Problem A. Decorating the Tree (Division 1 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	512 mebibytes

Every year Vasya has winter holidays from January, 1 for a week or more. He considers it as the most boring time of the year. His university is closed, exams still not started, and TV shows only the oldest movies. Luckily, Vasya has a New Year tree at home and he constantly invents more complicated ways to decorate his tree.

Vasya wants to use n sets and put them to n different places of his tree. More formally, he represnted his New Year tree as a graph theory tree with n nodes. Vasya wants to put exactly one decoration set to every node of the tree. Decoration set for node v_i must be of size c_i.

Vasya can buy decoration sets in the shop next to his house. Any decoration set is defined by two parameters: its *size* and its *base*. Such a set consists of *size* balls with radii *base*, *base*+1, ..., *base*+*size*−1. The set costs $base + size - 1$ rubles, i. e. the price only depends on the radius of the largest ball. For any given positive integers *size* and *base* the shop can quickly deliver any amount of decoration sets.

Vasya has a special requirement to the decoration. He does not want two decoration sets on adjacent nodes to share a ball of the same size. At the same time, he has nothing against having same balls or even same decoration sets on nodes which are not connected with an edge.

Please note, that Vasya has to buy exactly n decoration sets. No decoration set can be split, and no two decoration sets can be merged together.

Please help Vasya to decorate his tree at a minimum cost.

Input

First line of the input contains a single integer, n ($1 \le n \le 2000$). Next line contains n integers, c_i describes required size of decoration set for i-th node ($1 \le c_i \le 10^9$). Next $n-1$ lines describe edges. Each line has format "`u v`" ($0 \le u, v \le n-1$) and means that nodes u and v are connected with the edge.

Output

Print minimum decorating cost on the single line.

Examples

standard input	standard output
5 5 1 5 1 2 0 1 1 2 2 3 2 4	17
2 1 2 0 1	4

Problem B. Double Towers of Hanoi

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

This problem is a modification of the well-known "Tower of Hanoi" problem. There are three rods and $2n$ disks of n different sizes, two of each size. Let the disks be numbered by integers from 1 to $2n$ in non-decreasing order of size. Thus, the disks with numbers $2i - 1$ and $2i$ have equal sizes. The disks can slide onto any rod. You can move only one disk from the top of any rod to the top of any other rod at a time. A larger disk can never be put over a smaller one.

Initially all the disks are on the first rod in the decreasing order of their numbers (counting from bottom to top). Finally they all should be posed to the second rod in a certain prescribed order. Your task is to find the minimal number of moves to get the final configuration from the initial one. Note that the disks with equal sizes but different numbers are considered to be different.

Input

The first line contains the single integer n ($1 \le n \le 2000$). The second line describes the final order of the disks on the second rod. It contains $2n$ numbers in order from bottom to top. It is guaranteed that each of $2n$ disks appears in the given sequence exactly once. The final configuration is correct, i.e. there is no larger disk lying on a smaller one.

Output

Output the required minimal number of moves.

Examples

standard input	standard output
1 2 1	3

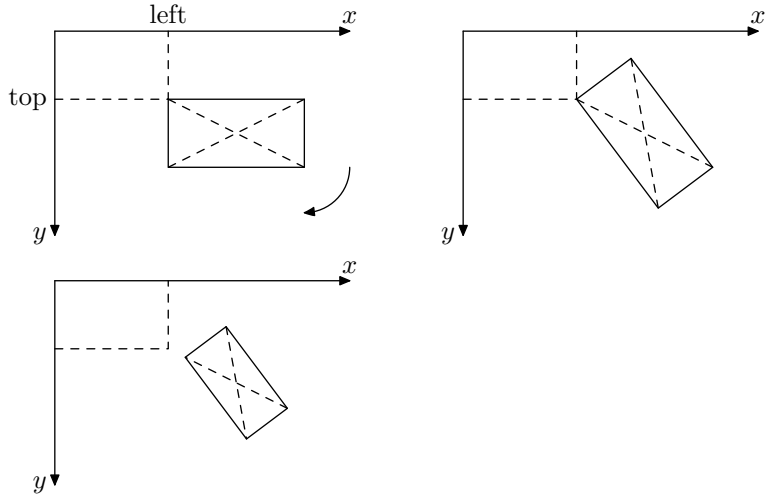
Problem C. Drawing with CSS

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Vasya is developing a browser online game and he wants to draw some objects in a browser.

