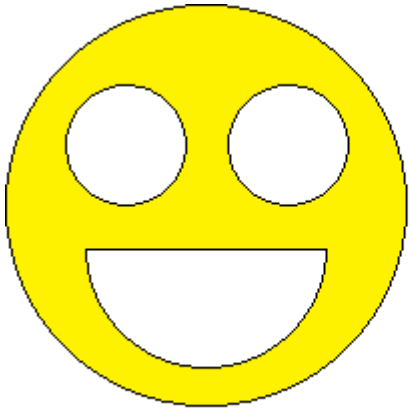
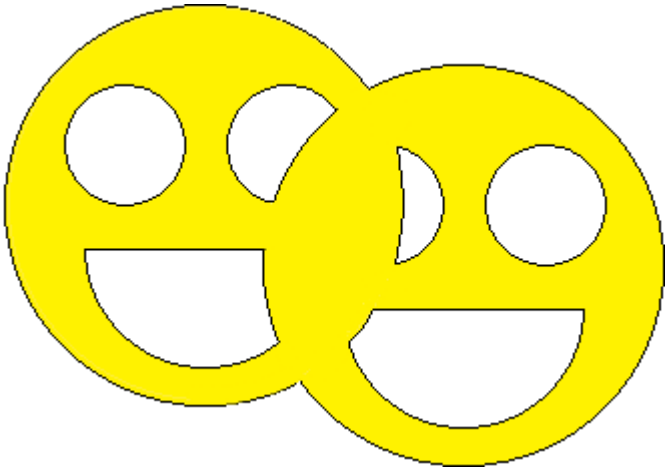
Problem A. Area of Smileys Union

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

A *smiley* with center (x_0, y_0) is the circle of radius 100 and center (x_0, y_0), with two circular holes of radius 30 and centers $(x_0 - 40, y_0 + 30)$ and $(x_0 + 40, y_0 + 30)$, and a semicircular hole that is the lower half of the circle of radius 60 and center $(x_0, y_0 - 20)$ (look at the picture for clarity).





You are given descriptions of two smileys. Calculate area of their union.

Input

Input contains four space-separated integers $-1000 \le x_1,\ y_1,\ x_2,\ y_2 \le 1000$, where $(x_1,\ y_1)$ are coordinates of center of first smiley, and $(x_2,\ y_2)$ — coordinates of center of second smiley.

Output

Print area of union of two smileys with absolute error 10^{-4} or less.

Examples

standard input	standard output
-1000 -1000 1000 1000	40212.3859659494
0 0 0 0	20106.1929829747
0 0 -10 0	23899.0852307386

Problem B. Bakery

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

The bakery may bake up to c_i loaf of bread in i-th day, and it will cost f_i coins per a loaf. The citizens buy d_i loafs per i-th day, and there is place for g_i additional loafs on the stock to stay overnight between day i and day $i+1$ for e_i coins per one loaf.

Your task is to find minimal amount of coins to support the citizens with bread for n days.

Input

The first line of the input contains one integer n ($1 \le n \le 10^5$).

Then n lines follow, each containing three integers c_i, f_i and d_i. Then $n-1$ lines follow, containing the integers g_i, e_i ($0 \le c_i, f_i, d_i, g_i, e_i \le 10^9$).

Output

Print one integer — the minimum amount of the coins, needed to support the citizens with bread for n days. If it is impossible to support the citizens at least for some day, print -1 instead.

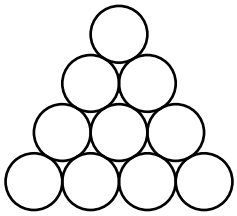
Examples

standard input	standard output
3 10 1 1 2 2 2 10 10 8 7 2 6 5	73

Problem C. Coverings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Consider the figure built from the circles of the same size, that have the following form. The figure consists of n layers, i-th of those layers consists of i circles. The adjacent circles in each layer touch, additionally, for each layer $i < n$ j-th circle in i-th layer touches j-th and $(j+1)$-th circle in $(i+1)$-th layer. At the figure you may see the example of such figure for $n = 4$.



As *triangle* we will consider three circles that touch each other.

