Problem A. MEX-Query

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 mebibytes

Well, you can't imagine what kind of a legend we could create for this problem. We wanted to create a long story on three pages with lots of unnecessary information. Another idea was to bind our task to some fictional situation, where two players play the game and they someway already know Grundy values for their positions... But then we remembered ***how much we dislike*** *such type of legends on contests, that have nothing common with the task, so all the spam in this statement ends in this paragraph.*

MEX (*minimum excludant*) of a finite set of nonnegative integers S is the minimal non-negative integer $x \notin S$.

You are given a 1-indexed array A consisting of N nonnegative integers, and also Q queries (L_i, R_i). For each query (L_i, R_i), find a MEX for the subsegment of this array with indices from L_i to R_i inclusive.

Input

The first line contains N, the size of the array ($1 \le N \le 1\,000\,000$).

The second line contains N integers A_i ($0 \le A_i \le 1\,000\,000$) separated by spaces.

The third line contains Q, the number of queries you need to process ($1 \le Q \le 1\,000\,000$).

The following Q lines contain the description of the queries in the form L_i R_i ($1 \le L_i \le R_i \le N$).

Output

For each query according to their order in the input, print the MEX-value of the corresponding subsegment on a separate line.

Examples

standard input	standard output
10 2 0 1 3 0 0 2 4 7 1 7 1 2 5 6 5 10 1 3 1 5 6 6 7 8	1 1 3 3 4 1 0

Problem B. Beer Quadrilaterals

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 seconds
Memory limit:	512 mebibytes

Andrew Six-Meters and Oleg sat on a two-dimensional plane and looked at four bottles of beer: A, B, C and D. They were discussing the idea of measuring angles in alcohol degrees such that the right angle consists of exactly k alcohol degrees, and other angles are measured proportionally. Suddenly, Andrew said:

"Look at these bottles! They form a strictly convex quadrilateral with a nice property: angles between all pairs of sides and diagonals contain an integer number of alcohol degrees!"

"That's pretty interesting! Also, as you can see, bottles A and C stand at points $(-1, 0)$ and $(1, 0)$ respectively, point B is above the line AC, and point D is under the line AC. I wonder how many ways are there to put bottles B and D so that they form such a type of quadrilateral."

Help friends to find an answer to their question.

Input

The input contains a single integer k ($2 \le k \le 150$), which is the number of alcohol degrees in the right angle.

Output

Output the number of ways to arrange convex quadrilaterals $ABCD$ with A and C in points $(-1, 0)$ and $(1, 0)$ respectively, B located above the line AC, and D located under the line AC, such that the nice Andrew-Oleg beer property is satisfied.

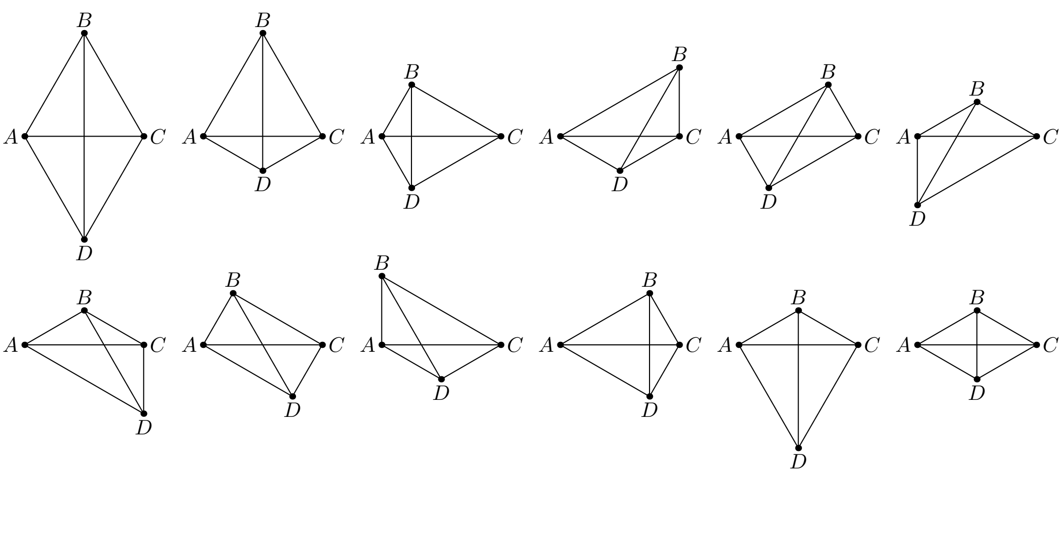
Examples

standard input	standard output
2	1
3	12

Note

In the first sample, one degree of alcohol is a half of the right angle, i. e. $\frac{\pi}{4}$ radians, so the only possible arrangement is a square with diagonal AC.

