

Problem A. Alice's Certification

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

The Macrostop Corp. conducts a promo-action and gives a beautiful calendar and a fine badge “Macrostop Certified Specialist in Ghost Protection” to anyone who successfully passes a certification exam.

The certification exam consists of K rounds. At each round the candidate is given N questions. Each question has two possible answers (one is correct, the other one is wrong) and the candidate has to choose one of them. A candidate is notified whether she passed the round or not after answering all the questions of the round. She is not informed, which of her answers were wrong. The round is considered successful if the candidate makes no more than one mistake in her answers (that is, the number of correct answers is at least $N - 1$). If a round is failed the candidate quits the exam, otherwise, she proceeds to the next round. On completing successfully the last round the candidate becomes a certified specialist.

The number of attempts to pass the exam for a candidate is not limited, but each time she has to start from the first round. However, it is known that the set of questions and correct answers does not change.

Alice knows nothing about Ghost Protection and does not want to prepare for the exam, but she likes the calendar very much and she is eager for it. What is the minimum number of attempts Alice has to take in the worst case to realize her dream?

Input

The first line of the input file contains two integer numbers N and K ($0 < K < 10$). To prevent cheating all the certification exams always have N equal to 2, 3, 7, 15, or 31.

Output

Print out the minimal number of attempts required to pass the exam in the worst case.

Example

<i>standard input</i>	<i>standard output</i>
3 2	3

Problem B. Baliset

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

The famous musician R. Whiteless of the Improbable Baliset Masters band is about to re-release his composition “Stroke on the Motor”. He wants to re-record the composition using its old tunes and chords and add some new effects and enhancements. To achieve this he uses his beloved electrobaliset Gender, but with different non-standard tunes.

The Western Music theory uses the *music scale* consisting of the repeating sequence of the following *notes*

... A A# B C C# D D# E F F# G G# A A# B C C# ...

The difference between any two consecutive notes is called a *semi-tone*. Any 12 consecutive notes (for example, from C to B) form an *octave*. Thus the music scale is divided into octaves. In order to distinguish the same note in different octaves, its name is followed

by the octave number, for example A#1, A#3, but for simplicity's sake we will ignore octave numbers and we will use just the names of the notes. A *chord* is a set of three or more different notes (for example C, E, G) played simultaneously. When a chord is played its notes may be repeated, for example C, E, C, E, E, G.

All the plucked stringed instruments (i. e. played with fingers or plectrum, like guitar or gusly) are constructed in a similar way. The instrument has M strings (for example, a classic guitar has six strings). When a string is *open* (i. e. non-pressed) it sounds as a certain note of the music scale. The sequence of notes played on open strings is called the *tune* of the instrument. For example, a classic guitar has the standard tune E, B, G, D, A, E, i. e. the open first string sounds as E (in certain octave, but as mentioned above, we simply ignore the octave number), the second open string sounds as B, etc. Strings are numbered starting from one.

The strings are strained over the *neck*, which is divided into *frets*. The instrument has N frets, numbered from 1 to N (for example, a classic guitar has 20 frets). Each string may be pressed to any fret independently from other strings. If a string is pressed to the j th fret, it sounds j semi-tones higher than when this string is open. For example, if an open string sounds as C, it sounds as F if pressed to the 5th fret. We assume that a string is never pressed to more than one fret at a time. One can simultaneously press several strings to certain frets. If a set of pressed and open strings sounds as one chord, such set is called a *fingering* of the chord. A fingering for the chord C, E, G on a classic guitar is shown in the Fig. 1. Historically, the strings are numbered from right to left. The first string is open, the second string is pressed to the first fret, etc.

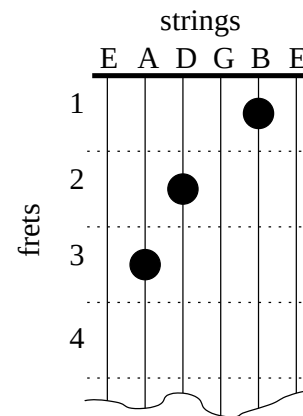


Figure 1: Fingering for the chord C, E, G on a classic guitar.

