Problem A. Apple

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

An "apple" as a union of one closed polygonal chain and one non-closed polygonal chain starting from the same vertex, where the closed chain has at least 3 vertices, not closed chain has at least 2 vertices, and they don't intersect each other except at the initial vertex.

Given the points and the set of segments connecting these points, determine whether they form an "apple".

Input

The first line contains two integers n and m ($1 \leqslant n, m \leqslant 100$): the number of the points and the number of connections between them respectively. Each of the following m lines contains two integers a_i and b_i ($1 \leqslant a_i < b_i \leqslant n, i = 1, 2, ..., m$), which denote that the points with numbers a_i and b_i are connected by the i-th segment. Each pair of points is connected by at most one segment.

Output

If the given chain is an "apple", print "`Yes`". Otherwise print "`No`".

Examples

input.txt	output.txt
9 9 1 2 2 3 3 5 5 6 6 4 4 7 1 7 7 9 8 9	Yes
5 5 1 2 2 3 1 3 1 4 1 5	No
4 4 1 2 2 3 1 3 1 4	Yes
5 4 1 2 2 3 1 3 1 4	No

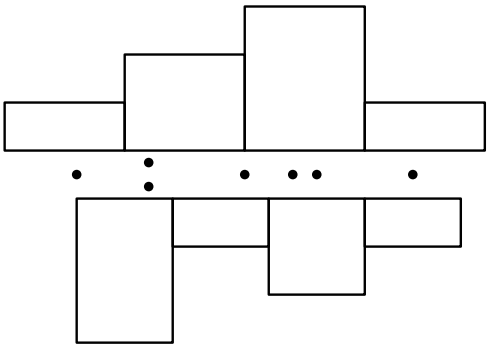
Problem B. Bar charts

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

Let a *data set* be a nondecreasing sequence of integers $A = (a_1, a_2, \cdots, a_n)$.

A bar chart is a graphical representation of a data set: several rectangles of various heights placed next to each other. Namely, the bar chart $H_{k,b,s}(A)$ is a set of k rectangles, the i-th of which has height h_i equal to the number of values from data set A that occur in the interval $[b + (i-1)s, b + is)$, where $1 \leqslant i \leqslant k$. The parameters k, b and s must be chosen so as to cover all the values from the data set, i.e. the following constraints must be held: $b \leqslant a_1$ and $a_n < b + ks$.

It's clear that different parameters k, b and s for the same data set A may produce different bar charts. For example, for the data set $A = (4, 7, 7, 11, 13, 14, 18)$, the bar charts $H_{4,1,5}(A)$ and $H_{4,4,4}(A)$ have the following form:



You are given two bar charts. Your task is to determine whether they can be produced from the same data set or not.

Input

The first two lines of the input describe the first bar chart. The first line contains the parameters k, b and s ($1 \leqslant k, b, s \leqslant 100$). The second line contains k integers, i-th of which is equal to the height of the i-th rectangle h_i ($0 \leqslant h_i \leqslant 100, 1 \leqslant i \leqslant k$).

The next two lines describe the second bar chart in the same format.

Output

Output «Yes» if both of the bar charts can be produced from the same data set. Otherwise output «No».

Examples

input.txt	output.txt
4 1 5 1 2 3 1 4 4 4 3 1 2 1	Yes
4 1 5 1 1 4 1 4 4 4 3 1 2 1	No

Problem C. Construction sets

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

The well-known manufacturer of construction toys OLEG is going through bad times now. All its factories have suddenly stopped producing construction pieces. The cause of the accident is shrouded in mistery, and the company management does its best to find any clues. However, the company needs to release on the market a bunch of a construction sets which apparently must be assembled from construction pieces the company has at the current moment.

There are n types of construction pieces. For a piece of i-th type, the mass m_i and the quantity c_i are known. The company wants to assemble from these pieces as many *pairwise identical* construction sets as possible. The total mass of pieces in every construction set must be no less than $m_{\min}$ and no more than $m_{\max}$. Two construction sets are identical if for every piece type both of the sets have the same number of pieces of this type.