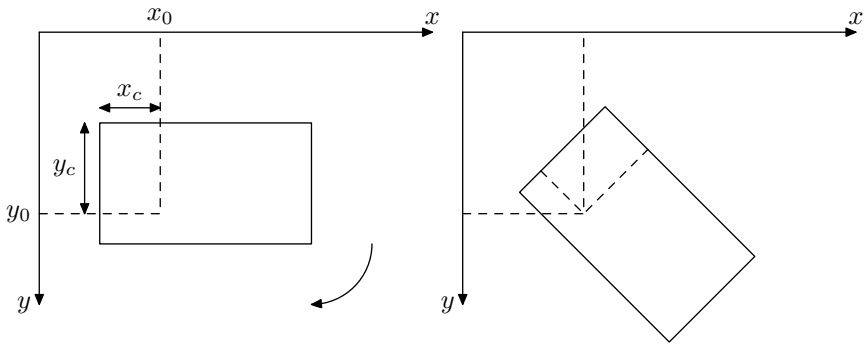
Drawing in a browser uses CSS rules those specify the position of an object, the size, the rotation and the scale. For simplicity in this task, all objects are rectangles. The coordinate system is defined in a browser so that the X-axis goes from the left to the right, and the Y-axis goes from the top to the bottom. A CSS rule specifies the position of a rectangle as a pair of coordinates of the top-left corner $(left, top)$, where $left$ means the X-coordinate, and top means the Y-coordinate. To rotate a rectangle a rule must specify the angle α in degrees to which the object will be rotated about its center in the clockwise direction. CSS rules support scaling of objects as well. To scale an object one should use CSS statement with width/height of the object image.

The order of a browser's actions is the following: the positioning of an rectangle according to the coordinates of the top-left corner, then rotation about the center, and the last action is scaling about the center.



Vasya's rectangles are not so simple, and it turned out that the images in them are such that they shouldn't be rotated and scaled about the center of the rectangle, but about the another point, which we denote as *pivot*.

Vasya defines the pivot as (x_c, y_c) for every rectangle that he has, where x_c is offset by the X-coordinate from the top-left corner, and y_c is offset by Y-coordinate. Now he wants to draw a rectangle so that the pivot gets exactly to the point (x_0, y_0) in the browser. After that, a rectangle is rotated by clockwise angle *alpha* about the pivot and scaled about the pivot by the factor s.



Your task is to generate CSS rule to place rectangles as Vasya wants.

Input

The first line of the input contains integer n ($1 \le n \le 10^4$), the number of rectangles. The following n lines contain descriptions of rectangles, one description per line. A description consists of seven integers w, h, x_c, y_c, x_0, y_0, α ($1 \le w \le 1000$, $1 \le h \le 1000$, $0 \le x_c \le w$, $0 \le y_c \le h$, $0 \le x_0 \le 1000$, $0 \le y_0 \le 1000$, $0 \le \alpha < 360$) and one real number s ($0 < s \le 10$) which is given with exactly one digit after the decimal point, where:

- w means the width of a rectangle, h is the height,
- the point (x_c, y_c) defines the position of the pivot in a rectangle,
- the point (x_0, y_0) defines the position of the pivot in a browser,
- α is the clockwise angle for rotation,
- s is the scale.

Output

Print on a single line a CSS rule which places the rectangle as Vasya needs in the format "`.itemN { width: W; height: H; left: L; top: T; transform: rotate(Adeg); }`", where:

- N means the index of a rectangle (all rectangles are numbered from 1 in the order as they are given in the input);
- W means the width of a rectangle painted in a browser;
- H means the height of a rectangle painted in a browser;
- (L, T) is the position of the top-left corner of a rectangle before rotating;
- A is the clockwise angle for rotating about the center of a rectangle.

Strictly follow the output format. The example have extra line-breaks because of limited table width. The values W, H, L, T, A must be printed with exactly one digit after the decimal point. The value A must satisfy the condition $0 \le A < 360$. Every CSS rule should be printed on exactly one line. All attributes are required and you can't remove any of them.

Examples

standard input
2 200 100 50 75 400 200 45 0.5 200 100 50 75 400 200 90 2.0
standard output
.item1 { width: 100.0px; height: 50.0px; left: 376.5px; top: 183.8px; transform: rotate(45.0deg); } .item2 { width: 400.0px; height: 200.0px; left: 250.0px; top: 200.0px; transform: rotate(90.0deg); }

Problem D. Olympic Games in Berland (Division 1 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

The International Olympic Committee has finally officially stated that the following Olympic Games will be held in one of the cities of Berland!

