Problem A. Balance

Input file:	`balance.in`
Output file:	`balance.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You have n coins, exactly one of those being fake (weighing different from all others). You have a balance which allows you to compare two groups of coins. Determine the fake coin in such a way that the number of weighings in the worst case is minimum possible.

Adhere to the output format shown below. Note that you must not describe impossible cases (i.e., in the example output there is no «`case =`» for «`weigh 1+2 vs 3+4`» because the pans will never be equal there). For every weighing the numbers of coins on both pans should be equal, and no coin should be simultaneously put on both pans.

Input

The input file contains the only integer n ($3 \le n \le 100$).

Output

Output the required script.

Example

balance.in	balance.out

4	need 2 weighings
	weigh 1 vs 2
	case <:
	  weigh 1+2 vs 3+4
	  case <:
	  fake 1
	  case >:
	  fake 2
	  end
	case =:
	  weigh 1 vs 3
	  case =:
	  fake 4
	  case <:
	  fake 3
	  case >:
	  fake 3
	  end
	case >:
	  weigh 1+2 vs 4+3
	  case >:
	  fake 1
	  case <:
	  fake 2
	  end
	end

Problem B. Cipher

Input file:	`cipher.in`
Output file:	`cipher.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

ASN has just invented a brand new cipher. Its key is just a $H \times W$ matrix of 0's and 1's. A tool by Macrosoft is recommended to be used as a manager of those keys. This tool stores a fingerprint for each key to protect from storage failures. Such a fingerprint is an $(H-1) \times (W-1)$ matrix consisting of 2×2 sums; i.e., if A is the key and B is the fingerprint, then $B_{ij} = A_{ij} + A_{i+1,j} + A_{i,j+1} + A_{i+1,j+1}$. Given the fingerprint, you are to find at least one key with such fingerprint, or to report that the fingerprint is corrupt (in case no key can produce it).

Input

The first line of the input file contains two numbers, H and W ($2 \le H, W \le 300$). The next $H-1$ lines contain $W-1$ characters each with no spaces in between, describing the fingerprint. Each of those characters will be either `0`, `1`, `2`, `3`, or `4`.

Output

Output the key using the format similar to that of the input file: output H lines containing W characters (`0` or `1`) each, with no spaces in between.

If the fingerprint is corrupt, output `CORRUPT` on the only line of output.

Example

cipher.in	cipher.out
3 4 222 222	0110 1001 0110

Problem C. Hyperboloid Distance

Input file:	distance.in
Output file:	distance.out
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given two points A and B on the surface of the hyperboloid $x^2 + y^2 - z^2 = 1$. Find the shortest distance between them along the surface.

Input

The input file contains six real numbers: x_A, y_A, z_A, x_B, y_B and z_B (coordinates of A and B), separated by spaces and/or line breaks ($-1 \le z_A, z_B \le 1$).

Output

Output the shortest distance between A and B along the surface of the hyperboloid. Your answer should be accurate within ± 0.1.

Example

distance.in	distance.out
1 0 0 0 1 0	1.5707963267

Problem D. Real Fun

Input file:	`fun.in`
Output file:	`fun.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Yesterday it was real fun.

Today you wake up and notice that something is just not right. Not just headache, but something else is constantly annoying you. But you just can't get what exactly. You walk around your room, enjoying the sunlight coming through the open window, through the holes in the roof... Stop. There were no holes in the roof until today. Definitely.

Suppressing the urge to call your friends and find out something about the origin of the holes, you decide to fix the roof first. In a modern way.

You've decided to nail 3 equal square boards with sides parrallel to the sides of the (of course, square) roof to close all the holes, and were just wondering what is the minimal required size for these boards.

Formally speaking, you are given n different points on a Cartesian plane and need to find minimal d such that three possibly overlapping $d \times d$ squares with sides parallel to coordinate axes can cover all the points (possibly just by the border).