Covering the figure with triangles is defined as the set of triangles such as each circle in the figure belongs to exactly one triangle.

We will encode the coverings in the following way.

- If the upper layer of the triangle contains one circle, and the lower layer contains two circles, then the **circles** of that triangle are denoted with the letter '`A`'.
- If the upper layer of the triangle contains two circles, and the lower layer contains one circle, then the **circles** of that triangle are denoted with the letter '`B`'.

The coverings then can be represented (and compared) as the ordered sequences of the strings, representing the layers from the top to the bottom.

Note that there are some n where covering does not exists (for example, $n = 4$ from the picture above). However, we will consider only n such as at least one covering exists.

Consider all possible coverings of the figure of n layers and order them lexicorgraphically. Your task is to find the encoding of k-th of those coverings.

Input

The first line of the input contains two integers n and k ($1 \leqslant n \leqslant 23$, $0 \leqslant k \leqslant 10^9$). You may assume that k-th covering of the figure of n layers exists. The covering are numbered from zero.

Output

Print n lines containing the encoding of the k-th in the lexicographical order covering of the figure of n layers. i-th line shall contain the letters, describing the circles of the i-th layer. The adjacent letters may be separated by any (possibly zero) number of the spaces. Spaces at the beginning and at the end of the string are tolerated as well.

Example

standard input	standard output
2 0	A A A

Problem D. Dense Polygon

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Given N points on the plane. Your task is to select **exactly** K of them such as they form the convex polygon S with the minimal area.

Input

The first line of the input contains two integers N and K ($3 \le N \le 100, 3 \le K \le 20$, $K \le N$). Each of the following N lines contains two integers x_i, y_i ($-10^4 \le x_i, y_i \le 10^4$) — the coordinates of one point. You may assume that no three points are collinear.

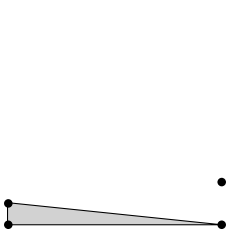
Output

Print the area of the polygon S with absolute or relative error 10^{-4}. If it is impossible to choose K points in a way that they form the convex polygon, consider the area as zero.

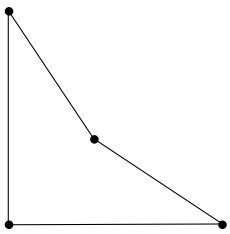
Examples

standard input	standard output
4 3 0 0 10 0 10 2 0 1	5.0
4 4 0 0 10 0 4 4 0 10	0

Note



Sample 1



Sample 2

Problem E. Effective Painting

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

Consider the following process. The line initially is uncolored.

The painter first paints the segment of the of color 1, connecting points l_1 and r_1 inclusively, then paints the segment of color 2 between points l_2 and r_2 and so on, ending with the color n and the segment from l_n to r_n, inclusively.

It appears that some colors are completely repainted by the segments of another colors that were painted later, so overall number of the visible colors is less than n.

The painter may consider another order of the painting the same segments in the same colors, such as the number of the visible colors is maximal possible. Your task is to find that number of colors.

Input

The first line of the input contains one integer n ($1 \le n \le 300$). Each of the following n lines contain two integers. i-th of those lines contains the integers l_i and r_i ($-1\,000\,000\,000 \le l_i < r_i \le 1\,000\,000\,000$).

Output

In the first line print the number of visible colors, if the coloring order will be chosen optimally.

The second line shall contain n integers; i-th of those integer denotes the index of the segment that shall be colored at i-th step. The segments numeration is 1-based; the segments are enumerated in the same order they are given in the input.

If there is more than one solution, print any of them.

Example

standard input	standard output
4 1 3 2 4 2 3 1 4	3 4 1 2 3

Problem F. Fibonacci Palindromes

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Consider the following binary strings:

$S_0 =$ "0"

$S_1 =$ "01"

$S_2 = S_1 S_0 =$ "010"

$S_3 = S_2 S_1 =$ "01001"

$S_4 = S_3 S_2 =$ "01001010"

$S_5 = S_4 S_3 =$ "0100101001001"

...

$S_n = S_{n-1} S_{n-2}$

...