Naturally, the country isn't prepared for the Olympics, but there is plenty of time. For example, all roads in Berland are unpaved, but according to the IOC rules, the country of the Olympics must have all cities connected by high speed modern highways.

Berland has n cities, some pairs of the cities are connected by two-way unpaved roads. For each road, we know the time t_j we need to turn it into a high speed modern highway. All possible expences have been accounted for, so the top priority is to prepare the country for the Olympics as quickly as possible. The only possible way to build a highway is to upgrage an unpaved road.

The Olympic Committee of Berland decided to connect some cities by highways so that it was possible to get from each city to any other one using only highways. Note that you need to make the minimum possible number of highways. There are a lot of groups of workers involved, so the road works are conducted simultaneously. Thus, the road works will end in time equal to the maximum t_j among all roads with construction works.

The City of the Olympics hasn't been yet defined. However, the Olympic Committe of Berland has already decided that to minimize extra traffic, there should be exactly one highway attached to this city.

The analytical department is facing the complex task of planning the works. For each city i find the minimum time for road works considering that city i is the City of the Olympics.

Input

The input contains one or multiple sets of test data separated by single line breaks.

Each set starts with a line that contains two integers n and m ($2 \le n \le 4 \cdot 10^5$, $0 \le m \le 4 \cdot 10^5$), where n is the number of cities in Berland and m is the number of unpaved roads. All roads are bidirectional, each pair of cities has at most one road between them. Then follow m lines that describe the roads. Line j contains three integers x_j, y_j and t_j ($1 \le x_j, y_j \le n$) and shows that the j-th road connects cities x_j and y_j ($x_j \ne y_j$), and it takes t_j ($1 \le t_j \le 10^9$) time units to make a highway instead of this road.

It is guaranteed that the total number of roads for all test data doesn't exceed $4 \cdot 10^5$, and the number of roads also doesn't exceed $4 \cdot 10^5$.

Output

For each set of test data print sequence $T_1, T_2, \ldots, T_n$, where T_i is the minimum possible time of ending the road works provided that the Olympiad will be held in city i. Print $T_i = -1$ if it is impossible to meet the requirements while conducting the Olympiad in city i.

Follow the format of the sample from the input.

Examples

standard input	standard output
4 6 1 2 2 2 3 3 3 1 1 3 4 1 4 1 2 4 2 10 3 3 1 2 1 2 3 1 3 1 2 3 2 1 2 2 2 3 4	Case 1: 3 2 2 2 Case 2: 1 2 1 Case 3: 4 -1 4

Problem E. Polycarpia insolitus

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Polycarpus is a well-known microbiologist, and now he breeds new varieties of bacteria. He has n bacteria of the type *polycarpia ordinarius* in his laboratory. For each bacterium, he knows its DNA sequence. He wants to produce new type of bacteria — *polycarpia insolitus*. Polycarp can obtain one new bacterium by placing two bacteria *polycarpia ordinarius* into one test-tube and applying specific chemical reaction. Two parental bacteria *polycarpia ordinarius* disappear during this process.

It is easy to see that Polycarpus can produce at most $\lfloor \frac{n}{2} \rfloor$ new bacteria. He knows that properties of newly created bacterium depend on the DNA sequences of two parental bacteria it was produced from. His latest researh shows that one of the most important properties, *viability*, is equal to the length of the longest common prefix of DNA sequences of the parental bacteria. Help him to combine *polycarpia ordinarius* in pairs in order to produce $\lfloor \frac{n}{2} \rfloor$ new bacteria with maximum total *viability*.

Input

The first line contains integer n ($2 \le n \le 10^5$). Each of the following n lines contains one DNA sequence of *polycarpia ordinarius* — non-empty string of letters `'A'`, `'C'`, `'G'` and `'T'`. Total length of all given sequences does not exceed 10^6.

Output

In the first line print the maximum total viability that can be achieved. Then print $\lfloor \frac{n}{2} \rfloor$ lines. Each line should contain two integers from the range $[1, n]$ — the numbers of parental bacteria that form a pair. Bacteria of the type *polycarpia ordinarius* are numbered as their DNA sequences are given in the input.