Input

The first line of the input file contains n ($4 \le n \le 20000$).

The next n lines contain two integer numbers each, x and y — the coordinates of the holes ($-10^9 \le x, y \le 10^9$). No two points coincide.

Output

Output the minimal possible d.

Example

fun.in	fun.out
4 0 1 0 -1 1 0 -1 0	1
12 0 1 0 -1 1 0 -1 0 10 1 10 -1 11 0 9 0 20 1 20 -1 21 0 19 0	2

Problem E. Hippopotamus

Input file:	`hippo.in`
Output file:	`hippo.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

After fixing your roof, you still think that it looks unpretty. So you opt for a new one, consisting of n consecutive long narrow boards. You have two types of boards: wooden ones and iron ones, giving you an amazing total of 2^n possible roofs.

But the safety should not be left aside. Having considered the weight and the cruising speed of a falling hippopotamus, you decide to have at least k iron boards among every m consecutive boards.

How many possibilities do you have?

Input

The input file contains three integers, n, m and k, separated by spaces and/or line breaks. $1 \le n \le 60$, $1 \le m \le 15$, $0 \le k \le m \le n$.

Output

Output the number of possibilities.

Example

hippo.in	hippo.out
10 2 1	144
5 5 2	26
3 2 2	1

Problem F. Ice-cream Tycoon

Input file:	`icecream.in`
Output file:	`icecream.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You've recently started an ice-cream business in a local school. During a day you have many suppliers delivering the ice-cream for you, and many students buying it from you. You are not allowed to set the prices, as you are told the price for each piece of ice-cream by the suppliers.

The day is described with a sequence of queries. Each query can be either `ARRIVE` n c, meaning that a supplier has delivered n pieces of ice-cream priced c each to you, or `BUY` n t, meaning that a student wants to buy n pieces of ice-cream, having a total of t money. The latter is processed as follows: in case n cheapest pieces of ice-cream you have cost no more than t (together), you sell those n cheapest pieces to the student; in case they cost more, she gets nothing. You start the day with no ice-cream.

For each student, output `HAPPY` if she gets her ice-cream, and `UNHAPPY` if she doesn't.

Input

The input file contains between 1 and 10^5 queries (inclusive), each on a separate line. The queries are formatted as described above, either `ARRIVE` n c or `BUY` n t, $1 \le n, c \le 10^6, 1 \le t \le 10^{12}$.

Output

For each `BUY`-query output one line, containing either the word `HAPPY` or the word `UNHAPPY` (answers should be in the same order as the corresponding queries).

Example

icecream.in	icecream.out
ARRIVE 1 1 ARRIVE 10 200 BUY 5 900 BUY 5 900 BUY 5 1000	HAPPY UNHAPPY HAPPY

Problem G. 4-3 King

Input file:	king.in
Output file:	king.out
Time limit:	2 seconds
Memory limit:	64 megabytes

The king of the Quadroland (or was it Triland?...) has died. After four or three days of mourning, his will was declared: all his kingdom was to be divided into four or three parts, one for each son. The kingdom itself can be represented as a quadrangle or a triangle on a plane, with each side associated with one of the sons. After the division is performed, each son should get a quadrangle or a triangle, one of the sides coinciding with his associated side of the original kingdom. The required ratio between areas of their parts is given. You are to perform such a division.

Input

The first line of the input file contains the number N ($3 \le N \le 4$) of sons (equal to the number of sides of the kingdom).

The next N lines contain the coordinates of the vertices of the kingdom. The vertices are given in either clockwise or counter-clockwise order. No two consecutive sides of the kingdom lie on the same line. Each coordinate is an integer not exceeding 100 by its absolute value.

The last, $(N+2)$-th line contains the required ratio formatted like $K_1 : K_2 : \ldots : K_N$, where K_1 corresponds to the side between 1st and 2nd vertices, K_2 — between 2nd and 3rd vertices, etc, K_N — between Nth and 1st vertices. Each K_i is an integer, $1 \le K_i \le 100$.

