CS 114, Fall 2021, Prof. Calvin

Problem set #9

Due: Wed. Nov. 11, 12:30 pm

The purpose of this assignment is to explore an application of selection. The goal is to "un-clutter" a set of images. The Birds.zip archive on Canvas contains a set of seventeen files, birds1.ppm through birds17.ppm, storing images in the PPM (Portable Pixel Map) format. The background is similar for each image, but each has some clutter (attacking birds) obstructing the background. The archive also contains Blur.java, which has the boilerplate code to open and read in the files, and also write an output file (it just averages the pixel values over the images). You need to write a program (call it Clear.java) that produces an image file with the clutter removed. Your program must be efficient.

The images are sampled from a scene in the Alfred Hitchcock move "The Birds"; you can see the clip at `https://www.youtube.com/watch?v=hplpQt424Ls`.

The PPM files are plain text files with the following format. The first line gives the image type, which is always "P3" for this exercise. The next line has two integers giving the number of columns and rows of the image. The next line has an integer giving the maximum level for any color. Then the rest of the file is a sequence of integers, three for each pixel, giving the red, green, and blue intensities.

For example, consider the following image file:

```
P3
3 4
255
255 0 0 0 255 0 0 0 255
0 0 0 255 255 255 120 120 120
255 255 0 255 0 255 0 255 255
100 100 100 200 200 200 0 100 0
```

The first three lines are the header, with the first line indicating that this is a P3 (plain) image. The next line tells us that this image has 3 columns and 4 rows of pixels. The next line gives the maximum intensity level of 255; all subsequent numbers in the file are between and 0 and the maximum 255. The following lines give the red, green, blue values for each pixel; the first pixel has intensities 255 (red), 0 (green) and 0 (blue), so the top left pixel is red. The next pixel is green and the last pixel on the first row is blue. The first pixel on the second row is black (all three intensities equal to zero).

On the Linux machines in the OSL lab, you can display a ppm image by clicking on it. On a Windows machine, you can display PPM images using, for example, the LibreOffice Draw program or the ppmReader.html file in the Birds directory (open it with the Chrome browser).

Be prepared to demonstrate your program during your next lab. Your program should efficiently produce an image that shows what the view from inside the car would be if there were no birds obstructing the view.

Upload your program to Vocareum. The image files are there, along with starter code to access them.

Exercises:

(1) Consider a version of the problem described above with n images, each having P pixels, with pixels having values between 1 and C. Briefly describe an efficient algorithm to solve this problem when n is large compared to P and C, and characterize the running time of your algorithm in terms of n, P, and C. State whether your analysis is average or worst case.
(2) Describe how to sort n numbers, each between 0 and $n^2 - 1$, in time $O(n)$. (Hint: how could you apply radix sort?)
(3) Suppose that we perform a mixture of n operations on an initially empty priority queue. Argue that in the worst case, these n operations take time $\Omega(n \lg(n))$. Assume that the order of elements in the priority queue is determined by pairwise comparisons, but make no other assumptions about how the priority queue is implemented.
(4) We used a binary decision tree to model a sorting algorithm; each comparison had two possible outcomes. The Java compareTo method gives three outcomes: less, equal, or greater. A sorting algorithm using three outcomes would be modeled with a ternary tree; each internal node has three children. Derive a lower bound for a sorting algorithm that uses such three-way comparisons.