Determine the maximal number of construction sets the company can release.

Input

The first line contains n, $m_{\min}$ and $m_{\max}$ ($1 \leqslant n \leqslant 50$, $1 \leqslant m_{\min} \leqslant m_{\max} \leqslant 10^4$). Each of the following n lines contain two integers m_i and c_i ($1 \leqslant m_i \leqslant 10^4$, $1 \leqslant c_i \leqslant 10^6$, $1 \leqslant i \leqslant n$).

Output

Output one integer, the maximal number of construction sets.

Example

input.txt	output.txt
3 12 13 3 8 4 6 7 9	4

Note

The construction set from the sample can be built in the following way: we get two pieces of type 1 and one piece of type 3. The total mass of the set is $3 + 3 + 7 = 13$, and we can assemble at most 4 such construction sets from the given pile of pieces.

Problem D. Dinner party

Input file:	input.txt
Output file:	output.txt
Time limit:	1 second
Memory limit:	256 megabytes

It is a holiday today and auntie Margaret expects many guests for a dinner party. Specially for the grand feast, uncle Henry brought from his hacienda n square tables and m chairs.

The auntie wants to arrange all the square tables to make one or more big rectangular tables, and to arrange the chairs along the perimeter of each big table. All n tables should be used to serve as much food as possible, and all m chairs should be used so that guests have enough space. Note that the chairs should be placed in all possible slots, so the total perimeter of all big rectangular tables is equal to the number of chairs.

For example, for $n = 10$ and $m = 18$, the square tables can be united into two big tables of size 2×4 and 1×2, and the total perimeter of them is precisely 18. There is another solution for these parameters: 2×3 and 2×2.

Help Margaret to arrange the furniture in the right way.

Input

The first line contains integer T — the number of tests ($1 \leqslant T \leqslant 200$).

Each of the following T lines contains two integers n and m ($1 \leqslant n \leqslant 1\,000$, $1 \leqslant m \leqslant 4\,000$).

Output

For every test output «Yes» if it's possible to arrange the tables in the right way, otherwise output «No». In case of a positive answer, in the next line output integer z, the total number of big tables, and after that ouput z lines containing two integers each: a_i and b_i, sizes of the big tables ($1 \leqslant i \leqslant z$).

In case of many answers, output any of them.

See the sample for a better understanding of the output format.

Example

input.txt	output.txt
4 10 18 6 12 5 20 1 2	Yes 2 1 2 2 4 No Yes 5 1 1 1 1 1 1 1 1 1 1 No

Problem E. Expression on dice

Input file:	`stdin`
Output file:	`stdout`
Time limit:	2 seconds
Memory limit:	256 megabytes
Queries limit:	1 000

This is an interactive problem.

In this problem, you need to construct any true arithmetic statement by using symbols written on die faces. There is an unlimited number of dice of 6 types. The faces of all types of dices follow:

- **Type 1**: `= < > != <= >=`
- **Type 2**: `4 + - ( ( )`
- **Type 3**: `0 / / / 8 +`
- **Type 4**: `2 3 4 5 - )`
- **Type 5**: `+ - * / 1 9`
- **Type 6**: `6 7 + - ( )`

The symbols that can be used are determined as follows. First, you need to choose one die of any type, roll it, and read the symbol. After that, choose the next die of any type, roll it, read the next symbol and so on. Once you decide that you have enough symbols, you need to build a true statement from **all** of the read symbols, satisfying the following grammar:

