```java
package edu.asu.msrs.artcelerationlibrary;

import android.util.Log;

/**
 * Created by tangmiao on 11/27/2016.
 * This transform was basically trans-form the color value of every pixel into another
 * color value for each channel of RGB image.   The  transform  was  based  on  the  specified
 * piece-wise  function.   Since  there were three channels, there were also three piecewise functions.
 * Every piecewise function was given by eight numbers, which represented 4 points(x value and value)
 * onthe  linear  piece  wise  function  plot.   Since  there  were  three  channels,  we  would be given
 * an array including 24 numbers in total.  These number would determine how the original figure will be transformed.
 */
public class ColorFilter {
    public static byte[] piecewiseprocess(byte[] pixels){
        String TAG = "ColorFilter";
        Log.d(TAG,"Start");

        int [] piecewiseArray = new int[]{26, 26, 30, 80, 100, 150, 170,230,
                1, 68, 30, 10, 150, 150, 200,
                30,100, 130, 130, 80, 200, 250, 240, 5};


        for (int i = 0; i < pixels.length/4; i++) {
            pixels[4*i+1] = ArrayOperater(pixels[4*i+1],0, piecewiseArray);
            pixels[4*i+2] = ArrayOperater(pixels[4*i+2],8, piecewiseArray);
            pixels[4*i+3] = ArrayOperater(pixels[4*i+3],16, piecewiseArray);
        }
        Log.d(TAG,"End");
        return pixels;


    }



    //Input: Original image pixels, different channel indexes, and piecewiseArray
    //Output: all the  image pixels after processed
    static public byte ArrayOperater(byte pixel1,int colorshift, int[] piecewiseArray) {

        int pixel = pixel1 & 0xFF;

        if (pixel< 0) {
            pixel = 0;
        }else if (pixel >= 0 || pixel < piecewiseArray[0+colorshift]) {
            pixel = (pixel)*(piecewiseArray[1+colorshift])/(piecewiseArray[0+colorshift]);
        }else if (pixel >= piecewiseArray[0+colorshift] || pixel < piecewiseArray[2+colorshift]) {
            pixel= piecewiseArray[0+colorshift]+(pixel-piecewiseArray[0+colorshift])*((piecewiseArray[3+colorshift]-piecewiseArray[1+colorshift])/(piecewiseArray[2+colorshift]-piecewiseArray[0+colorshift]));
        }else if (pixel >= piecewiseArray[2+colorshift] || pixel < piecewiseArray[4+colorshift]) {
            pixel = (piecewiseArray[2+colorshift]+(pixel-piecewiseArray[2+colorshift])*((piecewiseArray[5+colorshift]-piecewiseArray[3+colorshift])/(piecewiseArray[4+colorshift]-piecewiseArray[2+colorshift])));
        }else if (pixel >= piecewiseArray[4+colorshift]|| pixel < piecewiseArray[6+colorshift]) {
            pixel = (piecewiseArray[4+colorshift]+(pixel-piecewiseArray[4+colorshift])*((piecewiseArray[7+colorshift]-piecewiseArray[5+colorshift])/(piecewiseArray[6+colorshift]-piecewiseArray[4+colorshift])));
        }else if (pixel >= piecewiseArray[6+colorshift]|| pixel < 255){
            pixel = (piecewiseArray[6+colorshift]+ (pixel - piecewiseArray[6+colorshift])* (255 - piecewiseArray[7+colorshift])/(255 - piecewiseArray[6+colorshift]));
        } else {
            pixel = 255;
        }


        return  (byte)pixel;
    }


}

```