Output

The first line of the output file should contain the description of the part corresponding to the side between 1st and 2nd vertices, the second line — between 2nd and 3rd vertices, etc.

Each description should consist of the number of vertices in the polygon (3 or 4), followed by their coordinates, in either clockwise or counter-clockwise order. Two ends of each side should be some neighbouring vertices of the corresponding polygon. The coordinates can be real numbers; if this is the case, print as many digits after the decimal point as possible.

In case the division is impossible, output -1 on the only line of output.

Example

king.in	king.out
3 0 0 10 0 0 10 4:2:4	3 0 0 10 0 4 4 3 10 0 0 10 4 4 3 0 10 0 0 4 4
4 0 0 0 30 100 30 100 0 1:3:1:3	3 0 0 0 30 25.0 15.0 4 0 30 100 30 75.0 15.0 25.0 15.0 3 100 30 100 0 75.0 15.0 4 0 0 100 0 75.0 15.0 25.0 15.0

Problem H. Circular Railway

Input file:	railway.in
Output file:	railway.out
Time limit:	2 seconds
Memory limit:	64 megabytes

There are L stations along a circular railway, numbered 1 through L. Trains travel in both directions, and take 1 minute to get from a station to the neighbouring one (i.e., between 1st and 2nd, between 2nd and 3rd, ..., between $(L-1)$-th and L-th and between L-th and 1-st).

There are n employee's houses along the railway, and n offices, each house or office located near a railway station. You are to establish a one-to-one correspondence between houses and offices in such a way that total travel time (sum of travel times of each employee) is minimized.

Input

The first line of the input file contains two integer numbers, n and L ($1 \le n \le 50000$, $2 \le L \le 10^9$). The second line contains n locations of the employee's houses, and the third line contains n locations of the offices. Each location is an integer number between 1 and L. Some houses or offices or both can be located at the same railway station.

Output

Output the minimal total travel time followed by the description of the one-to-one correspondence. The description should be represented by n numbers (one for each employee, ordered as in the input), denoting the 1-based index of the office assigned to the corresponding employee.

Example

railway.in	railway.out
3 15 1 2 10 11 12 13	9 2 3 1
4 12 2 5 8 11 6 9 12 3	4 4 1 2 3

Problem I. Shortest Paths

Input file:	`shortest.in`
Output file:	`shortest.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given a graph with one vertex marked as source s and one as destination t. Each edge of the graph has a positive length. Find the shortest path from s to t. Then find the second-shortest path (the shortest one of all the paths from s to t except the one you've just found). Then the third-shortest, and so on. Output the lengths of first k such paths.

Note that these paths may not be simple, i.e. they may contain some vertex or edge several times (see the 2nd example).

Input

The first line of the input file contains n, the number of vertices of the graph, m, the number of edges of the graph, and k, the number of paths sought ($2 \le n \le 10000$, $2 \le m \le 50000$, $2 \le k \le 10000$).

The second line of the input file contains s and t (integers between 1 and n, inclusive, $s \ne t$).

The next m lines contain the descriptions of the edges, each description consisting of three integer numbers: a b c, denoting the edge from a to b with length c ($1 \le a, b \le n$, $a \ne b$, $1 \le c \le 1000$). There may be more than one edge for the same a and b.

Output

Output k integer numbers in non-decreasing order — the lengths of the paths. In case there are less than k different paths from s to t, output `NO` instead of the lengths of all non-existent paths.

Example

shortest.in	shortest.out
4 5 5 1 4 1 2 1 2 3 1 3 4 1 1 3 1 2 4 1	2 2 3 NO NO
4 4 5 1 4 1 2 10 2 3 10 3 4 10 3 2 10	30 50 70 90 110
2 2 10 1 2 1 2 5 2 1 7	5 17 29 41 53 65 77 89 101 113