$$
\begin{aligned}
\langle \text{Statement} \rangle &\longrightarrow \langle \text{Expression} \rangle \langle \text{RelationalOperator} \rangle \langle \text{Expression} \rangle \\
\langle \text{RelationalOperator} \rangle &\longrightarrow \texttt{=} \mid \texttt{<} \mid \texttt{>} \mid \texttt{!=} \mid \texttt{<=} \mid \texttt{>=} \\
\langle \text{Expression} \rangle &\longrightarrow \langle \text{Term} \rangle \mid \langle \text{Expression} \rangle \texttt{+} \langle \text{Term} \rangle \mid \langle \text{Expression} \rangle \texttt{-} \langle \text{Term} \rangle \\
\langle \text{Term} \rangle &\longrightarrow \langle \text{Factor} \rangle \mid \langle \text{Term} \rangle \texttt{*} \langle \text{Factor} \rangle \mid \langle \text{Term} \rangle \texttt{/} \langle \text{Factor} \rangle \\
\langle \text{Factor} \rangle &\longrightarrow \langle \text{Number} \rangle \mid (\langle \text{Expression} \rangle) \\
\langle \text{Number} \rangle &\longrightarrow \langle \text{Digit} \rangle \mid \langle \text{NonZeroDigit} \rangle \langle \text{Digits} \rangle \\
\langle \text{Digits} \rangle &\longrightarrow \langle \text{Digit} \rangle \mid \langle \text{Digits} \rangle \langle \text{Digit} \rangle \\
\langle \text{NonZeroDigit} \rangle &\longrightarrow 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
\langle \text{Digit} \rangle &\longrightarrow 0 \mid \langle \text{NonZeroDigit} \rangle
\end{aligned}
$$

Here "/" means division of **reals**, not integer division. Also, division by 0 is not allowed.

Interaction protocol

Your program should output one of two commands:

1. "`<num>`", where `<num>` is the type of the die to be rolled. For this query the jury program will output a single line containing a symbol.

2. "0 <ans>", where <ans> is your statement.

Don't forget to flush the standard output after printing each line.

Examples

stdout	stdin
2 6 3 4 5 4 6 6 4 2 1 3 4 0 5-7>4-((5+50))	) (+ 5 - 4 (7 5) > 0 5
3 4 1 3 4 5 4 0 2/4!=4/9	/ 2 != / 4 9 4

Problem F. Flight trip

Input file:	stdin
Output file:	stdout
Time limit:	2 seconds
Memory limit:	256 megabytes
Queries limit:	25 000

This is an interactive problem.

Assume that Earth is an ideal sphere whose equator's length is 40 000 km. A position on the planet is defined by latitude φ ($-90^\circ \leqslant \varphi \leqslant 90^\circ$) and longitude λ ($-180^\circ \leqslant \lambda < 180^\circ$). Moving to the north increases the latitude and moving to the east inreases the logitude.

Its surface is divided into 24 time zones, where the i-th time zone ($i = -11, -10, ..., 11$) covers the interval of $[15i - 7.5, 15i + 7.5)$ degrees of longitude, and the 12-th time zone covers the union of the intervals of $[-180, -172.5)$ and $[172.5, 180)$ degrees of longitude. The local time in time zone i is i hours later than the local time in time zone 0. The North and South Poles have time zone 0.

You are in a helicopter (that can turn as fast as 1 440°/`min` and goes at 3`km`/`min`) located at an unknown point near the surface and moving in an unknown direction. Your goal is to reach a given location on the planet using only information about the local time in the territories you fly over.

Interaction protocol

First the jury program will output two real numbers with 10 decimal places: the latitude and longitude of the target location. It will be enough to approach the point as near as 10^{-5} km.

In the next line, the jury program prints the local time in the following format: "`<day of the week> <hours>:<minute>`", where the hours and minutes contain exactly two digits, and the day of the week is one of the following: "`mon`", "`tue`", "`wed`", "`thu`", "`fri`", "`sat`", "`sun`". Each test always starts at Monday, `00:00` local time of your starting position.

After that, your program should make queries consisting of two real numbers a and d ($-360 < a < 360$, $0 \leqslant d < 40\,000$), which denote that first you will rotate a° counterclockwise, and then fly straight for a distance of d km. These numbers should be a decimal fraction with no more than 12 places after the decimal point. For each query, the jury program will output either a string containing the local time in the format described above, or the line "`---00:00`", which means you have reached the target location with necessary accuracy and your program should quit.

Don't forget to flush the standard output after printing each line.

