Authors: Alexander Marcoux and Rahul Gupta

The region growing algorithm (stored in RegionFinder) begins by looping over all pixels in the frame. We simultaneously use a buffered image to store information about the pixels we visit and those that remain unvisited and store the regions of visited and unvisited (toVisit) pixels in array lists. If a pixel has not been visited and is of the correct color (which is checked using colorMatch, comparing the absolute value of each channel), we start a new region array list and add the corresponding pixel to said array list. Because we are keeping track of pixels visited and unvisited, we have access to those pixels which need to be visited. While the size of the toVisit array list is not zero, we remove a pixel and add it to the visited region, then calculate the neighboring pixels and visit them, adding them to the visited list as well. To mark visited pixels we are coloring the subsequent image (which is originally black) with white pixels to signify which pixels have been visited. If the region is big enough (the value of the threshold is established in the instance variables) we will add it to our array list of array lists (regions).

For the extra credit, we changed the handleKeyPress method, using a switch-case syntax instead of the if-else format. We kept the same keyPress options as the original document but added an option for the user to change the color of the image using the spacebar. When the spacebar is pressed, the user is prompted to type the rgb values they wish to see in the image. The input is read using BufferedReader and the rgb values are parsed into integers to pass into the paintColor object which controls the color of the image. To catch exceptions (errors from the user input) we prompt the reader to try again and reprompt them to enter the rgb values they wish to see. Every time the spacebar is pressed, the image tracking process is paused until the user correctly responds.