Examples

standard input	standard output
4 ACAT CAG CC ACGT	3 1 4 2 3
3 AC CG AGC	1 1 3

Problem F. Berland-Strike

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

A brand new Berland-Strike version 1.7 is released today! All Berland gamers rejoice and anticipate changes in the gameplay. And there are some! Unexpectedly for everyone game developers added puzzle elements into that classical shooter. Specifically, now if berrorists plant a bomb, counter-berrorists need to defuse it by solving a mini-puzzle.

The planted bomb has a sequence of n bridges attached. Each bridge has two contacts with which it is attached to the bomb. A contact is characterized by an integer number. Different contacts might be characterized by the same number, including if they belong to the same bridge. The first contact of the i-th bridge is numbered with x_i and the second — with y_i.

To solve the puzzle player can perform two actions. The first is reversing a bridge. The reverse action reverses contacts of the bridge, i.e. its first contact becomes its second one and its second contact becomes its first one. The second action is to swap any two bridges in the sequence. Player may reverse and swap any number of bridges. Each bridge could be reversed and swapped with other bridges many times.

The aim of the puzzle is to order the sequence of bridges so that the first contacts of bridges in the sequence are ordered in non-descending order while the second contacts must be ordered in non-ascending order.

For instance, assume that there are 4 bridges: $(1,5)$, $(7,1)$, $(3,8)$, $(5,6)$. After player reverses pairs 2, 3 and 4 the sequence looks as follows: $(1,5)$, $(1,7)$, $(8,3)$, $(6,5)$. Then player swaps pairs 1 and 2: $(1,7)$, $(1,5)$, $(8,3)$, $(6,5)$. And finally player swaps pairs 3 and 4: $(1,7)$, $(1,5)$, $(6,5)$, $(8,3)$. And that essentially solves the puzzle as the resulting sequence of the first contacts is non-descending: $(1 \le 1 \le 6 \le 8)$ and the sequence of the second contacts is non-ascending: $7 \ge 5 \ge 5 \ge 3$.

Your task is to write a program which helps solving the puzzle. The bomb has been planted!

Input

The first line of the input contains positive integer T — number of test cases in the input. Then follow T test cases.

The first line of each test case contains positive integer n $(1 \le n \le 2 \cdot 10^5)$ — number of bridges in the test case. Each of the next n lines contains two integers x_i and y_i $(0 \le x_i, y_i \le 10^9)$.

Total number of bridges in all tests cases does not exceed $2 \cdot 10^5$.

Output

Output answer for each test case separately. On the first line print "`yes`" (without quotes) if there is a solution for the test case or "`no`" otherwise. If solution exists, on the next n lines print bridges in the required order. If there are multiple solutions print any of them.

Examples

standard input	standard output
2 4 1 5 7 1 3 8 5 6 2 100 100 201 201	yes 1 7 1 5 6 5 8 3 no

Problem G. Palette (Division 1 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

One not so famous artist Palevich is trying to use up-to-date technologies in his paintings. To make the process of creating his masterpieces easier, he decided to create a palette which contains all combinations of basic colors. In case you are new to this, there are exactly n basic colors. In this problem we will denote them by capital English letters `'A'`, `'B'`, `'C'`, ..., n-th letter of the alphabet.

He has already decided that his palette will have the form of table $r \times c$, $rc = 2^n$. Each cell of the table should contain unique combination of basic colors. By combination we mean a subset of colors. So there are 2^n combinations in total, including one special combination of zero colors.

Before making a real palette, Palevich is going to see how it will look. He will draw it in his favourite graphics editor. The editor provides some simple tools that Palevich will use to draw the palette. First, he creates n layers, one layer per basic color. In the first layer, he selects simple (without self-crossings and self-intersection) polygon. The edges of the polygon should be parts of grid lines of the table. Then he fills all cells of the table that are inside the selected polygon with color `'A'`, creating solid connected figure of the color 'A'. He repeats this process for every other color (in a separate layer), and then merges the layers. This causes colors to mix and create different combinations.

Consider the following example: there are three basic colors `'A'`, `'B'` and `'C'`, and Palevich wants to create palette in 2×4 table. First, he creates three layers and paints them like in the picture.

A	
A	
A	
A	

B	B
B	B

C	C
C	C

Then he merges all layers into one, producing the following palette. It is easy to see that each cell of the table contains unique combination of colors.