Examples

stdout	stdin
	-15.5837712391 -118.7986557214
	mon 00:00
0 1000	mon 04:33
90 0	mon 04:33
10.101010 25252.52525252	sat 13:50
123 4567.89	--- 00:00

Problem G. Glasses with solutions

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

There are n glasses with salt solutions. For each solution, its total mass and the mass of the salt contained in it are given. Write a program that finds the number of non-empty subsets of these n glasses such that if all the glasses in the subset are poured into an empty glass, the resulting solution will have the given salt mass fraction.

Input

The first line contains three integers n, a, b ($2 \leqslant n \leqslant 35$, $0 \leqslant a \leqslant 10\,000$, $\max(a, 1) \leqslant b \leqslant 10\,000$, $\mathrm{GCD}(a, b) = 1$): the number of glasses with solutions, and the numerator and denominator of the fraction $\frac{a}{b}$ denoting the desired mass fraction of salt.

The following n lines contain two integers m_i and t_i ($0 \leqslant m_i \leqslant 10\,000$, $\max(m_i, 1) \leqslant t_i \leqslant 10\,000$): the mass of salt and the total mass of i-th solution ($i = 1, 2, ..., n$).

Output

Output a single integer: the number of ways to obtain a solution with salt mass fraction equal to $\frac{a}{b}$.

Examples

input.txt	output.txt
5 1 2 1 2 1 2 1 2 1 2 1 4	15
2 0 1 0 1 1 1	1

Problem H. Hamburgers

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	3 seconds
Memory limit:	256 megabytes

There are several cafes in the city that provide a huge variety of different hamburgers.

Each hamburger and the preferences of each person in this problem will be characterized by a list of additional ingredients for hamburgers. We say that a person likes a hamburger if the hamburger has all the person's favorite ingredients. A person will be satisfied by a visit to a cafe if its menu has a hamburger that they like.

Write a program that, given a company of friends, will suggest a cafe such that most of the group will be satisfied by a visit there.

Input

The first line contains one integer n ($1 \leqslant n \leqslant 1\,000$): number of companies of friends.

Then n data blocks that describe each group follow. The first line of each block contains a single non-negative integer k_i: the number of people in the i-th group ($i = 1, 2, ..., n$). Next k_i lines contain a non-empty string that describes preferences of a particular person in this group. Each such string consists of no more than 6 different lowercase Latin letters. Each letter represents one of the 26 possible additional ingredients.

The sum of all k_i does not exceed 50 000.

Next line contains one integer m ($1 \leqslant m \leqslant 1\,000$): the number of cafes. Then m data blocks that describe each cafe follow. The first line of each block contains one non-negative integer l_j: the number of hamburgers on the menu of the j-th cafe ($j = 1, 2, ..., m$). Next l_j lines contain non-empty strings of up to 6 characters each that denote additional ingredients of hamburgers, in the format described above.

The sum of all l_j does not exceed 50 000.

Output

For each group output a single line with one integer from 1 to m: number of the best cafe for the group. If there are several such numbers, output the **minimal** of them.

Examples

input.txt	output.txt
5 1 a 3 a ba cb 3 vba d ba 3 ca da da 3 as ba af 2 4 abc abcd a a 2 abcdef abcdev	1 1 2 1 2

Problem I. Intricate path

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

In this task you are asked to create a testing area with obstacles for the robot, which must enter the area through one of its corners, collide with its interior exactly k times, and then exit it.

The robot has a program that consists of exactly k commands that denote either a left or a right turn. The robot can only move forward, but after collision, it executes the next command of its program and performs either a left or a right turn.

Formally, the testing area is a rectangular grid where the robot and the obstacles occupy exactly one square and where the robot moves only in a direction parallel to one of its sides. The robot collides with an obstacle if and only if the obstacle is located next towards the robot's direction of movement.

Input

The first line contains string s ($1 \leqslant |s| \leqslant 50$): the program for the robot. This line can only contain characters "`L`" and "`R`" that are the commands for turning left and right respectively.