The possible arrangements for the second sample are presented below in natural scale:



Problem C. The Palindrome Extraction

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

Do you like palindromes as much as we do? Then this problem and the next one are dedicated to you.

As you know, a palindrome is a string S that is equal to the reverse of itself: $S = \text{rev}(S)$.

You are given a string S. Your task is to find five strings, A, B, C, D and E, such that the following conditions are satisfied:

1. $S = A + B + C + D + E$, where $+$ denotes string concatenation (each of the strings A, B, C, D and E can be empty).
2. $B + D$ is a palindrome.
3. $|B + D|$ is maximal possible, where $|X|$ denotes the length of string X.

One of our team members says that his grandma would solve this problem in less then a minute. What about you?

Input

The input contains the string S ($1 \le |S| \le 100\,000$). The string consists of lowercase English letters.

Output

On the first line of output, print the maximal possible value of $|B + D|$.

On the second and the third lines, print two numbers denoting 1-indexed positions of the first and the last characters of substrings B and D in string S. If one of the substrings is empty, output "`-1 -1`" for its positions.

If there are several possible answers, print any one of them.

Examples

standard input	standard output

`abcdcxxxba`	`7`
	`1 5`
	`9 10`

Problem D. Short Enough Task

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	512 mebibytes

Do you like expected values as much as we do? Then this problem and the next one are dedicated to you!

Let us generate a *random string* using the following algorithm: each of N characters of the *random string* is equiprobably chosen from the alphabet of size K.

Your task is to calculate the expected number of subpalindromes in such a *random string*.

Input

The input contains two integers N and K, the length of the string and the size of the alphabet respectively ($1 \le N, K \le 10^9$).

Output

Output the expected number of subpalindromes. Your answer will be considered correct if its absolute or relative error is less than 10^{-6}.

Examples

standard input	standard output
3 2	4.5000000000

Note

As you remember, a *subpalindrome* is a non-empty substring of the original string that can be read identically in both directions.

Problem E. Another Short Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

Consider an initial set A consisting of N points in three-dimensional space. Each point is associated with a real value p_i that is equal to the probability that i-th point is included in the final set B. Your task is to calculate the expected number of *border points* for convex hull of B.

A point $p \in B$ is called a *border point* for the convex hull of set B if there is no way to choose points $a, b, c, d \in B$ such that p lies strictly inside the tetrahedron $abcd$.

Input

The first line contains the number of points N ($1 \le N \le 50$). The next N lines describe points of A in the form p_i x_i y_i z_i, which are the probability of the i-th point to be included in the final set and the coordinates of the i-th point ($0.00 \le p_i \le 1.00$, $-1000 \le x_i, y_i, z_i \le 1000$). Probabilities are given with exactly two digits after the decimal point.

It is guaranteed that no two points of A coincide, no three points of A are collinear and no four points of A are coplanar.

Output

Output the expected number of *border points*. Your answer will be considered correct if its absolute error is less then 10^{-6}.

Examples

standard input	standard output
5 0.20 0 0 0 0.40 0 0 4 0.60 0 4 0 0.80 4 0 0 0.50 1 1 1	2.4808000000

Problem F. Just Another Sequence Problem

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

You are given a sequence $(A_1, A_2, \ldots, A_N)$ consisting of integers. Your task is to split this sequence into contiguous non-empty parts such that the value $S_1 \cdot S_2 + S_2 \cdot S_3 + \cdots + S_{k-1} \cdot S_k$ is as large as possible, where k is the total number of parts and S_i is the sum of all integers in i-th part (in the order from left to right).

Note that if $k = 1$, this value is assumed to be equal to zero.

Input

The first line contains the integer N, the length of the sequence ($1 \le N \le 2000$).

The second line contains integers $A_1, A_2, \ldots, A_N$ separated by spaces ($-1000 \le A_i \le 1000$).

Output

Output the maximal possible value of $S_1 \cdot S_2 + S_2 \cdot S_3 + \cdots + S_{k-1} \cdot S_k$.

Examples

standard input	standard output
6 2 -1 4 3 -1 0	13

Note

In the example case, the optimal partition is $(2, -1), (4), (3), (-1, 0)$, it produces the value $S_1 \cdot S_2 + S_2 \cdot S_3 + S_3 \cdot S_4 = 1 \cdot 4 + 4 \cdot 3 + 3 \cdot (-1) = 13$.

