CS 1120: Media Computation Spring 2017
Lab 3: Using conditional statements to change color

Many of you will remember the iconic poster supporting then candidate Barack Obama during the 2008 presidential campaign.

This isn't meant as a political statement – it was simply a strong piece of graphic design that used only four colors. In that regard it is a great example for an activity on the effect of "posterizing" In the reading this week you found out that the term "posterize" means to take an image and convert it from many colors down to a much smaller set of colors. The term isn't very hard to understand. In the old days of printing posters with the screen print process it was important to reproduce images with only a small set of colors since each color was a separate screen and required additional time and labor to print. Let's look at the two program's in your textbook that were posterize functions.

Program 52 (p.124) is a function that reduced a picture from the 16.7 million colors available in our color encoding system down to only 64 colors. It did this by changing the value of red in each pixel from one of 256 possible values to one of only 4 values. Independent of this, it set green, and then blue, to one of only 4 values. If you have 4 choices of red, and 4 choices of green, and 4 choices of blue, then there are 4x4x4 = 64 possible combinations.

Program 53 (p. 125) does this a very different way. Program 41 decided to posterize down to only two colors. To do this it calculated the luminance value of each pixel in the picture. The code then decided that "dark pixels" (ones with low luminance) should go to black while "light pixels" (ones with high luminance) should go to white.

We are going to do something slightly different. We are going to write a filter function that takes a picture and turns it into an Obama-style poster.

But how do we do that?

First, open the Obama image in the explorer (use the explore() finction) and answer the following questions.
http://sergey.cs.uni.edu/courses/cs1120/spring2017/media/obama.jpg

[Q1] What are the RGB values of the light blue color?

[Q2] What is the luminance value of the light blue color?

[Q3] What are the RGB values of the dark blue color?

[Q4] What is the luminance value of the dark blue color?

[Q5] What are the RGB values of the tan color?

[Q6] What is the luminance value of the tan color?

[Q7] What are the RGB values of the red color?

[Q8] What is the luminance value of the red color?

[Q9] Put those four colors in order from “darkest” to “lightest” by using their luminance values.

Now that you have this information, the code for obamafy() isn’t too difficult to write. Notice that when you look at the code from Program 52 and Program 53, that 53 is much closer to what we want to do. I would encourage you to use Program 53 as a guide and write a function called obamafy() that:

- Takes in a picture
- Iterates over each pixel in the picture
- For each pixel, calculates its luminance value
- Divides the 256 possible luminance values – from 0 to 255 – into four equal regions.
- Pixels with the darkest luminance values should be set to the darkest of the four Obama poster colors. Pixels with the lightest luminance values should be set to the lightest of the four Obama poster colors. Of course the two middle colors should also be there.

When you write your function, make sure the body of your loop contains only the essential code - i.e., statements that must be executed at each iteration of the loop.

When this function is done it should show/return the image so that we get something like:

Keep in mind that different photos may require different "regions". A photo that is mostly dark or mostly light may result in a less optimal/balanced "obamafication."

[Q10] Write down your function definition on the response sheet..