Output

Choose the size of the testing area n ($1 \leqslant n \leqslant 50$) and m ($1 \leqslant m \leqslant 50$) and output n lines containing m characters «`@`», «`.`», «`#`», which denote the following:

1. Character "`.`" denotes an empty cell;
2. Character "`#`" denotes a cell with an obstacle;
3. Character "`@`" denotes the initial position of the robot, which faces right. This character must occur exactly once and must be either in **first** or **last** row of the **first** column.

Examples

input.txt	output.txt
L	@#
LLLLLL	###### .....# ####.# #....# ###.## ###.## @...##
RLLRRLLRRLLRRLLRRLLRRLLL	@## ..# ###
LRRLLLLRRRLRRRRR	.......#... ...#......# .#......#.. ......#.... ..#........ ....#... ....#. @.#..#.....

Problem J. Jumps through the Hyperspace

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

Year 2333. Since the discovery of the Hyperspace, interplanetary journeys became ordinary business. By this time, the humanity has already expanded to n different planets. Each of the planets has a hyperport which allows to travel between planets via hyperjumps.

Due to a peculiarity in the curvature of spacetime, the duration of a hyperjump is not a constant value. It depends on the beginning time of the hyperjump and its type. Namely, for hyperjump of type i and the beginning time s, the total travel time is $T = ((a_i s + b_i) \mod c_i) + d_i$, where a_i, b_i, c_i and d_i are parameters of the hyperjump of i-th type.

You are on the planet with number 1, and you want to get to the planet with number n. What is the minimum time you need to spend?

Input

The first line contains three integers n, m and s — the number of planets, the number of hyperjump types and the current time ($2 \leqslant n \leqslant 2000$, $1 \leqslant m \leqslant 50$, $1 \leqslant s \leqslant 2000$).

Next, there are m lines, i-th of which contains symbol z_i — the identifier of the hyperjump of the i-th type (the lowercase or the uppercase latin letter), and four integers a_i, b_i, c_i and d_i — parameters of the hyperjump of the i-th type ($1 \leqslant a_i, b_i, c_i, d_i \leqslant 2000$). Identifiers of the hyperjump types are pairwise distinct.

Each of the following n lines contains precisely n symbols. The j-th symbol in the i-th line describes a type of the hyperjump that allows to travel from the i-th planet onto the j-th one. If there is no hyperjump from i-th to j-th planet, there will be the symbol «.» instead.

Output

Output one integer — the minumum time you need to travel from the planet with number 1 to the planet with number n.

If it's impossible to travel from planet 1 to planet n via a sequence of one or several hyperjumps, output -1.

Example

input.txt	output.txt
3 2 1 A 1 1 5 1 a 2 2 7 1 .A. ..a a..	6

Note

Clarification to the sample: the best way to travel from planet 1 to planet 2 is to jump immediately at time 1 and to arrive at time 4; while jumping from planet 2 onto planet 3, you should wait two units of time first and then do the hyperjump by one unit of time and get to the destination at time 7.

Problem K. King's island

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

King of Byteland wants to create an artifical island with the following property: it must be a simple non-selfintersecting polygon with n vertices; all vertices must have integer coordinates; no three consecutive vertices must lie on one line; lengths of all sides walls must be integers, and no wall must be parallel to coordinate axes. Given n, build such an island with absolute coordinates no more than 10^4.

Input

The only line contains integer n ($3 \leqslant n \leqslant 30$).

Output

Output n lines, each containing 2 integers x_i, y_i ($-10\,000 \leqslant x_i, y_i \leqslant 10\,000$), coordinates of the i-th vertex. The vertices must be ordered either clockwise or counter-clockwise.