A chord may have several different fingerings. A fingering for a chord may use a part of all the strings, for example on a classic guitar there is a fingering for the chord D, F, A, which uses only strings 1–4.

Generally, a fingering must satisfy the following properties:

- The fingering contains only consecutive strings.
- All the chord's notes appear in the fingering.
- No other notes appear in the fingering.

Let F_{max} be the maximal fret number of all the pressed strings in the fingering (0 is no string is pressed). Let F_{min} be the minimal fret number of all the pressed strings (0 is no string is pressed). The *height* H of the fingering is calculated as $H = F_{max} - F_{min} + 1$ if $F_{min} > 0$ and $H = 0$ if $F_{min} = 0$. The fingering in the Fig. 1 has height 3.

Let S_{max} be the maximal number of a string in the fingering. Let S_{min} be the minimal number of a string in the fingering. The *width*

W of the fingering is calculated as $W = S_{max} - S_{min} + 1$. The fingering in the Fig. 1 has width 6.

Baliset is a plucked stringed instrument. However, it usually has quite large number of strings and calculation of the best fingering is not a trivial task.

The *best fingering* is the fingering satisfying the following conditions (in the order of decreasing priority).

- The fingering has the largest possible width.
- The fingering has the smallest possible height.
- The value of F_{min} is the smallest possible.
- The value of S_{min} is the smallest possible.

R. Whiteless asks you to write a program, which builds the best fingering for a given chord on the baliset with a given tune. He does not want to deal with fingerings of large height, because such fingerings are hard to play, so he limits the maximal height of the fingering by some value L .

Input

The input has the structure as follows. The first line contains three integers M, N, L , where $3 \leq M \leq 5000$, $0 \leq N \leq 100$, $0 \leq L \leq N$. The next line contains the specification of the baliset's tune as the sequence of M notes separated by spaces. The next line contains the number of notes P in the chord ($3 \leq P \leq 5$). The next line contains P notes of the chord. All the notes are different.

Output

If a fingering satisfying the input data does not exist print -1 four times. Otherwise, print the following four numbers describing the best fingering: $S_{min}, S_{max}, F_{min}, F_{max}$.

Example

<i>standard input</i>	<i>standard output</i>
6 24 4 E B G D A E 3 C E G	1 6 1 3
6 24 0 E B G# D A E 3 C E G	-1 -1 -1 -1

Problem C. Central Square

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

Once upon a time there was a Great Anniversary of Glorious City. Before the anniversary the Mayor decided to repair Central Square of the city. Central Square had the size of $X \times Y$ meters and it was paved with rectangular stones of different sizes. The sides of the stones were parallel to the sides of the square. In the years before the anniversary the cracks between the stones were filled with pressed dirt, so when the stones were removed from the square and sent to a restoration workshop, their previous locations were clear due to the remainings of the dirt between the stones.

But during the restoration someone had stolen some stones (maybe for his new mansion — who knows?). So, in order to hold *esprit de corps* to the leaders of the foreign powers the Mayor had to start a public investigation of the incident. The first task of the Mayor's investigation was to determine how many stones were stolen. The Mayor had decided to make a plan of the square with all the dirt

strips between the stones. Due to bad weather some fragments of the strips were destroyed, but fortunately all the dirt on the corners was intact. Thus, on the plan of the square the strips of dirt were drawn as crossing segments. Fortunately again, at least one meter of a dirt strip before each crossing point remained intact. The Mayor asks you to write a program to calculate the original number of stones on Central Square.

Input

The first line of the input contains three integer numbers X, Y, K , where $0 < X, Y \leq 1000000$, $0 < K \leq 5000$. X is the length of the square from East to West, Y is the length of the square from South to North. $(0, 0)$ is the South-Western corner of the square, (X, Y) is the North-Eastern corner of the square. K is the number of segments. The next K lines contain the descriptions of the segments. Each line contains 4 integer numbers x_1, y_1, x_2, y_2 ($0 \leq x_1, x_2 \leq X$, $0 \leq y_1, y_2 \leq Y$) — coordinates of the ends of the corresponding segment. The segments are parallel to the sides of the square. Segments do not overlap but can intersect each other.

