Problem 239: Highlight of My Day

Difficulty: Hard

Author: Gary Hoffmann, Denver, Colorado, United States

Originally Published: Code Quest 2024

Problem Background

In computer graphics, shaders are a special type of program that run directly on the GPU (Graphics Processing Unit) in order to perform specific tasks often related to computer graphics or massively parallel computations like artificial intelligence neural networks.

Your team is working with Lockheed Martin's Aeronautics company to develop a virtual reality trainer for aircraft maintainers. The training software is required to highlight objects the maintainer is required to interact with in order to compete their task. Your team has been asked to develop a graphics algorithm to accomplish this. The software is already able to generate graphics to show these objects; your team just needs to highlight and outline them.

Problem Description

Your program will read in a highlight color followed by a pair of 2-dimensional arrays, representing an "image mask" and the content of the image. These arrays will be the same size, but can be any size, and each element represents a single pixel within the image.

The "image mask" is used to identify which parts of the image contain the object you should be highlighting. Each pixel will be represented by a 1 or a 0; a 1 indicates that that pixels is part of the object, and a 0 indicates it is part of the background.

The second array represents the image itself, which must be modified to include the object's highlighting. Each element of this array contains three integer values, between 0 and 255 inclusive, representing the red, green, and blue color components (respectively) of that pixel. For example, a pixel might be represented as 128,154,205 if it has a red component of 128, a green component of 154, and a blue component of 205. The highlight color will also be represented as a triplet of RGB component values.

Your algorithm must first determine which pixels should be modified, and to what degree, using the image mask:

- If the pixel is part of the object, it should have 50% (0.5) of the highlight color.
- If the pixel is part of the background but is adjacent - including diagonally - to at least one object pixel, it will become part of the border and should have 100% (1.0) of the highlight color.
- Other background pixels should have 0% (0.0) of the highlight color.

Once the percentage is determined, the result color of each pixel can be calculated according to the following formulas, where P represents the highlight percentage (determined above), C represents the current pixel color, H represents the highlight color, and R represents the result color. C, H, and R are each separated into their RGB color components; for example, C_r, C_g, and C_b.

$$R_r = (1.0 - P) \times C_r + P \times H_r$$
$$R_g = (1.0 - P) \times C_g + P \times H_g$$
$$R_b = (1.0 - P) \times C_b + P \times H_b$$

When calculating each color component's value, round to the nearest integer value.

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include:

- A line containing an RGB color value, representing the highlight color. The value is provided in (R,G,B) format, where R, G, and B are each integers between 0 and 255 inclusive.
- A line containing two positive integers, separated by spaces, representing the width, W, and height, H, of the image in pixels, respectively
- H lines, each containing W 0's and 1's separated by spaces, representing the image mask
- H lines, each containing W RGB color values separated by spaces. Each value will be provided in (R,G,B) format, where R, G, and B are each integers between 0 and 255 inclusive.

```
2
(128,128,0)
6 6
0 0 0 0 0 0
0 0 0 0 0 0
0 0 1 1 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
(255,128,230) (77,77,77) (102,102,255) (230,26,255) (102,51,51) (153,230,26)
(128,255,128) (230,77,102) (26,26,179) (230,255,51) (204,77,179) (179,230,179)
(255,77,102) (153,77,51) (102,77,255) (230,51,128) (26,179,204) (0,26,102)
(77,102,179) (102,230,204) (102,179,51) (179,102,51) (77,77,77) (230,26,204)
(102,230,179) (0,77,230) (0,26,230) (204,153,102) (153,153,179) (230,102,153)
(230,128,128) (153,102,179) (77,102,179) (230,153,77) (26,153,204) (153,153,179)
(51,51,255)
5 5
0 0 0 0 0
0 0 1 0 0
0 1 1 1 0
0 0 1 0 0
0 0 0 0 0
(26,153,153) (26,128,26) (179,153,26) (179,102,51) (77,26,128)
(179,128,26) (230,255,153) (102,204,230) (179,128,77) (77,26,26)
(128,204,128) (51,0,77) (102,179,128) (179,204,230) (102,26,51)
(204,26,102) (204,255,255) (77,128,230) (153,179,77) (128,51,230)
(153,51,230) (77,128,102) (204,51,230) (204,204,0) (26,128,179)
```

Sample Output

For each test case, your program must print the RGB color values of each pixel in the resulting image, in the same H-by-W layout as given in the input. Each RGB color value should be in (R,G,B) format, where R, G, and B are each integers between 0 and 255 inclusive.

```
(255,128,230) (77,77,77) (102,102,255) (230,26,255) (102,51,51) (153,230,26)
(128,255,128) (128,128,0) (128,128,0) (128,128,0) (128,128,0) (179,230,179)
(255,77,102) (128,128,0) (115,103,128) (179,90,64) (128,128,0) (0,26,102)
(77,102,179) (128,128,0) (115,154,26) (128,128,0) (128,128,0) (230,26,204)
(102,230,179) (128,128,0) (128,128,0) (128,128,0) (153,153,179) (230,102,153)
(230,128,128) (153,102,179) (77,102,179) (230,153,77) (26,153,204) (153,153,179)
(26,153,153) (51,51,255) (51,51,255) (51,51,255) (77,26,128)
(51,51,255) (51,51,255) (77,128,243) (51,51,255) (51,51,255)
(51,51,255) (51,26,166) (77,115,192) (115,128,243) (51,51,255)
(51,51,255) (51,51,255) (64,90,243) (51,51,255) (51,51,255)
(153,51,230) (51,51,255) (51,51,255) (51,51,255) (26,128,179)
```