2001/2002 ACM SOUTHERN CALIFORNIA REGIONAL SCHOLASTIC PROGRAMMING CONTEST

Problem 5
GIF Merge

GIF files are a popular way of storing images, but, being an old format, they suffer from some of the limitations of the display hardware that was available at the time the format was devised. One of these limitations is a maximum color palette of 256 colors. This means that from the millions of colors that can be represented using 8 bits each for red, green and blue, at most 256 of them are used to represent every color in a single GIF image. Through a clever choice of these colors, and a technique (called dithering) for approximating the missing colors, a reasonably good quality image can still be produced.

We want to be able to insert one GIF into another GIF as a small inset image, using the palette of the larger GIF, and using dithering to represent the colors from the smaller GIF that are not shared by the two GIFs. Your task is to write a program to tell us if this is possible or not.

When a pixel must be assigned the wrong color because the right color is not in the color palette, dithering is used to adjust the colors of the pixels around it, so that the average color around that pixel is close to the correct one. For example, if the desired color is redder than the color chosen from the palette, the surrounding pixels are made redder than they should have been to compensate for it. Since the colors of these pixels must also be chosen from the palette, the question is whether the desired color can be obtained by averaging colors chosen from the palette.

Each color is given as a triplet (r, g, b) of intensities for red, green and blue, so it can represent a point on a 3-dimensional grid, with the intensity of red on the X axis, green on the Y, and blue on the Z. Colors in the palette are fixed points on this grid, while an average color can be viewed as a movable point whose position is determined by the pull of perfect springs attached between it and the palette colors that make up the average. If palette colors surround the desired color, no matter how remotely, springs can be attached that will pull the average there. But, if all of the palette colors are to one side of the desired color, so that a plane could be interposed between it and them, then no spring can cross the plane to pull the average there.

Take a palette with just the colors (64,64,64), (255,64,64), (64,255,64) and (64,64,255). Interpreted as points, the colors define a triangular pyramid whose sides are the planes $red = 64$, $green = 64$, $blue = 64$ and $red + green + blue = 383$. If a color is outside this pyramid, a plane can be placed between it and the pyramid, separating it from all of the colors that define the pyramid. However, if a color is on the surface or inside the pyramid, no plane can separate it from every color that defines the pyramid. Thus, for this example, the color (r, g, b) can be dithered if $r \geq 64$ and $g \geq 64$ and $b \geq 64$ and $r + g + b \leq 383$. It follows that the color (127,127,127) can be dithered using this color palette, while the colors (16,64,106) and (130,130,130) can not.

Your input consists of one or more sets of color palettes, multiple sets being separated from each other by empty lines, and the last set being followed by end-of-file. In each set, the first color palette is the palette for the main GIF, while those immediately following it without any empty lines, are the color palettes for one or more inset GIFs. Each color palette begins with a line of at most 255 bytes, containing the name of the GIF. The next line contains a single integer from 1 to 256, giving the number of colors in the palette. Each succeeding line represents one of these colors, as many lines as there are colors in the palette. These lines each contain three integers that represent the intensities of red, green and blue in the color. The integers range from 0 to 255, and are separated from each other by any combination of spaces or tabs.

The output for each set of palettes will be a list of the names of the GIFs in that set, in the same order as the input, one name per line, with nothing preceding or following the name, except that the name of each inset GIF will be followed by a single space followed by the text '`yes`' if every color in that palette can be dithered from colors in the main GIF's palette, or the text '`no`' if any of the colors in that palette cannot be dithered from colors in the main palette. The output for each successive set of palettes will be separated from the previous set by a single empty line.

Problem 5
GIF Merge (continued)

Sample Input

```
LovelyScene.gif
4
64 64 64
255 64 64
64 255 64
64 64 255
GreenishBlue.gif
1
16 64 106
Gray130.gif
1
130 130 130
Gray127.gif
1
127 127 127
LineDrawing.gif
2
64  64 106
64 106  64
AnotherLineDrawing.gif
2
88 123 75
96 12 167

FunnyImage.gif
6
25 35 45
22 32 42
20 30 40
25 35 45
23 33 43
30 40 50
FirstColor.gif
1
10 20 30
FunnyInset.gif
3
20 30 40
25 35 45
30 40 50
WhatAnotherColor.gif
1
40 50 60
HowManySingleColorGifsCanYouStand.gif
1
30 20 10
```

Problem 5
GIF Merge (still continued)

Sample Output

```
LovelyScene.gif
GreenishBlue.gif no
Gray130.gif no
Gray127.gif yes
LineDrawing.gif yes
AnotherLineDrawing.gif no

FunnyImage.gif
FirstColor.gif no
FunnyInset.gif yes
WhatAnotherColor.gif no
HowManySingleColorGifsCanYouStand.gif no
```

Useful Formulas

The vector from point (x_1, y_1, z_1) to point (x_2, y_2, z_2) is $(x_2 - x_1, y_2 - y_1, z_2 - z_1)$.

The dot product of two 3-dimensional vectors is the product of the lengths of the two vectors with the cosine of the angle between them. It can be computed as: $(x_1, y_1, z_1) \bullet (x_2, y_2, z_2) = x_1x_2 + y_1y_2 + z_1z_2$.

The cross product of two 3-dimensional vectors is a vector perpendicular to both vectors, whose length is the product of the lengths of the two vectors with the sine of the angle between them, and whose direction would be to your left if you stood so the first vector pointed up and you faced in the direction of the angle between the vectors. It can be computed as: $(x_1, y_1, z_1) \times (x_2, y_2, z_2) = (y_1z_2 - z_1y_2, z_1x_2 - x_1z_2, x_1y_2 - y_1x_2)$.

You can test if three points are collinear by taking the cross product of the vector from the first point to the second with the vector from the first point to the third, and checking that all three components of the cross product are 0.

A recursive definition for computing the value of a determinant of order n is

$$|\, a_{11} \,| = a_{11}$$

$$D = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix} = \sum_{j=1}^{n} a_{1j} \cdot \text{cofactor } A_{1j}$$

where the cofactor A_{ij} of the element a_{ij} of determinant D is defined to be the product of $(-1)^{i+j}$ with the determinant derived from D by deleting the ith row and the jth column.

The value of the determinant $\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$ is twice the (signed) area of the triangle defined by the points (x_1, y_1), (x_2, y_2) and (x_3, y_3), where the sign is negative if the points define the triangle in a clockwise direction, and positive if counter-clockwise.

The value of the determinant $\begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{vmatrix}$ is six times the (signed) volume of the triangular pyramid defined by the points (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3) and (x_4, y_4, z_4), where the sign is negative if (viewed from point 4) the first 3 points define the triangle in a clockwise direction, and positive if counter-clockwise.