Output

Print out the original number of stones on Central Square.

Example

<i>standard input</i>	<i>standard output</i>
3 6 12 0 0 1 0 2 0 3 0 0 3 1 3 2 3 3 3 0 6 1 6 2 6 3 6 0 0 0 1 0 2 0 4 0 5 0 6 3 0 3 1 3 2 3 4 3 5 3 6	2
2 2 6 0 0 2 0 0 0 0 2 0 2 2 2 2 0 2 2 0 1 2 1 1 1 1 2	3

Problem D. Dozor

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

It is a well-known fact that nowadays Moscow is patrolled by two opposing forces: Day "Dozor" and Night "Dozor". To avoid any clashes between them the city is divided by imaginary straight lines into sectors and the sectors are allotted to Day and Night Dozors. No two sectors belonging to the same Dozor share a common segment of the boundary. On this imaginary map Red Square is located at the origin $(0, 0)$ and no separation line goes through it. Currently Red Square is under control of Day Dozor.

You have lost yourself somewhere in Moscow. The only thing you know about your location is your coordinates. Given them, you have to find out who controls the the sector you are in.

Input

The first line contains two integers x and y which are the coor-

ordinates of your location. The second line contains the number of lines N ($1 \leq N \leq 1000$) separating the map. The next N lines describe these separation lines. Each description consists of four numbers x_1, y_1, x_2, y_2 denoting two distinct points (x_1, y_1) and (x_2, y_2) on the line. All separation lines are distinct.

You may assume that the point (x, y) is not on any of the separation lines. All coordinates in the input are integers not exceeding 10 000 by absolute value.

Output

Output the single word “Day” if you are currently in a sector controlled by Day Dozor and “Night” otherwise.

Example

<i>standard input</i>	<i>standard output</i>
1 2 2 1 0 0 1 3 0 4 1	Night
2 2 2 1 0 0 1 2 0 0 2	Day

Problem E. Expression

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

Many interesting and non-trivial problems of modern computer science and discrete mathematics are related to so called *Boolean functions*. A Boolean function f is just a mapping from B^n to B , where $B = \{\text{FALSE}, \text{TRUE}\}$ is the set of Boolean values and n is a natural number.

Boolean functions are usually given by *expressions*. In this problem an expression is allowed to contain variables **a**, **b**, **c**, parentheses (**(** and **)**), logical constants **0** (FALSE) and **1** (TRUE), and the following *Boolean operations*:

! **&** **|** **->**

This list contains the operations in the order of decreasing priority (the first one has the highest priority, the last one — the lowest). Here symbol **!** denotes negation (“NOT”), **&** denotes conjunction (“AND”), **|** denotes disjunction (“OR”), and the sequence of two symbols **->** denotes implication. The latter one can be expressed in terms of disjunction and negation as $\mathbf{a} \rightarrow \mathbf{b} = \mathbf{!a} | \mathbf{b}$.

Negation is a prefix unary operation (**!a** is a well-formed expression whereas **b!** and **c!d** are ill-formed). All other operations are binary and right-associative (which means that $\mathbf{a} \rightarrow \mathbf{b} \rightarrow \mathbf{c}$ is equivalent to $\mathbf{a} \rightarrow (\mathbf{b} \rightarrow \mathbf{c})$, for example).

The longer is an expression, the harder it is to understand it and examine its properties. So here comes the natural idea to find (for a given function f) an expression representing f and having the shortest possible length. The length of an expression is the number of symbols in it. Note that implication is written as a sequence of two characters and thus it contributes two symbols to the total length. This is widely known to be a hard task but we hope you are gifted enough to solve it for a special case of $1 \leq n \leq 3$.