Problem G. Total LCS

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

Let us define a function $\mathrm{LCS}(S, T)$ to be the length of the *longest common subsequence* of two strings S and T. More formally, $\mathrm{LCS}(S, T)$ is the length of the longest (possibly empty) string which is a subsequence of S and a subsequence of T. A string A is a subsequence of a string B if A can be obtained from B by erasing some (possibly none) characters (without permuting them!).

You are given two strings S and T. For a pair (i, j) where $1 \le i \le j \le |T|$, let us define $T[i..j]$ to be a substring of T consisting of symbols on positions from i to j inclusive.

Calculate all values of $\mathrm{LCS}(S, T[i..j])$.

Input

The two lines of input contain the strings S and T ($1 \le |S|, |T| \le 2000$). The strings consist of lowercase English letters.

Output

Output the values of $\mathrm{LCS}(S, T[i..j])$ over all possible pairs (i, j) according to the lexicographical order of pairs (i, j).

The checking program ignores all whitespace (including line breaks), so it is up to you to format the output like a table (as we did in the example) or otherwise.

Examples

standard input	standard output

abac
cbabc
	1 1 2 2 3
	  1 2 2 3
	    1 2 3
	      1 2
	        1

Problem H. Cyclic String

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

You are given N characters written on a circle. Let us split this circle in K places so that no character is cut apart and no two cuts pass through the same place. After this operation K strings $S_1, S_2, \ldots, S_K$ are formed by reading characters on each part clockwise.

Your task is to find such a way of splitting that $\max\{S_1, S_2, \ldots, S_K\}$ is minimal possible. Note that $S_1, S_2, \ldots, S_K$ are strings that are compared *lexicographically*.

Input

The first line contains N lowercase English letters ($1 \le N \le 2000$), which are the characters written on circle in clockwise order.

The second line contains the integer K ($1 \le K \le |S|$).

Output

Output K numbers: 1-based starting positions of substrings that form the answer in increasing order.

Examples

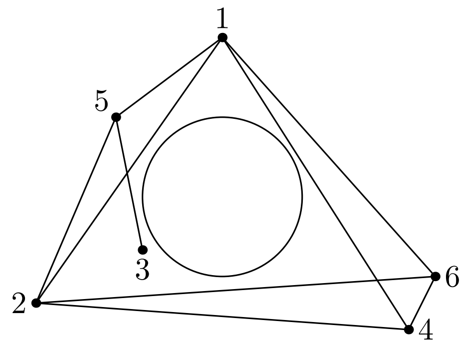
standard input	standard output
abcaab 1	4
abacabadaba 5	1 3 5 9 11

Problem I. Circle Clique

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Do you think that nothing can really surprise you on a programming contest? Are you sure? Then how would you like this! Given a graph consisting of N vertices, find a maximal clique in it, where N is up to 5000!

...that's how the statement could look if this was "Robert Tarjan Contest". Unfortunately, it is only "MSU Trinity Contest", therefore we should add a bit more restrictions so that at least somebody could solve this problem.



You are given a disk of radius R with its center at the origin and N integer points outside that disk. Let us consider a graph on these points as vertices, where points A and B are connected by an edge if and only if the line AB does not intersect the disk.

Find the maximal clique in this graph.

Input

The first line contains two integers N and R, the number of points and the radius of the disk respectively ($1 \le N \le 2000$, $1 \le R \le 5000$).

The following N lines contain the coordinates of points in the form x_i y_i ($-5000 \le x_i, y_i \le 5000$).

It is guaranteed that all points are different, each point lies strictly outside the disk, and no two points lie on a common tangent line to the disk.

Output

On the first line, print the size of the maximal clique. On the second line, print the numbers of points forming that clique. If there are several possible answers, print any one of them.

Examples

standard input	standard output
6 3 0 6 -7 -4 -3 -2 7 -5 -2 3 8 -3	4 1 2 6 4

Problem J. Dividing Area

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

Once upon a time there lived an angry university teacher. He liked to punish students in different ways, giving them tasks they were not able to solve. The more complex the task was, the more excitement he gained. One day he was especially happy...

First he marked N red points on an infinite board. Then he started asking Oleg two types of queries.

Type 1. Draw a segment connecting red points a and b. It is guaranteed that this segment doesn't contain other red points and inner points of the already drawn segments.

Type 0. Calculate the area of the connected region adjacent to the *left* side of some already drawn segment ab. Here, *left* means the direction that is rotated 90° counter-clockwise from the direction ab.

This teacher managed to ask Q queries. Note that the teacher was very cruel and he stopped asking only when there was no way to draw a new segment satisfying the conditions above. Nevertheless Oleg successfully answered all his questions!

Can you repeat his achievement?

