input will contain an integer specifying the number of weddings to consider.
Each subsequent line will contain two integers: the number of bottles of wine purchased for a wedding, N, follow by a dish number, D.

Outputs
For each wedding, there will be two lines of output. The first line will contain the minimum number of rats (dishes) required to test the wine for that wedding. The second line will contain the numbers of the wine bottles whose wine would be poured into the D^{th} dish, in wine bottle number order. If the D^{th} dish is empty, output a blank line.

Sample inputs
```
3
4 1
16 3
12 4
```
Sample outputs
```
3
1 3
5
4 5 6 7 12 13 14 15
4
8 9 10 11 12
```