If $n = 1$ then the function is of the form $f(a)$, if $n = 2$ then it is of the form $f(a, b)$, and finally if $n = 3$ it has the form $f(a, b, c)$ (that

is, you can only use variable a in the first case, variables a, b in the second, and all the variables a, b, c in the third).

The function f will be given to you by a *truth table*, which is constructed in the way as follows. Consider all possible Boolean sequences of length n and arrange them lexicographically in ascending order. For example, for $n = 2$ you get the sequence (0, 0), (0, 1), (1, 0), (1, 1). Now apply the function f , the resulting sequence of 2^n Boolean values is called the truth table of f .

Input

The first line of the input contains a single integer n ($1 \leq n \leq 3$). The second line contains the truth table of f where TRUE is represented by 1 and FALSE is represented by 0. The values of f in the table are separated by spaces.

Output

Output a single line containing the shortest expression consisting of variables, parentheses, logical constants, Boolean operations, and representing the given function. The line should contain the expression without blanks or tabs. If there are several possible expressions of the same shortest length you may output any of them.

Example

<i>standard input</i>	<i>standard output</i>
1 1 0	!a
2 1 1 0 1	a->b

Problem F. File Sorting

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

The Acme Corp. develops a new operating system called *Advanced Computer Manager* (ACM). The essential part of this operating system is the file manager. The file system of the ACM consists of files and directories. Each file or directory has the following attributes: *name*, *extension*, *time*, *size*. *name* is a non-empty sequence of at most 8 lowercase Latin letters or digits, *extension* is a sequence of at most 3 lowercase Latin letters or digits, *time* is the time of the last modification of the file, and is written as

dd.mm.yy hh:mm

where *yy* ($00 \leq yy \leq 99$) are the last two digits of the year Y ($1951 \leq Y \leq 2050$), *mm* is a month ($01 \leq mm \leq 12$), *dd* is a day ($01 \leq dd \leq 31$), *hh* is an hour ($00 \leq hh \leq 23$), *mm* is a minute ($00 \leq mm \leq 59$). All numeric parts of a time have two digits (i.e. all numbers are left-padded with zero if necessary). Finally, *size* is the size of the file in bytes ($0 \leq size \leq 1073741824$). It is guaranteed that there are no different files having the same pair *name*, *extension*.

A typical operation of the file manager is sorting of files in some directory according to a user-specified criterion. A user specifies the order of file attributes priority, which is a permutation of the four attributes: **name**, **extension**, **time**, **size**. The first attribute has the highest priority, i. e. the files must be sorted in ascending order of the first attribute. If several files have the same value of the first attribute, a tie is resolved by the second attribute, then by the third attribute, and finally by the fourth attribute.

Write a program, which sorts the files in a directory according to a specified criterion.

Input

The first line of the input file contains the number of files N ($0 < N < 1025$). The next N lines describe the attributes of the files. Each description is located in a separate line. The description has the form as follows:

name.extension time size

Name and extension fields are separated with “.” (dot) character; *extension*, *time*, and *size* are separated with one space character. The last line contains the specification of the sorting criterion: four attribute names separated with one space symbol.

Output

Print out the names and extensions of the files in the required order. Information for each file should be printed on a separate line. Name and extension should be separated with “.” (dot) character.

Example

<i>standard input</i>	<i>standard output</i>
3	f.12
f.12 23.10.04 10:00 2	g.12
f.2 23.10.04 10:00 2	f.2
g.12 23.10.04 10:00 2	
size time extension name	

Problem G. Game of Xunix

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

The World Championship for Advanced Computer Misuse is being held in Flower City. The schedule includes battling in games like Qua-Qua, Half-Death, and Xunix. The winner of several previous years — Mr. Ignoramus — is not quite sure in his strength and he looks for ways to improve his performance in all these kinds of games, and especially in the game of Xunix.