As you can see, the string S_{n-1} is the prefix of the string S_n for any $n > 0$. Consider the infinite string S_∞, that is an *limit* of strings S_n, i.e. the string such as for any $n \geq 0$ the string S_n is the prefix of the string S_∞.

We will call S_∞ *the Fibonacci string*.

You are given several substrings of the Fibonacci string. Check if they are the palindromes.

Input

The first line of the input contains one integer n — the number of the substrings ($1 \leq n \leq 10^4$).

Then the substrings follow. Each substring is given by two integers *start* and *len* and corresponds to the substring of S_∞, starting from the position *start* (in 1-based numeration) and consisting of *len* characters ($1 \leq start, len \leq 10^9$).

Output

Print n lines, each line per one substring. If the given substring is a palindrome, print "YES", otherwise print "NO".

Example

standard input	standard output
4 2 4 3 5 3 7 1 11	YES NO YES YES

Problem G. Guess The Integer

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

You are playing with an Sphinx, and your task is to guess an integer in the interval $[0, N)$, i.e. if the number is x, then the inequalities $0 \le x < N$ are satisfied.

You may ask the questions in the form "Does the x belong to the cyclic interval $[a, b)$ where $0 \le a, b \le N$?", to which you will get a "`Yes`" or "`No`" answer.

The *cyclic interval* is defined in the following way:

- $[a, b) = \{x | a \le x < b\}$ if $a < b$;
- $[a, b) = \{x | a \le x < N \text{ or } 0 \le x < b\}$ if $a > b$
- If $a = b$, then the interval $[a, b)$ is empty.

But Sphinx is bored, so he decided to make the game more interesting: she can decide to give a false answer to **each** question starting at a certain point. I.e. you know that up to a certain successive question all the answers of the Sphinx are correct, and after that they are all lies, but you do not know how many correct answers to expect. The Sphinx is very capricious and can start lying from the very beginning, or can give correct answers all the time.

Write the program, that correctly finds any integer in the interval $[0, N)$ with no more than **56** questions.

Note that you shall be able to guess the secret integer for every first question, from which the lying begins.

Interaction Protocol

First the Sphinx tells you the integer N ($1 \le N \le 10^{15}$) — the upper limit for the secret value of x.

To ask the Sphinx about the cyclic interval $[a, b)$, use the query in the format `?` a b ($0 \le a, b < N$). The Sphinx will answer 1, if she did not started to lie and the integer x belongs to that cyclic interval, or if she started to lie and the integer x does not belong to that cyclic interval, and 0, if she did not started to lie and the integer x does not belong to that cyclic interval, or if she started to lie and the integer x belongs to that cyclic interval.

Note that the interactor is **adaptive**, i.e. the answers are consistent with the previous answers (and the rules of the game), but the final integer may be defined at the process of the interaction.

To tell the answer, use the format `!` x.

Do not forget to flush the output after each query.

Example

standard input	standard output
4	? 2 0
1	? 2 2
0	? 0 4
0	? 2 3
0	! 2

Problem H. Hypercycles

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	1024 mebibytes

Given the string S. Lets call the substring T of the string S *hypercycle*, if **all** cyclic shifts of the string T can be found in string S as the substrings. Consider the *frequency* of the substring as the number of different places in S, where this substring can be found.

Your task is to answer the following queries: given integer L_i, find the hypercycle of length L_i such as the sum of frequencies of all **distinct** strings that are its cyclic shifts is the maximal possible and print that sum. If there is no such hypercycle, the sum is equal to 0.

Input

The first line of the input contains two space-separated integers N and M ($1 \le N \le 5 \cdot 10^5$, $1 \le M \le 8$) — the length of the string S and the number of queries.

The second line contains the string S, consisting of N uppercase English letters.

Then M lines follow, each containing one integer L_i ($1 \le L_i \le N$).

Output

Print M lines — the answer to the problem for each L_i.

Examples

standard input	standard output
16 2 AABAABACDABAABAA 6 3	4 10
3 3 AAA 1 2 3	3 2 1

Problem I. Interactive Smiley Face

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Igor likes smiley faces a lot. He wrote a program that generates really large pictures of white and black pixels with smiley faces. Depending on Igor's mood, a picture can be a happy or a sad smiley. In the sequel white pixels are represented with 0's, and black pixels are represented with 1's.

