Transformation

Ivancho is really keen on watching movies. Moreover he loves watching how the movies are made, especially the movies he has watched. What impress him the most is how using computer graphics and other complicated technics, a given picture can be transformed to into completely different one. For example, you can see giant robots walking on the streets that is absolutely imposibble.

That's how Ivancho found his new hobby. He takes two pictures in RGB format (each pixel is described by three values – the amount of red, green and blue color inside it). The two pictures have equal sizes. Then he tries to make the first picture as close as it is possible to the second one using same operation for number of times. The operation is considered as it follows:

1) We chose two rectangular sectors of pixels with equal sizes from the first picture.
2) We average the first sector with the second one. In other words, each pixel in the first sector gets new value for each of the three colors. This value equals to the average value of its moment value and the value of the pixel which corresponds to it if we place the two sectors one over another.
3) The second sector doesn't change.

Distance between two pictures is defined as a sum of distances between each two pixels with equal coordinates. The distance between two pixels equals to square root of the sum of squares of differences between the values for red, green and blue color for the two pixels. In other words, if the values for red, green and blue color for the first pixel are r1, g1, b1 and r2, g2, b2 and the differences are r = r1-r2, g = g1-g2, b = b1-b2 . Then the distance between these two pixels equals to *sqrt (r*r + g*g + b*b)*.

Ivancho spent a lot of time using the operation and realized that transforming one picture into another isn't so trivial. Now he wants to check how you will solve this problem. Write a program **transformation** that makes a picture as similar as it is possible to another one using the defined operation.

Input: On the first row of the input file **transformation.in** will be written three numbers **H**, **W** and **K** – the height and width in pixels of the two pictures and the maximum number of operations that you are allowed to use. Follow six tables HxW - R1, G1, B1, R2, G2, B2 that describe the values for red, green and blue color contained in each pixel in the two pictures. The first three tables describe the first picture and the second three tables describe the second picture. For each picture the first table corresponds with the red color, the second table with the green color and the third table with the blue color.

Output: The first row of the output file **transformation.out** must contain a single number **P** - the number of operations used by your program. Follow P rows describing each operation. Each row must contain six numbers x1, y1, x2, y2, n, m – the coordinates of upper-left corner of the first rectangle, the coordinates of upper-left corner of the second rectangle, the height and width of the two rectangles in pixels.

Constraints:

0 <= R1[i][j], R2[i][j], G1[i][j], G2[i][j], B1[i][j], B2[i][j] <= 255

H, W = 200

1 <= K <= 10000

The sum of the arrays of all rectangles in the output must be less or equal to 100*H*W

Grading:

If the output file contains incorrect operation your solution will receive 0 points. Otherwise, it will be graded by the formula $(best/yours)^2$. Where yours is the distance between the two pictures after applying the operations of your solution and best is the minimum distance achieved by any of the contestants on this test case.
Time limit: 15 sec.