The game of Xunix is played on a rectangular board of size $X \times Y$ ($2 \leq X, Y \leq 2000$), where X and Y are integers. The lower left corner of the board has coordinates $(0, 0)$ and the upper right corner has coordinates (X, Y) . The board is encircled with walls. Two antagonistic players participate in the game. The first player is given N balls of radius 1 each. He puts all the balls on the board, so the center of the i th ball has coordinates (x_i, y_i) ($0 < x_i < X$, $0 < y_i < Y$), where x_i, y_i are integers. Also, the first player assigns some speed to each ball by specifying two integer numbers v_i and w_i , where $-100 \leq v_i \leq 100$, $-100 \leq w_i \leq 100$, such that the i th ball initially has the speed vector (v_i, w_i) . When the game starts the balls start moving on the board with given speeds. The balls bounce against the walls of the board according to the laws of elastic impact but they do not interact with each other. After the game starts, the first player cannot change the speed or location of any ball.

The aim of the second player is to erect new walls on the board in order to minimize the area of the regions of the board, where the balls fly. The only operation allowed to the second player is building of a straight segment of wall connecting two walls, which are already mounted. The new wall may be either horizontal or vertical, no diagonal walls are allowed. The width of all walls is assumed to be zero. In order to erect a wall at an instant T (T is a non-negative integer) the second player chooses a point (x_T, y_T) on an existing wall and a direction r_T (0 — right, 1 — up, 2 — left, 3 — down). The new wall is erected starting from the given point and

up to another wall in the given direction. The wall building process is instantaneous. However, the wall building fails if there is a ball on the way, i. e. if the new wall would intersect a ball on the board. In this case no wall is erected at all. If the new wall only touches the balls on the board, the wall building succeeds.

Mr. Ignoramus asks you to write a program, which is given the description of the initial configuration of balls and the log of the turns made by the second player. The program should print the total area of regions free of balls, i. e. where the balls cannot fly.

Input

The input for your program is as follows. On the first line three integers X, Y, N ($1 \leq N \leq 1000$) are specified. The next N lines contain the description of the initial location and speed of the balls. Each description is placed on a separate line and consists of four integers x_i, y_i, v_i, w_i . Then the turns of the second player are specified. Each turn specification is written on a separate line and consists of four integers T, x_T, y_T, r_T . The turn specification is terminated by a line containing the sequence of four -1 . No two turns have the same T . The sequence of turns is sorted in increasing order of T . T does not exceed 1 000 000 000. The number of turns in the input does not exceed 5000.

Output

Print out the total area of all the ball-free board regions. A *ball-free region* is a region without balls inside and completely separated by walls from all the regions with balls.

Example

<i>standard input</i>	<i>standard output</i>
10 10 1	50
2 2 1 0	
0 5 0 1	
-1 -1 -1 -1	

Problem H. Hi-Wi-Fi

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

Young researchers from a far west country have recently come up with a new exciting idea about wireless networks. They have built a wireless network consisting of similar replication stations with various overlapping circular coverage areas. The new idea was to allow a user to roam along the country without having to break any active connections when he moves from one station to another.

However, not all of the country area is completely covered with wireless stations and there are several “islands” of such non-interrupted network access. Therefore there is a small chance that a user will have his connection broken when he moves. For that purpose researchers have developed an intelligent software capable of sustaining wireless connection for a limited time without a wireless station nearby. This software can only work for a limited time, therefore the total sum of distances traveled outside of coverage areas should be below some defined limit.

Your task is therefore to check whether a user can travel between his current position and a given destination point without connection breaks.

Input

The first line contains four integer numbers $X_{start}, Y_{start}, X_{end}$ and Y_{end} delimited by a single space — these are the coordinates of the user’s current position and destination respectively. The second line

contains integer N — the number of wireless stations available in a far west country ($1 \leq N \leq 1000$). Next N lines contain three integer numbers X_i , Y_i and R_i — coordinates and the coverage radius of each station respectively. The last line contains a single integer number M — the limit on total sum of distances that can be safely traveled without connection break.

All numbers X_{start} , Y_{start} , X_{end} , Y_{end} , M , X , Y , X_i , Y_i , R_i do not exceed 32 000 by absolute value.