Igor's program behaves as follows. First, a basic 41×41 rectangular pattern is fixed (patterns can be found in the archive under `Samples ZIP` tab in folder, related to this problem). Then, an odd integer $k \geq 1$ and the pattern is magnified k times. That is, the result will be a $(41 \cdot k) \times (41 \cdot k)$ rectangle, with each $k \times k$ block having the same color as the corresponding pixel of the basic pattern.

After that, a happy smile or a sad smile consisting of white pixels is put on the face (see basic patterns for smiles below). The program can choose an integer $m \geq 1$ and magnify the smile m times (magnification behaves the same way as with the face pattern). The magnified smile is then superimposed over the face in such a way that:

- All pixels of the smile should lie strictly below than the center pixel of the face (note that since $41 \cdot k$ is odd, center is defined unambigiously).
- No white pixel of the smile is put over a previously white pixel of the face pattern.
- Moreover, the border of the face should stay untouched by pixels of the smile. By border of the face we mean extreme black pixels in a row a column.

After this step, the resulting picture is placed on a $10^9 \times 10^9$ rectangular grid of initially white pixels. All pixels of the smiley face should be inside the grid. Pixels of the grid have usual integer Cartesian coordinates (x, y) ranging from 1 to 10^9. Note that the $0x$ axis is pointed rightwards, and $0y$ axis is pointed upwards, that is, the bottom pixels of the grid (and, therefore, the smiley face) have lower y coordinate.

Igor challenged you to guess the mood of his smiley face. Since the grid is very large and cannot be sent to you directly, Igor implemented a system to answer your queries. A query is a rectangle with integer coordinates of corners; the answer to the query is the number of black pixels inside the rectangle.

Your task is to guess the mood of the smiley face by asking at most 500 queries.

Interaction Protocol

Initially your program receives no input.

To ask a query, your program should print four integers x_1, y_1, x_2, y_2 ($1 \leq x_1 \leq x_2 \leq 10^9$, $1 \leq y_1 \leq y_2 \leq 10^9$) on a separate line — coordinates of the bottom left and top right corners of the rectangle respectively. Do not forget to flush your output after each query.

After each query your program is fed one integer — the number of black pixels within the rectangle.

To give the answer, your program should print -1, then at new line print "`HAPPY`" (without quotes) if smiley is happy, or "`SAD`" otherwise. Then flush your output and exit from the program.

Note

Patterns can be found in the archive under `Samples ZIP` tab. `base.dat` contains 41×41 base pattern, `happy.dat` contains happy smile pattern, `sad.dat` contains sad smile pattern, each of files `face1.txt` and `face2.txt` contains example pictures of base with added smiles.

Problem J. Join The Relay

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

N runners are planning to run a relay on the distance L meters, such as each runner will run on the distance not less than D meters. Their task is to finish whole distance with the summary time W seconds. Each runner may be in bad mood and run one meter for S_i seconds, or in good mood and run one meter for T_i seconds. But before the runner starts, noone knows which mood will be on.

Your task is to distribute the distance between runners, such as if they all will be in the bad mood, they will not use more than W seconds for whole distance, and if they all will be in the good mood, the time is as small as possible.

Input

The first line of the input contains integers N ($1 \le N \le 200\,000$) — the number of the runners, D ($0 \le D \le 10$) — the minimal distance for one runner, L ($1 \le L \le 10^5$) — the total distance, W ($1 \le W \le 10^9$) — time limit for finish (even in the worst case). If $D = 0$ you may skip some runners.

Then N lines follow, describing the runners. Each line contains two integers S_i and T_i ($1 \le T_i \le S_i \le 200\,000$) that are denoting the time used for i-th runner to run one meter while being in bad and in good mood, respectively.

Output

Print one integer — the fastest time to finish in case when all runners are in the good mood, just as if they all are in bad mood, the time to finish is not greater than W, with absolute or relative error 10^{-6} or better. If there is no way to secure time W, print "`No solution`" instead.

