Problem A. Squares

Input file:	a.in
Output file:	a.out
Time limit:	6 seconds
Memory limit:	256 Mebibytes

A long sheet of crossed paper contains a rectangular strip which is N squares wide. The strip is said to be "correctly" coloured if each of its N rows contains several coloured suqares in succession. Given a "correct" colouring, you should find the minimum number of squares to be coloured so that, first, the colouring remains "correct" and, second, there exists a coloured rectangle of prescribed length and width N.

Input

The first line of the input file contains an integer d ($1 \le d \le 100$) — the number of test samples. The first line of each test sample contains integers N ($1 \le N \le 10^5$) — the width of the strip and K ($1 \le k \le 10^9$) — the length of the required rectangle. The next line describes the initial colouring specified by N pairs of integers a_i and b_i — x-coordinates of the left and right sides of the coloured rectangle in the ith row from the top, respectively ($0 \le a_i < b_i \le 10^9$).

Output

For each test sample the output file should contain a single integer — the minimum number of squares to be coloured to obtain a rectangle of the required length.

Example

a.in	a.out
1 3 2 0 3 3 4 5 8	5

Problem B. Cyclic letters

Input file:	b.in
Output file:	b.out
Time limit:	4 seconds
Memory limit:	256 Mebibytes

Given a string of N Latin letters, find the lexicograhpically least string that can be obtained from the given one with the help of the cyclic shift opeation, which consists in moving the last letter in the string to the beginning.

Input

The first line of the input file contains a positive integer d ($1 \le d \le 100$) — the number of test samples.

Each test sample is specified by a line containing a string of 1 to 10^6 lower case Latin letters.

Output

For each test sample the output file should contain the lexicographically least string that can be obtained from the given one in the way mentioned above.

Example

b.in	b.out
3 zyx kkl icpc	xzy kkl cicp

Problem C. Shortest Path (Division 1 Only!)

Input file:	c.in
Output file:	c.out
Time limit:	4 seconds
Memory limit:	256 Mebibytes

Given a convex polyhedron with N faces and two points on its surface, find the shortest path between these points that goes over the polyhedral surface.

Input

The first line of the input file contains a positive integer d ($1 \le d \le 150$) — the number of test samples.

The first line of each test sample contains an integer N — the number of faces of the polyhedron ($4 \le N \le 15$). The faces are described in the following N lines, a line for each face. The description starts with an integer K — the number of vertices of the face ($3 \le K \le N-1$). Then follow triples of the coordinates x, y, z of all K vertices of the face. It is guaranteed that no pair of faces lie in the same plane.

The last line of each test case contains six numbers — x_a, y_a, z_a — the coordinates of the initial point and x_b, y_b, z_b — the coordinates of the terminal point.

All the coordinates in the input file are integers not greater than 1000 in absolute value.

Output

For each test sample the output file should contain the legnth of the shortest path between the given points over the polyhedral surface calculated with an accuracy of 2 decimal digits.

Example

c.in	c.out
1 6 4 0 0 0 0 0 1 0 1 1 0 1 0 4 1 0 0 1 0 1 1 1 1 1 1 0 4 0 0 0 0 1 0 1 1 0 1 0 0 4 0 0 1 0 1 1 1 1 1 1 0 1 4 0 0 0 0 0 1 1 0 1 1 0 0 4 0 1 0 0 1 1 1 1 1 1 1 0 0 0 0 1 1 1	2.24

Problem D. Tournament

Input file:	d.in
Output file:	d.out
Time limit:	4 seconds
Memory limit:	256 Megabytes

In order to carry out the demonstrative tournament it is necessary to select 3 tennis players from N available. Each player has it's own rating, and all N ratings are distinct. A tennis player with higher rating will always defeat a player with lower rating. Local bookmakers don't know anything about these ratings, but they know the final scores of some matches between players. You have to choose 3 players in such a way, so that bookmakers would not be able to predict the result of the match between any two of them.

Input

The first line of input contains a single positive integer number d ($1 \le d \le 50$) - amount of test samples.

The first line of each test sample contains a single number N - amount of tennis players ($3 \le N \le 1000$). Tennis players are numbered from 1 to N in order of filling. Each of the following N lines contains N symbols 0 or 1. 1 in the j-th column of i-th row means that bookmakers know that the i-th player defeated the j-th one, otherwise the symbol is 0. It is guaranteed that the bookmaker's information is consistent.

Output

For each test sample you should output on a separate line word `YES` and 3 player's numbers, if there is a triple of players that satisfy the requirements. In case there are many triples, you may select any of them. In case there are no triples, output `NO`.

