Problem A. Antiufology

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

Given a (not necessarily convex) polygon, find a set of K horizontal lines that divide it into $K + 1$ (not necessarily connected) parts of equal area.

The polygon will not have any degenerate vertices — in other words, each two consecutive sides will not belong to the same line. Horizontal lines are those containing all points with the same Y-coordinate.

Input

The first line of the input file contains two integers: N and K, $3 \leq N \leq 50\,000$, $1 \leq K \leq 50\,000$, where N is the number of vertices of the polygon, and K is the number of lines to produce.

The next N lines contain the vertices of the polygon in the clockwise order. Each vertex is defined by its coordinates X and Y not exceeding 10000 by absolute value.

Output

The output file should contain K lines, each of those containing the Y-coordinate of one horizontal line. Those coordinates should be in inreasing order.

Your output will be considered correct if each number is within 10^{-4} of the correct answer.

Sample input and output

antiufo.in	antiufo.out

8 1
1 1
1 4
3 4
3 0
-3 0
-3 4
-1 4
-1 1
	1.7500

Problem B. Fire

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

In this problem, we're concerned with predicting the behavior of a forest fire.

The forest is given as a convex polygon, and the fire has started simultenously in several points inside this polygon, and is spreading from each of those points in all directions with the same constant speed.

Which point in the forest will be the last to catch fire?

Input

The first line of the input file contains the number n $(3 \le n \le 200)$ of the vertices of the polygon. The next n lines contain two integers each, x_i and y_i — the coordinates of the vertices, given either in clockwise or in counter-clockwise order. $(n + 2)$-th line contains the number k of fire starting points $(1 \le k \le 200)$. The last k lines contain two integers each, x_j and y_j — the coordinates of fire starting points. All coordinates don't exceed 1000 by absolute value.

Output

Output the coordinates of a point inside or on the border of the forest that will be the last to catch fire. If there are several such points, output any. Your output will be considered correct if each coordinate is within 10^{-4} of the correct one.

Sample input and output

fire.in	fire.out

4
0 0
4 -4
8 0
4 4
4
1 0
7 0
4 3
4 -3
	4.0000 0.0000

Problem C. Formation

Input file:	`form.in`
Output file:	`form.out`
Time limit:	4 seconds
Memory limit:	64 megabytes

Given a permutation of integers between 1 and N, count how many of its subsequences of length $K + 1$ are increasing.

The answer is guaranteed not to exceed $8 \cdot 10^{18}$.

A subsequence of a sequence is the result of removal of some (maybe none, maybe all) its elements.

Input

The first line of the input file contains N ($1 \le N \le 100\,000$) and K ($0 \le K \le 10$). The next N lines contain one integer each, describing the given permutation. Each number between 1 and N will appear exactly once among those lines.

Output

Output one integer — the sought number of subsequences.

Sample input and output

form.in	form.out

5 2
1
2
3
5
4
	7

Problem D. King's travel

Input file:	`king.in`
Output file:	`king.out`
Time limit:	9 seconds
Memory limit:	64 megabytes

A lone king is on cell (x_s, y_s) of an infinite chessboard, and he wants to get to cell (x_t, y_t). During one step, he can move to any of the 8 cells that are adjacent to his current cell.

However, there are some cells that are occupied by other chess pieces and thus the king can't move there. Please help the king to find the shortest path to the destination cell that does not visit any of the occupied cells.

Input

The first line of the input file contains the starting cell coordinates x_s, y_s. The second line of the input file contains the destination cell coordinates x_t, y_t.

The next line contains n — the number of occupied cells ($0 \le n \le 500$). The next n lines contain two integers each, the coordinates of the occupied cells.

All coordinates don't exceed 10^9 by absolute value. All occupied cells are different, and the starting cell is not occupied. The destination cell, unfortunately, can be occupied.

Output

Output one integer — the smallest number of steps that the king needs to get to the destination cell. When this is impossible, output -1.

Sample input and output

`king.in`	`king.out`

`0 0`
`2 2`
`1`
`1 1`
	`3`

`1 1`
`4 6`
`7`
`0 0`
`0 1`
`0 2`
`2 0`
`2 1`
`2 2`
`1 2`
	`9`

Problem E. Necklace

Input file:	`necklace.in`
Output file:	`necklace.out`
Time limit:	3 seconds
Memory limit:	64 megabytes

You are given a necklace of n precious stones arranged in a circle. i-th precious stone in the necklace costs a_i.

You need to cut the necklace at k points so that it is split into k consecutive fragments in such a way that the total costs of the fragments are as fair as possible. More spceifically, you should minimize the difference between the maximal total cost of a fragment and the minimal total cost of a fragment.

Input

The first line of the input file contains two integers n and k ($1 \le k \le n \le 50$). The second line of the input file contains n integers $a_1, a_2, \ldots, a_n$ — the costs of the stones in the necklace ($1 \le a_i \le 10^6$).

Output

On the first line of the output file print the minimal possible difference.

The next k lines should contain several integers each. Each line should contain the numbers of the stones that comprise one fragment, in any order. Stones are numbered from 1 to n.

Sample input and output

necklace.in	necklace.out

5 3
1 9 2 7 6
	2
	1 5
	2
	3 4

Problem F. Rectangles

Input file:	`rects.in`
Output file:	`rects.out`
Time limit:	3 seconds
Memory limit:	64 megabytes

You are given n rectangles on a plane with sides parallel to coordinate axes and with no two rectangles having a common point. Each rectangle is assigned an integer.

We define rectangle B to lie *further* than rectangle A when the top left corner of rectangle B is strictly to the right and to the bottom of the bottom right corner of rectangle A.

We define a sequence of rectangles $R_1, R_2, \ldots, R_k$ to be a *chain*, when for all i rectangle R_i lies further than rectangle R_{i-1}. We define the *weight* of a chain to be the sum of the numbers assigned to its rectangles.

Find the chain of rectangles with maximal weight.

Input

The first line of the input file contains n — the number of rectangles ($1 \le n \le 100\,000$).

Let the x axis be directed from left to right, and the y axis be directed from bottom to top. The next n lines contain five integers each, describing the rectangles. Each rectangle is described by the coordinates $x_{i,1}, y_{i,1}$ of its bottom left corner, the coordinates $x_{i,2}, y_{i,2}$ of its top right corner, and its assigned number a_i.

Coordinates don't exceed 10^9 by absolute value. Numbers assigned to rectangles are positive and don't exceed 10^9. No rectangle lies inside another rectangle, and no two rectangles intersect.

Output

On the first line of the output file print one integer — the maximal possible weight of a chain of rectangles. On the second line of the output file print space-separated numbers of rectangles that form such chain, in the order they appear in the chain. When there are several possible solutions, output any.

Sample input and output

rects.in	rects.out

4
1 1 2 2 6
3 1 4 2 5
0 3 1 4 5
5 1 6 2 4
	10
	3 2