A	
A,C	C
A,B,C	B,C
A,B	B

As you see, Palevich has managed to create 2×4 palette, and even a 4×4 one, but he is completely unable to create larger palettes. Your task is to help him.

Input

Input contains two integer numbers r and c ($4 \le r, c \le 256$, $rc = 2^n$ for some integer n).

Output

Print n layers, the first one should correspond to the color `'A'`, the second layer — to the color `'B'`, and so on. Print each layer as r lines of c characters each. Print the cells that are inside the filled polygon as letter of corresponding color, all other cells print as `'.'` (dot). Print a blank line between consecutive layers. If there are multiple solutions, output any of them.

Examples

standard input	standard output
4 4	AAAA AAAA .... BBB. B... BBBB .... .CCC .C.. CC.. CC.. .D.. DDD. D.D. D.D.

Problem H. Polycarpus VS Dequeue (Division 1 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	12 seconds
Memory limit:	512 mebibytes

This is an interactive problem, so don't forget to perform a `flush` *operation after each line of output. As the problem is interactive, you can process another query only after you've printed the answer to the previous one.*

During the IT lessons, Polycarpus studied many data structures: stacks, queues, dequeues and even heaps. It's time to have the exam!

Polycarpus got a question on dequeues and the boy answered the question brilliantly. He began from a beautiful phrase: "In computer science, a double-ended queue (dequeue, often abbreviated to deque, pronounced deck) is an abstract data type that generalizes a queue, for which elements can be added to or removed from either the front (head) or back (tail). It is also often called a head-tail linked list."

Now it's time to do the practical part of the question. The teacher is sure that if a student knows a data structure well, then he can emulate its behavior in his mind without any problems.

Now Polycarpus is standing at the whiteboard divided into 10^9 lines. They are numbered from 1 to 10^9 from top to bottom. Each line is either empty or it contains a dequeue operation. Here is a list of possible operations:

- «`push_front` x» — add element x to the beginning of the dequeue,
- «`push_back` x» — add element x to the end of the dequeue,
- «`pop_front`» — remove an element from the beginning of the dequeue,
- «`pop_back`» — remove an element from the end of the dequeue.

At the beginning of the practical task, Polycarpus is standing in front of the empty whiteboard and the teacher checks his knowledge by saying the commands of type:

- "Let's write the following operation to the line i ..." (names a specific operation),
- "Now let's remove an operation from the line i",
- "Tell me, what elements will be on the ends of the dequeue, if starting from an empty dequeue, we execute all lines, on by one, from the first to the i-th one?".

Thus, the task for Polycarpus isn't that easy after all! The teacher has a luxuriant imagination, so he names the commands very quickly and they go in an unpredictable order.

Polycarpus couldn't cope with the given task but he volunteered to make a program that will process the queries of the three types. Having written such program, he got the mark he wanted, "A".

We ask you to repeat Polycarpus' achievement and implement the program that processes queries.

Input

The first line of the input contains integer n ($1 \le n \le 4 \cdot 10^5$) — the number of the queries. Then follow n lines of one on the following types:

- "i *OPERATION*" — fill the line i with operation "*OPERATION*",
- "i `-`" — remove an operation from the i-th line,
- "i `?`" — name the dequeue's elements on the end after executing the operations from top to bottom by the moment you stop after processing the line i.

The teacher always maintain correct state of the dequeue, that is:

- if you need to fill the line i, then it's empty at that moment,
- if you need to remove an operation from the line i, then this line contains an operation at that moment,
- after each query, the sequential execution of all operations from the first to the last line doesn't call `pop_front`/`pop_back` from an empty dequeue.

Values x for operations "`push_front` x" and "`push_back` x" are integers from 1 to 10^9, inclusive.

As the problem is interactive, you can read a query only after you've printed the output to the previous one.

Output

After you process each query, print a line with two numbers. Print line "`0 0`" after you process queries that add/remove operations. Print line "`front back`" after you process a request of type '`?`', where `front` is the value from the begin of the dequeue and `back` is the value from its end. Print "`-1 -1`" if the dequeue is empty.

Don't forget to execute the `flush` operation after you've printed each line of the output.