Examples

standard input	standard output
2 1 16 113 8 3 6 6	70.5000000
2 0 10 42 2 1 3 2	10.0000000
3 3 20 10 7 1 7 2 7 3	No solution

Note

In the first sample the first runner runs 8.5 meters, second — 7.5 meters.

In the second sample $D = 0$, so we can use only one the first runner to run whole distance.

Problem K. King's Voyage

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

In the Treeland kingdom, N cities are connected by $N-1$ bidirectional roads such as any two cities are connected by the roads directly or indirectly. For each road, its length is known.

The King plans to visit some cities. The Minister of Ceremonies knows that for city k there are F_k citizens that want to talk with the King. The people may even go to another city to talk with King, but only if the home city is on the distance D or less from it. Visit of the king in any city costs S; one citizen pays T to the royal treasury for the King's audience.

Find the maximal earnings for the royal treasury, if the visit is planned wisely (and note that it is possible to visit 0 cities at all).

Input

The first line of the input contains four integers N, T, D and S — the number of the cities, the payment for the audience, the maximal distance that can be covered by person from another city and the cost of King's visit in one city, respectively ($1 \le N \le 1000$, $1 \le T \le 2000$, $1 \le D \le 10^6$, $1 \le S \le 10^9$).

Each of the following N lines contains one integer F_i — the number of citizens of city i that wants to have an audience from the king ($0 \le F_i \le 1000$).

Each of the following $N-1$ lines contains three integers A_i, B_i, C_i ($1 \le A_i, B_i \le N$, $1 \le C_i \le 2000$)— the number of cities, connected by that road, and its length, respectively. You may assume that the given graph is a tree.

Output

Print one integer — maximal earnings for the royal treasury.

Examples

standard input	standard output
7 10 5 100 5 11 0 0 13 10 1 1 3 3 2 3 3 4 3 2 4 7 4 1 5 5 4 6 6	100

Problem L. Lazy Teacher

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 mebibytes

At the math lesson the teacher wrote an integer matrix with N rows and M columns. The numbers A_{ij} in each row i of the matrix are placed in strictly ascending order; the same is true for the numbers in each of the j columns — they are in strictly ascending order.

Then he asked Alice to perform Q actions of the following two types:

- 1 X — Alice must answer whether the number X is contained in the matrix.
- 2 R C Y, Alice shall add Y to each of the numbers in the matrix that are down and to the right (inclusive) of the cell with row R and column C.

The teacher is too lazy to check the answers by self, so he asked you to write the program that will simulate whole process.

Input

The first line of the input contains two integers N and M — the number of rows and columns in the matrix, respectively ($1 \le N, M \le 1000$).

Each of the following N lines contains M integers — the initial values of A_{ij}. ($1 \le A_{ij} \le 10^9$, $A_{ik} < A_{il}$ if $k < l$, $A_{kj} < A_{lj}$ if $k < l$).

Then one integer Q follows — the number of the queries ($1 \le Q \le 2 \cdot 10^4$). Each of the following Q lines contains description of one query. The query of the first type have the form 1 X ($1 \le x \le 2 \cdot 10^9$). The query of the second type have the form 2 R C Y ($1 \le R \le N$, $1 \le C \le M$, $1 \le Y \le 5 \cdot 10^4$).

Output

For each query of type 1, print 1, if X can be found in the matrix, and 0 otherwise.

Example

standard input	standard output
3 4 11 13 17 18 13 14 19 22 17 20 23 42 9 1 42 1 24 2 2 2 5 1 24 2 1 3 7 1 20 1 32 1 19 1 42	1 0 1 0 0 1 0

Problem M. Manipulation With Strings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Given N pairwise distinct strings. Your task is to choose as many of those strings as possible to form the sequence of strings S_i, such as S_{i-1} is the substring of S_i for $i > 1$.

Input

The first line of the input contains one integer N ($1 \le N \le 10^4$). Then N non-empty strings follow, one string per line. The length of one string does not exceed 1000. Thr strings are composed from lowercase English letters and are pairwise distinct. The summary length of all lines does not exceed 10^6.

Output

Print one integer — the length of the sequence S.

Examples

standard input	standard output
7 metrostroitelei metro beschenko stroi metrostroitel stroitel metr	4
2 abcd bc	2