import java.io.File;

import java.io.IOException;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Color;

public class FindColor

{

public static int[] whatColor(File f) throws IOException

{

// Create return array, read image

int[] rgb = new int[3];

Bitmap img = BitmapFactory.decodeFile(f.toString());

// Instantiate

Color color;

int x, y, rgbVal;

// Obtain location of center pixel

x = img.getWidth() / 2;

y = img.getHeight() /2;

// Get RGB value from pixel

rgb = img.getPixel(x, y);

// Divide by color and return

rgb[0] = (rgbVal >>> 16);

rgb[1] = (rgbVal >> 8);

rgb[2] = (rgbVal);

return rgb;

}

public static int[] findColorsRec(File f) throws IOException

{

// Create return array, read image

int[] rgb = new int[3];

Bitmap img = BitmapFactory.decodeFile(f.toString());

// Instantiate

Color color;

int rgbVal = 0, x, y, count = 1, r, g, b, rTemp, gTemp, bTemp;

int rAverage = 0, gAverage = 0, bAverage = 0;

// Obtain location of center pixel

x = img.getWidth() / 2;

y = img.getHeight() /2;

// Get RGB value from center pixel

rgbVal = img.getPixel(x, y);

r = (rgbVal >> 16) & 0xff;

g = (rgbVal >> 8) & 0xff;

b = (rgbVal) & 0xff;

rAverage += r;

gAverage += g;

bAverage += b;

//Check upper pixels

for(int h = y; h > 0; h--)

{

rgbVal = img.getRGB(x, h);

rTemp = (rgbVal >> 16) & 0xff;

gTemp = (rgbVal >> 8) & 0xff;

bTemp = (rgbVal) & 0xff;

if(rTemp > r-30 && rTemp < r+30 && gTemp > g-30 && gTemp < g+30 && bTemp > b-30 && bTemp < b+30)

{

count++;

rAverage += rTemp;

gAverage += gTemp;

bAverage += bTemp;

}

}

//Check lower pixels

for(int h = y; h < img.getHeight(); h++)

{

rgbVal = img.getPixel(x, h);

rTemp = (rgbVal >> 16) & 0xff;

gTemp = (rgbVal >> 8) & 0xff;

bTemp = (rgbVal) & 0xff;

if(rTemp > r-30 && rTemp < r+30 && gTemp > g-30 && gTemp < g+30 && bTemp > b-30 && bTemp < b+30)

{

count++;

rAverage += rTemp;

gAverage += gTemp;

bAverage += bTemp;

}

}

//Check left pixels

for(int w = x; w < 0; w--)

{

rgbVal = img.getPixel(w, y);

rTemp = (rgbVal >> 16) & 0xff;

gTemp = (rgbVal >> 8) & 0xff;

bTemp = (rgbVal) & 0xff;

if(rTemp > r-30 && rTemp < r+30 && gTemp > g-30 && gTemp < g+30 && bTemp > b-30 && bTemp < b+30)

{

count++;

rAverage += rTemp;

gAverage += gTemp;

bAverage += bTemp;

}

}

//Check right pixels

for(int w = x; w < img.getWidth(); w++)

{

rgbVal = img.getPixel(w, y);

rTemp = (rgbVal >> 16) & 0xff;

gTemp = (rgbVal >> 8) & 0xff;

bTemp = (rgbVal) & 0xff;

if(rTemp > r-30 && rTemp < r+30 && gTemp > g-30 && gTemp < g+30 && bTemp > b-30 && bTemp < b+30)

{

count++;

rAverage += rTemp;

gAverage += gTemp;

bAverage += bTemp;

}

}

rAverage = rAverage / count;

gAverage = gAverage / count;

bAverage = bAverage / count;

rgb[0] = rAverage;

rgb[1] = gAverage;

rgb[2] = bAverage;

return rgb;

}

public static void main(String[] args) throws IOException

{

File f = new File("C:/Users/Steve/Downloads/chicken.jpg");

int[] colors = findColorsRec(f);

System.out.println("--------------Recursive--------------");

System.out.println("Red: " + colors[0]);

System.out.println("Blue: " + colors[1]);

System.out.println("Green: " + colors[2]);

colors = whatColor(f);

System.out.println("--------------Center--------------");

System.out.println("Red: " + colors[0]);

System.out.println("Blue: " + colors[1]);

System.out.println("Green: " + colors[2]);

}

}