Examples

standard input	standard output
9 1 push_back 10 2 push_front 20 2 ? 1 - 2 ? 1 push_back 30 3 pop_front 3 ? 2 ?	0 0 0 0 20 10 0 0 20 20 0 0 0 0 30 30 20 30
11 20 push_back 4 20 ? 40 pop_back 40 ? 30 push_back 3 30 ? 10 push_front 1 10 ? 20 ? 30 ? 40 ?	0 0 4 4 0 0 -1 -1 0 0 4 3 0 0 1 1 1 4 1 3 1 4

Note

Because of poor performance of interactive queries testing, input data will be written in the solution input stream by blocks. The total number of blocks does not exceed 1000. Each block contains one or more queries. We do not specify the method how queries are divided into blocks.

Problem I. Presents (Division 1 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

The day care center has a New Year party. The children dance around the New Year Tree and Santa Claus is going to give out presents. Santa Claus is worrying about something: he may not have enough presents for everybody.

Let's assume that the day care has n children numbered with integers from 0 to $n - 1$. Recently, they are standing in a circle in the order they are indexed: each i-th child is followed by the $(i + 1)$-th one ($0 \le i \le n - 2$), and the $(n - 1)$-th child is followed by the zeroth one. Santa Claus has m presents ($1 \le m \le n$) which he will give to only the children who's been good throughout the year. As Santa Claus is actually the day care center caregiver in Santa's clothes, he knows the numbers a_1, a_2, ..., a_m of the kids who's been good. By coincidence (if it is a coincidence), those are the children of the parents who paid for the presents.

Santa Claus decided to give at least some hope for the other children and he suggests to distribute the presents via a counting game. He starts from kid number s and gives a present to each k-th kid in a circle until he runs out of presents. In other words, the presents go to kids with numbers $(s + i \cdot k) \mod n$, $i = 0, 1, \ldots, m - 1$. Note that the kids, who have already got presents, are not excluded from the circle and continue to participate in the counting. Help Santa Claus choose such numbers s and k, that each good kid got a present.

Input

The first line contains two integers — n and m ($2 \le n \le 10^9$, $1 \le m \le \min(n, 10^6)$). The next line contains m integers from 0 to $n - 1$ — the numbers of the good kids. All numbers are distinct and given in the increasing order.

Output

Print two integers s and k ($0 \le s \le n - 1$, $1 \le k \le n - 1$) that give the solution for the problem. if there are multiple solutions, you can print any of them. If there is no solution, print "`-1 -1`".

Examples

standard input	standard output
5 3 1 2 4	2 2
6 3 1 2 4	-1 -1

Problem J. Gennady and Problems

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

It is well known that Gennady is a great master of solving programming tasks. The time that is takes Gennady to solve a problem does not depend on it's difficulty, but only on how tired Gennady is.

Today Gennady is going to solve n problems. He knows that he will solve the first problem in one minute, the second problem in two minutes, the third one in three minutes, and so on.

But sometimes Gennady can make pauses in solving problems and restore his power by eating a chocolate bar. To take a rest and eat a chocolate bar, Gennady spends exactly t minutes. He can eat a chocolate bar only in moments between solving problems. He should completely solve a problem to eat a chocolate bar. After Gennady eats a chocolate bar, he again solves next problem in one minute! But power of Gennady does not restore completely: the time of solving the following problems depends on the number of eaten chocolate bars. If he has eaten x chocolate bars, he spends $(x + 1)$ minutes more on each succeeding problem (he spends $(x + 2)$ minutes on the second problem, $(2x + 3)$ minutes on the third, and so on).

For example, if after solving five problems Gennady eats the first chocolate bar, the 6-th, the 7-th and the 8-th problem will take him one, three and five minutes correspondingly. If he eats one more chocolate bar after that, he will solve the 9-th problem in one minute and the 10-th problem in four minutes. If it takes $t = 4$ to eat a chocolate bar, then to solve ten problems using this strategy he will spend $1 + 2 + 3 + 4 + 5 + 4 + 1 + 3 + 5 + 4 + 1 + 4 = 37$ minutes.

For sure, Gennady wants to solve all n problems as soon as possible. Help him to arrange pauses in an optimal way. Assume that he has an unlimited number of chocolate bars.

Input

Input contains two integers n and t ($1 \le n, t \le 10^6$).

Output

In the first line print the minimum time that Gennady can solve all problems in. In the second line print k — the number of chocolate bars he should eat to do this. In the third line print the increasing sequence of k integers $b_1, b_2, \ldots, b_k$, where b_i denotes that Gennady should make the i-th pause after solving b_i problems.