Input

The first line contains the integer N, the number of red points on the board ($1 \le N \le 200\,000$).

The following N lines contain the coordinates of red points in the form x_i y_i ($-2 \cdot 10^8 \le x_i, y_i \le 2 \cdot 10^8$). Points are numbered from 0 to $N-1$.

The next line contains an integer Q, the number of queries ($1 \le Q \le 1\,000\,000$).

The remaining Q lines contain the queries in the form t_i a_i b_i which are the type of the query and the numbers of endpoints of the segment in the query ($0 \le a_i, b_i < N$, $a_i \ne b_i$).

Output

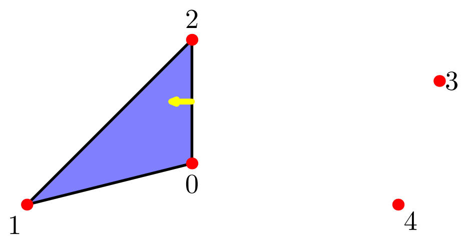
For each query of type 0, output an integer number, which is the **doubled** area of the region located to the left from the segment if it is finite, or -1 if it is not.

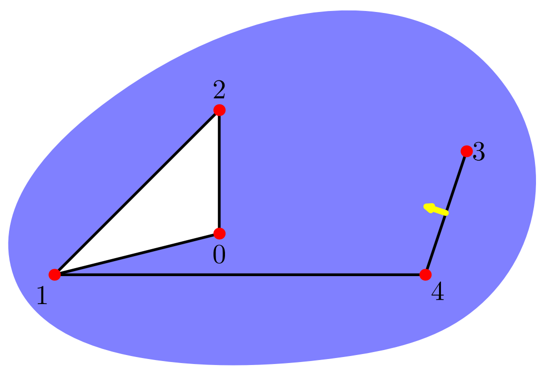
Examples

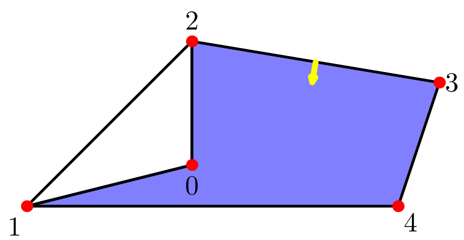
standard input	standard output
5 -1 0 -5 -1 -1 3 5 2 4 -1 14 1 0 1 1 2 1 1 2 0 0 0 2 1 4 1 1 3 4 0 4 3 1 3 2 0 3 2 1 2 4 0 3 2 0 4 2 1 4 0 0 2 4	12 -1 43 19 24 19

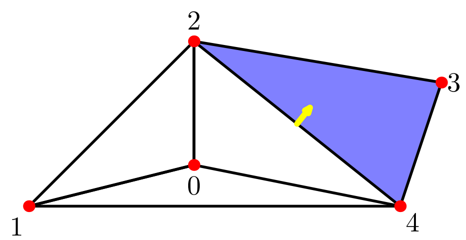
Note

The following pictures given below clarify some queries of the example test. The queries are numbered from zero; pictures are related to queries 3, 6, 8, 13 respectively.









Problem K. Tree Problem

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

You are given a tree consisting of N vertices numbered from 1 to N.

A *simple path* P between vertices a and b is a sequence of k vertices $(a = P_1, P_2, \ldots, P_k = b)$ such that any two consequent points are connected by an edge and each vertex appears in the path at most once. Note that a end b can be equal. We also say that an edge lies on a simple path if it connects two subsequent vertices of that path.

The *neighborhood* of a simple path P consists of all edges that have exactly one endpoint in P.

Each edge can be either blocked or unblocked. Initially, all edges are blocked. Write a program that handles Q queries of two types:

0. Count how many blocked edges lie on the simple path between vertices a and b.
1. Unblock all edges lying on the simple path between vertices a and b and block all edges from neighborhood of that path.

Input

The first line contains number of vertices N $(1 \le N \le 200\,000)$.

Each of the following $N - 1$ lines contains a description of an edge in the form a_i b_i. It is guaranteed that the given graph is a tree.

The next line contains number of queries Q $(1 \le Q \le 300\,000)$.

The remaining Q lines contain queries in the form t_i a_i b_i, denoting the type of the query and the endpoints respectively.

Output

For each query of type 0, output the answer to the query on a separate line.

Examples

standard input	standard output
19 1 2 2 3 1 5 5 4 5 6 6 7 6 8 1 11 11 12 11 13 11 10 10 9 13 14 13 15 15 16 15 17 15 18 15 19 6 1 19 8 0 16 2 0 16 3 1 12 9 0 19 8 0 16 9	2 3 2 2