Department of Electrical Engineering
National Taiwan University
Algorithms, Spring 2019

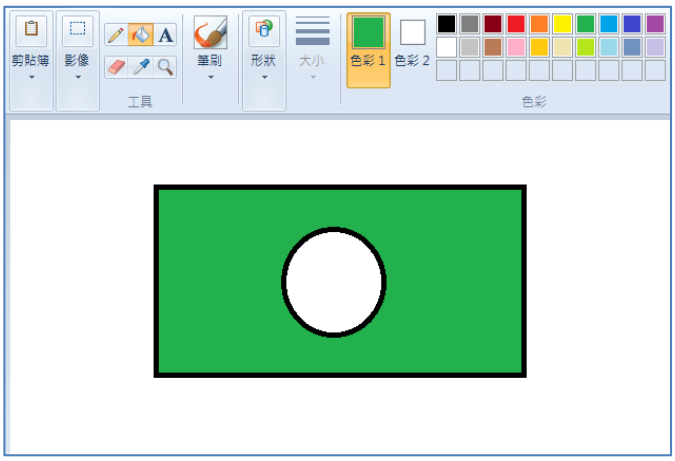
March 31, 2019
Iris Hui-Ru Jiang

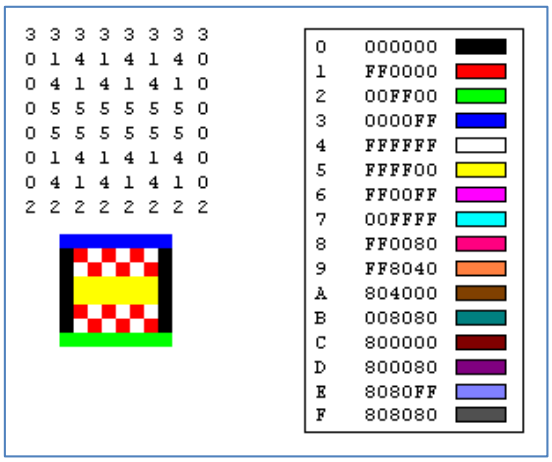
Programming Assignment #1: Paint Bucket

Due 11:59pm, April 21

1. Introduction

Most of the computer aided painting tools provide a function called Paint Bucket. Select a pixel in the image, and Paint Bucket will color the area around the pixel based on color similarity.





Suppose the image is in bitmap file format, which is an array of bits that specify the color of each pixel in the image. For example, if each pixel is represented by 4 bits, then a given pixel can be assigned with 16 (2^4) different colors. With the help of a color table, we can map numbers in the bitmap to specific colors.

If we want to implement the Paint Bucket function on a bitmap image file, we can first model the image as a graph, and then fill the connected component containing the selected pixel (the surrounding pixels with same color). (See slide 22 of Chapter 3 lecture notes.)

2. Problem statement

2.1. Brief description

Given an image in the format similar to a bitmap file, we want to color the area around a selected pixel with a specified color. Write a program to fill all adjacent pixels with the same color and compute how many connected components in this image.

2.2. Input/output specification

The program should be invoked like this:

```
./paintBucket [image*.in] [x] [y] [color] [image*.out]
```

- Input

(1) [image*.in]: The input image file. This file has $h + 1$ lines. The first line of this file contains

two integers w and h, which specify the size of the image (w*h). In the following h lines, each line has w characters (from 0 to F), indicates the color of each pixel from the upper left corner to the bottom right corner on the image.

(2) [x]: An integer to specify the x-coordinate of the selected pixel. x = 0 means the leftmost column. ($0 \le x < w$)

(3) [y]: An integer to specify the y-coordinate of the selected pixel. y = 0 means the first (uppermost) row. ($0 \le y < h$)

(4) [color]: An character to indicate the color to fill. (0~9 and A~F)

- Output

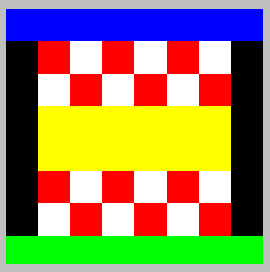
The program should generate an output image file named [image*.out] after the input image is colored. The output image file format is very similar to the input but has $h + 2$ lines. The first $h + 1$ lines are same as the input, which describe the information of the image **after coloring**. The last line should report the number of connected components **after coloring**.

Sample input and output (The image is for reference only)

```
./paintBucket sample.in 1 3 B sample1.out
./paintBucket sample.in 3 1 4 sample2.out
```

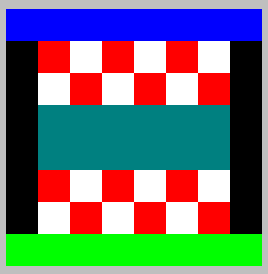
sample.in

```
8 8
3 3 3 3 3 3 3 3
0 1 4 1 4 1 4 0
0 4 1 4 1 4 1 0
0 5 5 5 5 5 5 0
0 5 5 5 5 5 5 0
0 1 4 1 4 1 4 0
0 4 1 4 1 4 1 0
2 2 2 2 2 2 2 2
```



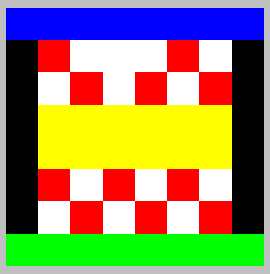
sample1.out

```
8 8
3 3 3 3 3 3 3 3
0 1 4 1 4 1 4 0
0 4 1 4 1 4 1 0
0 B B B B B B 0
0 B B B B B B 0
0 1 4 1 4 1 4 0
0 4 1 4 1 4 1 0
2 2 2 2 2 2 2 2
29
```



sample2.out

```
8 8
3 3 3 3 3 3 3 3
0 1 4 4 4 1 4 0
0 4 1 4 1 4 1 0
0 5 5 5 5 5 5 0
0 5 5 5 5 5 5 0
0 1 4 1 4 1 4 0
0 4 1 4 1 4 1 0
2 2 2 2 2 2 2 2
26
```



3. Evaluation

Your program will be evaluated by the correctness and runtime. The output file of your program will be compared with the correct answer. 60% credits are given for each case if your coloring results (the first $h + 1$ lines) are correct. You will get the other 40% credits if you report the correct number of connected components (the last line). If your implementation takes more than **1 minute** to complete in some cases, these cases will be graded fail even if the results are correct. Please make sure your program can be executed correctly on **EDA Union servers** before submission.

4. References

[1] https://docs.microsoft.com/zh-tw/dotnet/framework/winforms/advanced/types-of-bitmaps

5. Submission

- ATTENTION: Plagiarism MUST be avoided. Every submission will be tested by a plagiarism catcher, running on all submissions from this class (and previous classes). If plagiarism is discovered, all students involved will receive only partial credit. Specifically, if the score received is x and there are n students involved, then the score for each student is x/n.
- Your program **MUST** support the following command-line parameters:

  ./paintBucket [image*.in] [*x*] [*y*] [color] [image*.out]

  If your program cannot support, you can fix your code once and get penalty on your total score.
- Please write a readme file (readme.txt) to describe how to compile and build your program. For example,

  g++ -o paintBucket source.cpp
  or
  make (if you have a Makefile)
- Compress your source code, binary code, and readme file to a zip file named as: StudentID_pa1.zip (e.g., b05901000_pa1.zip)
- Submit the zip file to ceiba