Example

*

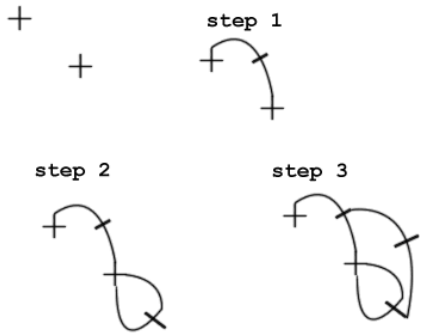
d.in	d.out
2 8 00111111 00011011 00000100 00001011 00000001 00000000 00000000 00000000 6 011111 001111 000010 000001 000000 000000	YES 6 5 7 NO

Problem E. Brussels Sprouts (Division 2 Only!)

Input file:	e.in
Output file:	e.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Brussels sprouts game is a two-player game with following rules. Game starts with N crosses, i.e. spot with four free ends. Each move involves joining two free ends with a curve (not crossing any existing line) and then putting a short stroke across the line to create two new free ends. Players make moves alternately. If player can't make a move, he loses.

For example, first 3 moves for $N = 2$ could be as shown in the following picture.



You're given N, determine player who will win the game. Assume that both players play optimally.

Input

The first and only line contains integer number N — count of crosses ($1 \le N \le 13$).

Output

If first player wins write word "First", and write "Second" otherwise.

Example

e.in	e.out
1	First

Problem F. Puzzle (Division 1 Only!)

Input file:	f.in
Output file:	f.out
Time limit:	15 seconds
Memory limit:	256 Mebibytes

There is a well-known riddle called «puzzle», which has one rectangle divided into $m \times n$ square blocks. Each side of the block can be either smooth or may have either a hollow or a flange. Two blocks may adjoin their sides only when the side of the first block has a hollow, and the side of the second block has a flange. While assembling the puzzle you can revolve the block on it's center horizontally, but you cannot turn it up. Your task is to find for a given set of blocks amount of different ways to assemble the puzzle. Two assemblies are considered identical, if each of mn positions has identically oriented blocks with identical configuration. Note that two assemblies that can be transformed to each other by rotation are still considered different.

Input

The first line of input contains a single natural number d ($1 \le d \le 100$) - amount of test samples.

Each test sample starts with a line containing m and n - puzzle dimensions ($1 \le n \le 6$, $1 \le m \le 5$). m lines follow with block descriptions. Each block description consists of 4 numbers, that define the side type in order of sides bypass: 0 if the side smooth, 1 if there is a hollow, and 2 if there is a flange.

Output

Output for each sample in a separate line a single number - amount of different ways to assemble the puzzle.

Example

*

f.in	f.out
2 3 3 0 0 1 2 0 0 1 2 0 0 1 2 0 0 1 2 0 1 1 2 0 1 1 2 0 1 1 2 0 1 1 2 2 2 2 2 3 3 0 0 1 2 0 0 1 2 0 0 1 2 0 0 1 2 0 1 1 2 0 1 1 2 0 1 1 2 0 1 2 2 2 2 2 2	1 0

Problem G. Sum of numbers

Input file:	g.in
Output file:	g.out
Time limit:	6 seconds
Memory limit:	256 Mebibytes

Two players play a match on a chessboard $m \times n$. Each cell contains a non-negative integer number not exceeding 2^{16}. The rules are very simple. The first player draws a polygon without intersections on the board. Some of the polygon edges follow the cell edges, while others go angularly with angles 45, 135, 225 and 315 to horizontal. Polygon vertices are located in cell's corneres. The second player writes down the sum of numbers located in cells that belong to the polygon (at least half of the cell must belong to polygon). After that the polygon is erased and the process repeats k times. Write a program that will calculate k sums written by the second player.

Input

The first line of input contains a single natural number d ($1 \le d \le 10$) - amount of test samples.

Each test sample starts with a line containing 3 numbers - n - width of board, m - height of board and k - amount of polygons ($1 \le n, m \le 1000$, $1 \le k \le 10^4$).

Each of the following m lines contain n numbers $g_{i,j}$ ($i = 1 \ldots m,\ j = 1 \ldots n,\ 0 \le g_{i,j} \le 2^{16}$). $g_{i,j}$ — number written in the cell with coordinates $i,\ j$.

The following k lines describe the polygons. i-th of these lines describes the polygon written during i-th iteration. Polygon description starts with a number of vertices l ($3 \le l \le 100$). After that l pairs $x,\ y$ follow - coordinates of vertices in consecutive order($0 \le x \le n,\ 0 \le y \le m$).

Output

For each test sample output k numbers, separated by space — sums written by the second player.