Output

Print out a single line containing number 1 if the user can reach the destination point without ever losing his network connection and 0, otherwise.

Example

<i>standard input</i>	<i>standard output</i>
0 0 10 10 1 1 1 5 10	1
0 0 10 0 4 0 0 2 3 0 2 6 0 2 9 0 2 0	1

Problem I. Irrationality

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 64 MB

Little Alice hates when irrational things happen. One day she came home crying and showed her elder brother Bob the formula her teacher had written in school:

$$\frac{1}{\sqrt{7} + \sqrt{3}}.$$

"That irrationality in the denominator is terrible!" — she said. Bob replied that there is a simple way to make the denominator free from irrationality — we just have to multiply it and the nominator by $\sqrt{7} - \sqrt{3}$. Clearly we get

$$\frac{1}{\sqrt{7} + \sqrt{3}} = \frac{\sqrt{7} - \sqrt{3}}{4}.$$

This funny trick impressed Alice very much. But next day she brought her brother a more complex formula:

$$\frac{1}{\sqrt{2} + \sqrt{3} + \sqrt{5}}.$$

Bob realized that he has to do something if he does not want to spend the rest of his life rationalizing the fractions. So he asked you to help him and write a program which converts fractions to the irrationality-free form.

He is pretty sure the initial fraction will be of the form

$$\frac{1}{\sqrt{p_1} + \dots + \sqrt{p_n}},$$

where p_i are distinct prime numbers less than 100 and $1 \leq n \leq 7$. The answer should be written as

$$\frac{a_1\sqrt{q_1} + \dots + a_m\sqrt{q_m}}{k},$$

where k , a_i , and q_i are integers. Additionally k , a_i , and q_i should obey the following requirements:

- k , a_i , and q_i do not exceed 10^{100} by absolute value;
- k is a positive integer;
- q_i are distinct positive square-free integers (that is, none of q_i is divisible by t^2 for some integer number $t > 1$);
- $0 < q_1 < q_2 < \dots < q_m$.

Input

The first line of the input contains n . The second line contains n prime numbers $p_1, \dots, p_n$.

Output

On the first line, output the denominator k and the number m of summands in the nominator. The following m lines should contain pairs a_i, q_i (each pair on a separate line). No leading zeroes are allowed in the output.

Example

<i>standard input</i>	<i>standard output</i>
2 7 3	4 2 -1 3 1 7
3 2 3 5	24 3 6 2 4 3 -2 30

Problem J. Jth Diagonal [MODIFIED]

Input file: *standard input*
Output file: *standard output*
Time limit: 8 seconds
Memory limit: 64 MB

In Flatland citizens have to pay taxes for their apartments which are just convex polygons on the plane. Due to aesthetic reasons, all corners of the polygon belong to the same circumference. According to the local law the tax rate for a given apartment is calculated depending on the length of its j th diagonal.

That is, suppose the apartment has n corners numbered by successive natural numbers from 1 to n . For each pair (u, v) , $1 \leq u < v \leq n$ consider the segment connecting corners u and v . Such a segment is called a *diagonal*. Now arrange all such segments in non-decreasing order of lengths and look at the j th place, this is exactly the j th diagonal. The diagonals are numbered starting from 1.

A tax officer has a hard day today as she needs to check the j th diagonals for numerous tax reports presented by citizens. Help her to accomplish this sophisticated task.

Input

The first line of the input contains integers n and j ($3 \leq n \leq 10\,000$, $1 \leq j \leq n(n-1)/2$) and is followed by n lines each describing a corner of the apartment. Each corner is given by the pair x, y of its coordinates. The coordinates are real not exceeding 10^4 by absolute value. The corners are listed in clockwise order. All corners are guaranteed to belong to the same circumference.

Output

Output the length of j -th diagonal with at least 6 digits after the decimal point.

Example

<i>standard input</i>	<i>standard output</i>
4 5 0 0 0 1 1 1 1 0	1.41421356237309505