Examples

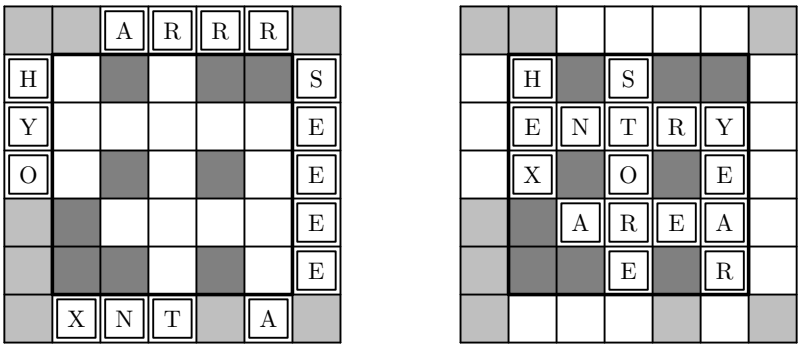
input.txt	output.txt
3	0 0 4 3 -20 21
4	0 3 4 0 0 -3 -4 0

Problem L. Lexica

Input file:	input.txt
Output file:	output.txt
Time limit:	2 seconds
Memory limit:	256 megabytes

A Lexica puzzle is a square divided into $n \times n$ calls. Some cells are active, others are not, and the active cells form a crossword grid. There are a number of tiles along the borders of the square, each bearing a single letter. The tiles placed along the left and the right borders can be moved onto active cells in the same row; the tiles along the top and the bottom border can be moved onto active cells in the same column. It is not allowed to place two or more tiles onto one active cell. The number of tiles is equal to the number of active cells. The goal of the game is to move all the tiles onto the active cells to get a valid filling of the crossword grid. This state of the game we call its solution.

For example, the initial state of the puzzle and its solution are shown on the following picture:



You are given a Lexica puzzle. It's known that for any two tiles, the first of which can be moved horizontally and the second one can be moved vertically, the letters on them are different. Determine the number of distinct solutions. Two solutions are distinct if there is an active cell such that the tiles on this cell in the corresponding solutions bear different letters.

Input

The first line contains integer n ($4 \leqslant n \leqslant 13$). The following $(n + 2)$ lines describe the puzzle. Each of them contains precisely $(n + 2)$ symbols. The active cells are denoted by «.», the tiles are denoted by uppercase latin letters, all other cells are denoted by «#». See the sample for a better understanding.

Output

Output one integer, the number of different solutions.

Example

input.txt	output.txt
5 ##ARRR# H.#.##S Y.....E O.#.#.E ##....E ###.#.E #XNT#A#	768

Problem M. Maximal paths

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given a complete binary tree containing n vertices. The vertices are labeled by integers 1 through n. For every vertex with number $i > 1$ its parent is the vertex with number $\lfloor i/2 \rfloor$. A digit 1 through 9 is written on each edge of the tree.

For every pair of vertices u and v, let $N(u, v)$ be a number that can be read from the edges while moving from u to v along the shortest path $p(u, v)$. For $u = v$ $N(u, v) = 0$.

For vertex v, let's define a set of paths *hung* on v $H(v)$ as the set of paths that contain vertex v but don't contain its parent (for vertex 1 all paths containing it are considered to be hung on it).

Let $M(v)$ be a path $p(u, w)$ hung on vertex v ($p(u, w) \in H(v)$) with maximal value $N(u, w)$. We call this path *the maximal*.

It this problem you have to find the maximal path $p(u, w) = M(v)$ for every vertex v, take the sum of the corresponding values $N(u, w)$ over all v and output it.

Input

The only line contains two integers n and $seed$ ($1 \leqslant n \leqslant 10^7$, $0 \leqslant seed < 2^{31}$).

Digits written on the edges can be generated with a linear congruential generator. First, sequence r should be generated in the following way:

$$r_1 = seed,$$
$$r_i = (1103515245 r_{i-1} + 12345) \mod 2^{31}, 2 \leqslant i \leqslant n.$$

After that, for each edge that connects vertices i and $\lfloor i/2 \rfloor$, the digit written on that edge can be calculated as $a_i = ((r_i >> 16) \mod 9) + 1$, $2 \leqslant i \leqslant n$.

Output

Output one integer, the sum.

Example

input.txt	output.txt
10 2017	102915

Note

The tree that corresponds to the sample is shown below. Values $N(u, w)$ for the maximal paths $M(v)$ are: 95796, 6974, 98, 41, 6, 0, 0, 0, 0, 0. The sum of those numbers is 102915.