Example

*

g.in	g.out
1 7 6 1 1 2 4 1 2 4 0 1 4 1 1 4 1 1 4 2 1 2 4 1 2 4 2 1 4 1 1 3 4 4 4 2 4 1 4 4 2 1 1 1 1 5 6 1 0 1 6 3 6 5 4 3 2 3 0	42

Problem H. Subset

Input file:	h.in
Output file:	h.out
Time limit:	4 seconds
Memory limit:	256 Mebibytes

Find the m-element subset of the ordered set $n = \{1,\ 2,\ \ldots\ n\}$ that comes gth in lexicographic order.

Input

The first line of the input file contains a positive integer d ($1 \le d \le 100$) — the number of test samples.

Each test sample is described by three integers m, n and g ($1 \le m \le n \le 500,\ 1 \le g$). It is guaranteed that there exist at least g different m-element subsets.

Output

For each test sample the output file should contain the subset that comes gth in lexicographic order. The elements of the subset should also come in lexicographic order.

Example

h.in	h.out
3 3 5 4 3 5 2 2 2 1	1 3 4 1 2 4 1 2

Problem I. Number (Division 1 Only!)

Input file:	`i.in`
Output file:	`i.out`
Time limit:	4 seconds
Memory limit:	256 Mebibytes

You are given a number m, a prime number p and a non-negative integer number a smaller than p. You would like to know if there is an integer number n, such that $n^n + m^m = a$ modulo p? If there are several solutions, you may output any of them.

Input

The first line of input contains a single natural number d ($1 \le d \le 300$) - amount of test samples.

Each test sample consists of a single line containing three integer numbers p, a and m ($2 \le p \le 10^9,\ 0 \le a < p,\ 1 \le m \le 20, m < p$). It is guaranteed that p is prime.

Output

For each test sample output in a separate line word `YES` and the number n separated by a single space, in case there is $n < 10^{1000}$ that satisfies the requirements, otherwise output `NO`.

Example

*

i.in	i.out
2 11 3 1 11 8 2	YES 567 YES 2

Problem J. Calendar

Input file:	`j.in`
Output file:	`j.out`
Time limit:	4 seconds
Memory limit:	256 Mebibytes

A new star system has been discovered recently that has k inhabited planets. Planets are numbered from 1 to k in order of their proximity to the start, i.e. the smaller is the number of planet, the smaller is it's distance from the star. Calendars for planets are made the following way: each planet has a prime number p_i - the calendar's «base». Week consists of 7 days that have the same names as on Earth. Month consist of p_i weeks, year consists of p_i months.

There are also leap years. Year is called leap, if it's number divides by $(p_i)^3$, or it divides by p_i, but not by $(p_i)^2$. Unlike at Earth, if the year is leap, then the last week of each month with prime index (months are numbered starting from 1) contains 8 days - after Sunday a second Sunday starts.

Day duration for each planet is coherent with Earth, i.e. the phrase «the same day as on Earth» is defined correctly and has sense. It is known, that the date 1 January of year 2000 corresponds to the first day of first month of first year for all k planets, and that it was Monday on all k planets. Your task is for a given Earth date (day, month and year) to calculate the corresponding day of week for each of k planets.

Recall that an Earth non-leap year has 365 days, 31 days for January, 28 days for February, 31 days for March, 30 days for April, 31 days for May, 30 days for June, 31 days for July, 31 days for August, 30 days for September, 31 days for October, 30 days for November, 31 days for December. Each year that divides by 400 or 4, but not by 100 is leap, i.e. February has 29 days, and so the year itself has 366 days. Also recall that today is Sunday, 27 september, 2009.

Input

The first line of input contains a single positive integer number d ($1 \le d \le 100$) - amount of test samples.

Each test sample contains two lines. The first line contains number k ($1 \le k \le 8$) - amount of planets in the system, followed by k integer numbers that define p_i ($2 \le p_i < p_{i+1} \le 50$) for each planet. All numbers p_i are prime. The second line contains the Earth date in d *mmm* *yyyy* format, where d is the day of month, *mmm* - symbolic description of month (`jan` — January, `feb` — February, `mar` — March, `apr` — April, `may` — May, `jun` — June, `jul` — July, `aug` — August, `sep` — September, `oct` — October, `nov` — November and `dec` — December), *yyyy* — year number ($2000 \le yyyy \le 3000$).

Output

For each test sample output in a separate line $k+1$ descriptions (`mon` — Monday, `tue` — Tuesday, `wed` — Wednesday, `thu` — Thursday, `fri` — Friday, `sat` — Saturday, `sun` — Sunday) for day of week on Earth (first description) and k planets ($i+1$-th description corresponds to i-th planet).

Example

*

j.in	j.out
3 4 5 7 11 13 1 jan 2000 4 2 7 19 31 10 feb 2084 4 11 23 43 47 25 jul 2987	sat mon mon mon mon thu sat thu sat sat wed fri thu fri fri

Problem K. Magic Temple (Division 2 Only!)