Examples

standard input	standard output
10 4	37 2 5 8
10 20	55 0

Problem K. TopoCM++

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Polycarp is taking part in a programming contest based on new and innovative rules called TopoCM++. There are n problems in the contest and the contest has unlimited duration. According to the rules, each problem has expected submission time, the expected submission time of the i-th problem is t_i.

Ideally, a contestant should solve each problem not later than in time t_i, but in the real world some contestants solve problems later. As the contest lasts infinitely, each contestant will solve all the problems sooner or later. The score of the contestant depends on the maximum delay among all the problems. Thus the goal of each contestant is to solve all the problems in time. If it is impossible, the goal is to minimize the maximum value of $r_i - t_i$, where r_i is the time when the i-th problem was solved.

Polycarp has made detailed analysis and found out that for the i-th problem he needs a_i time units to come up with an algorithm and b_i time units to code it. So in total, he solves the i-th problem in $a_i + b_i$ time units.

Polycarp is not a multitasking operating system, so he can't do several things at the same time. Polycarp is planning to organize his work in the following way. In total, he has $2n$ jobs: n jobs to come up with an algorithm (thinking jobs) and n jobs to write a code (coding jobs). He will do the jobs in any order with the only constraint: for each problem he should first come up with an algorithm (think) and only after it he can write a code (code) for the problem. He can't interrupt jobs. Once he has started the job, he should finish it before switching to any other one.

Each time Polycarp switches between a type of activity (from thinking to code or back) he spends some time to be ready for new activity. Polycarp knows two values f_t and f_c, where f_c is the time he needs to prepare for thinking and f_c is the time he needs to prepare for coding. He needs f_t time units to prepare before the first job (for sure, it is a thinking job) and each time he changes the type of jobs, he needs f_c or f_t time units.

Help Polycarp to find the order of jobs to solve all the problems in time or (if it is impossible) to minimize the delay.

Input

The first line of the input contains three integers n, f_t and f_c ($1 \le n, f_t, f_c \le 2 \cdot 10^5$). The following n lines contain the descriptions of the problems, one description per line. The i-th line contains a_i, b_i and t_i ($1 \le a_i, b_i \le 2 \cdot 10^5$; $1 \le t_i \le 10^{12}$), where a_i is the time to think for the i-th problem, b_i is the time to code the i-th problem and t_i is the maximum time time to solve it in time.

Output

Print the required minumum delay to solve all the problems. Formally, it is the minimum value for the $\max\{r_i - t_i\}$, among all problems solved not in time, where r_i is the time when Polycarp solved the i-th problem.

Print the sequence of $2n$ numbers in the second line. For each i between 1 and n it should contain exactly once $-i$ and i, where $-i$ stands for the job to come up with an algorithm for the i-th problem and i stands for the coding job for the i-th problem.

Examples

standard input	standard output
5 2 2 3 3 4 2 1 21 1 3 8 1 3 20 1 2 16	8 -4 -3 -1 1 3 -2 -5 5 2 4

Problem L. Area Coverage (Division 2 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Given the satellite photographs of some secret object (that is to say, the rectangular area covered by each, since the contents of the photographs themselves are of course top-secret!), can you work out what the total area is of all that is photographed? Note that certain areas can appear on multiple photographs and should be counted only once.

Input

On the first line one positive number T: the number of test cases, at most 100. After that per test case:

- one line with an integer n ($1 \le n \le 1000$): the number of photographs.
- n lines with four space-separated integers x_1, y_1, x_2 and y_2 ($0 \le x_1, y_1, x_2, y_2 \le 10^6$, $x_1 < x_2$ and $y_1 < y_2$): the coordinates of the southwest and northeast corner, respectively, of each photograph.

The photographs are all rectangular in shape with their other corners at (x_1, y_2) and (x_2, y_1). The coordinates correspond to a flat two-dimensional space (i.e. we assume the Earth to be flat).

Output

Per test case — one line with an integer: the total area of all that appears on the photographs.

Examples

standard input	*standard output*
2 3 0 6 20 16 14 0 24 10 50 50 60 60 2 0 0 20 10 10 4 14 8	376 200

Problem M. New Car (Division 2 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

In Byteland, before some bureaucrat is allowed to order a car and all the parts he needs, he will first need to determine the exact amount of money that he needs. Management will then decide whether he gets to realize his vision or not.

This sort of bureaucratic nonsense is not something that our bureaucrat wants to waste any time on. He therefore wants you to help him. Given the price of the car, a listing of the parts and the price per piece of each part, work out what the total cost is of the project.

Input

On the first line one positive number: the number of test cases, at most 100. After that per test case:

- one line with an integer s ($1 \le s \le 10^5$): the stock price of the car.
- one line with an integer n ($0 \le n \le 1000$): the number of different parts.
- n lines with two space-separated integers q_i and p_i ($1 \le q_i \le 100$ and $1 \le p_i \le 10^4$): for each different part i, the number of pieces q_i required of such part, and the price p_i for a single piece.

Output

Print one integer — the total amount of money needed to buy the car and all the parts.

Example

standard input	*standard output*
2 10000 2 1 2000 3 400 50000 0	13200 50000

Problem N. Concave Quadrilaterals (Division 2 Only!)

Input file:	concave.in
Output file:	concave.out
Time limit:	2 seconds
Memory limit:	512 mebibytes

Given is a grid consisting of r rows by c columns of grid points.

Count the number of ways in which it is possible to draw a strictly concave quadrilateral with vertices in four of the given grid points. (A quadrilateral is strictly concave if and only if one of its two diagonals contains a point that is strictly outside the quadrilateral. Shifted and/or rotated copies of the same quadrilateral count as distinct ways of drawing it.)

Input

The input has a single line with the two positive integers r and c. You may assume that the total number of grid points is at most 3000.

Output

Output a single line with a single integer: the number of concave quadrilaterals with vertices on the grid.

Examples

concave.in	concave.out
2 10	0
3 3	24

Problem O. Opening The Lock (Division 2 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

The key to door consists of four digits. Hacker Vasya has received a list of possible keys that he needs to try out. Trying out the whole list will take too long, so Vasya needs to find a way to reduce the list.

Lock built in next way: key is valid if using addition, subtraction, multiplication, division (not an integer division, i.e. $6/5 = 1.2$ and not 1) and parentheses, from digits of this key is possible to build an expression with value 24.

For example, the key (4, 7, 8, 8) satisfies the condition above, because $(7 - 8/8) \cdot 4 = 24$. The key (1, 1, 2, 4) does not satisfy it, nor does (1, 1, 1, 1). These keys cannot possibly be the valid key and do not need to be tried.

Write a program that takes the list of possible keys and outputs for each key whether it satisfies the lock condition or not.

Input

On the first line one positive number: the number of test cases, at most 60. Each test case consists of one line with four space-separated integers a, b, c, d ($1 \le a, b, c, d \le 9$): a possible key.

Output

Print one line with either "YES" or "NO", indicating whether the key satisfies the lock condition or not.

Example

standard input	*standard output*
4 4 7 8 8 1 1 2 4 1 1 1 1 1 3 4 6	YES NO NO YES

Problem P. BSA (Division 2 Only!)

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Byteland security agency started to snoop e-mails of some anti-government group. Assume that each e-mail consists only of words made up of lower-case letters, commas, periods, and spaces (space, tab, newline). There is a list of keywords that raise alarms. If at least a certain number of keywords occur in the text (at least once each), then the e-mail raises an alarm. You are to write a program that determines which e-mails raise alarms.

Input

The first line is the number T of input data sets ($1 \le T \le 40$), followed by the T data sets, each of the following form: The first line contains three integers n, k, d, separated by spaces. $1 \le n \le 100$ is the number of words that raise alarms; $1 \le k \le 100$ is the number of lines in the e-mail that follows, and $1 \le d \le 100$ is the number of keywords that need to occur to raise an alarm.

This is followed by n lines, each containing a word of at most 20 lowercase letters (and no other characters). Next come k lines, each containing at most 80 characters, a mix of lowercase letter, commas, periods, and different forms of spaces. No words will break across lines (a newline character ends a word, and a new word starts on the next line). We are only looking for exact matches, so "`revolution`" and "`revolutions`" do not match.

Output

For each data set, output on separate line "`Danger`" or "`No`", depending on whether the the number of distinct keywords in the e-mail meets (or exceeds) the threshold or not.

Example

standard input
2 5 1 3 acm contest icpc opencup winner acm contest acm me you .,contests acm nwerc icpcicpc .. contest 4 2 3 acm contest icpc opencup acm contest me acm bill .,contest acm nwerc, hello world icpc acmacm .. contest is over

standard output
No Danger