Input file:	k.in
Output file:	k.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Long ago Maya tribes erected a magic temple in honor to one of the most respected gods — rain god Chak. The wall number is increased by 1 every year. That was imputed to Chak's powerful magic. The wall was written by priests with all good had happened that year. And since god Chak liked fine forms the form of the temple is a regular polygon.

From time immemorial a legend arised saying that the day, when the god of death Kimi had triangulated the temple in such a way that all the result triangles were isosceles, would become darken by a terrible misfortune. The fearful monster Ctulhu would be awaken, thirdly George Bush junior would put to the vote and Apocalypse would happen.

Your task is to write a program which calculates our death date.

Input

Input file contains two numbers: N and C, where N is the number of temple's walls at the present moment and C is the present year ($3 \le N \le 10^{1000}$, $0 \le C \le 10^{1000}$).

Output

Output file contains just one number — the date of our death.

Examples

k.in	k.out
6 2009	2009
7 2009	2010

Problem L. DNA (Division 2 Only!)

Input file:	`l.in`
Output file:	`l.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

A biologist is experimenting with DNA modification of bacterial colonies being grown in a linear array of culture dishes. By changing the DNA, he is able "program"the bacteria to respond to the population density of the neighboring dishes. Population is measured on a four point scale (from 0 to 3). The DNA information is represented as an array `DNA`, indexed from 0 to 9, of population density values and is interpreted as follows:

- In any given culture dish, let K be the sum of that culture dish's density and the densities of the dish immediately to the left and the dish immediately to the right. Then, by the next day, that dish will have a population density of `DNA[K]`.
- The dish at the far left of the line is considered to have a left neighbor with population density 0.
- The dish at the far right of the line is considered to have a right neighbor with population density 0.

Now, clearly, some DNA programs cause all the bacteria to die off (e.g., [0,0,0,0,0,0,0,0,0,0]). Others result in immediate population explosions (e.g., [3,3,3,3,3,3,3,3,3,3]). The biologist is interested in how some of the less obvious intermediate DNA programs might behave.

Write a program to simulate the culture growth in a line of 40 dishes, assuming that dish 20 (assuming they are numbered from 0 to 39) starts with a population density of 1 and all other dishes start with a population density of 0.

Input

The first line of the input file contains a positive integer d ($1 \le d \le 100$) — the number of datasets.

Each dataset will consist of one DNA program (10 integer values) on one line.

Output

For each dataset, print the densities of the 40 dishes for 30 days, starting with the first day. Each day's printout should occupy one line of 40 characters. Each dish is represented by a single character on that line. Zero population densities are to be printed as the character '.'. Population density 1 will be printed as the character '!'. Population density 2 will be printed as the character 'x'. Population density 3 will be printed as the character 'W'.

Example

1.in

```
1
0 1 2 0 1 3 3 2 3 0
```

1.out

```
..................!.................
.................!!!................
................!x.x!...............
...............!..!..!..............
..............!!!!!!!!!.............
.............!x.......x!............
............!..x.....x..!...........
...........!!!xxx...xxx!!!..........
...........!x.!WW!x.x!WW!.x!........
..........!...!xxW.!.Wxx!...!.......
.........!!!.!.WxW!!!WxW.!.!!!......
.........!x.xx!!WWWW.WWWW!!xx.x!....
........!..!!W!Wx..WWW..xW!W!!..!....
.......!!!!xWWxWWx.W.W.xWWxWWx!!!!...
......!x..!WWWWWWWW.W.WWWWWWWW!..x!..
.....!..x!!x......WW.WW......x!!x..!..
....!!!x.!!.x.....WWWWW.....x.!!.x!!!.
...!x.!..xx.xx....W...W....xx.xx..!.x!..
..!...!!x!!!!!x..........x!!!!!x!!...!.
.!!!.!x!!!...!.x........x.!...!!!x!.!!!
!x.xx.!!.x!.!!.xx.......xx.!!.!x.!!.xx.x
..!!!.xx...xxx.!!x.....x!!.xxx...xx.!!!x
.!x.x.!!x.x!W!.x!.x...x.!x.!W!x.x!!.x.!.
!..!x.x!.!.WW!....xx.xx....!WW.!.!x.x.!!
!!!..!..x!!Wx!!..x!!!!!x..!!xW!!x..!x.xx
x.x!!!!x.!WWW!x!x.!...!.x!x!WWW!.x!..!!!
x!.!..!..!x.xW!W..!!.!!..W!Wx.x!...!!x.x
..x!!!!!!..!WWx!.!xxxxx!.!xWW!..!.!x!.!x
.x.!....x!!!xWW.x.WWWWW.x.WWx!!!!x.!.x..
xx.!!..x.!.!WWWWxWW...WWxWWWW!..!..!.xx.
```