```java
package Assign_5_A;

import Media.*;                  // for Pictures and Sounds
import java.awt.*;               // for Color objects and methods
import static java.lang.Math.*;  // for math constants and functions
import static java.awt.Color.*;  // for Color constants

/** This class creates a "Pencil Sketch" version of a photo using Edge Detection, by comparing the intensity of a pixel
  * with the pixel immediately below it. If the absolute difference in the intensities is smaller than a value
  * TOLERANCE (a constant with value 10.0), the pixel is set to white, otherwise it is set to black.
  *
  * @author Sawyer Fenwick st#6005011
  * @version 1.0 November 17 2016                                                */

public class Edge_Detection {

    // instance variables
    private PictureDisplayer display;
    final double TOLERANCE = 10.0;

    /** This constructor displays a picture on the display and runs the method "edgeDetection"
      * which turns it into a black and white "Pencil Sketch".*/
    public Edge_Detection ( ) {

        Picture pic;
        pic = new Picture();
        display = new PictureDisplayer(pic);
        display.waitForUser();
        edgeDetection(pic);
        display.close();
        // statements including call to method

    }; // constructor

      public void edgeDetection(Picture aPic){

        double topIntensity;
        double lowIntensity;
        double prevLowInt;
        double prevTopInt;

        int height = 480;
        int width = 640;
        int x = 0;
        int y = 0;

        //for loop for all pixels except the final row
        for(int r = 0; r < height - 1; r++){
          for(int c = 0; c < width; c++){

            //topPixel
            Pixel t = aPic.getPixel(x + c , y + r);
            Color topColor = t.getColor();
            topIntensity = intensity(topColor);
            prevTopInt = topIntensity;

            //lowPixel
            Pixel l = aPic.getPixel(x + c, y + r +1);
            Color lowColor = l.getColor();
            lowIntensity = intensity(lowColor);
            prevLowInt = lowIntensity;

            //absoulteValue
            double result = Math.abs(topIntensity - lowIntensity);

            if((result < TOLERANCE)){
              t.setColor(WHITE);
            }else{
              t.setColor(BLACK);
            };

          }

        }

        //for loop for the final row
        for(int r = 0; r == 0; r ++){
          for(int c = 0; c < width; c++){

            Pixel p = aPic.getPixel(x + c, 479);
            Color col = p.getColor();
            topIntensity = intensity(col);
            double result = topIntensity - topIntensity;

            if(result < TOLERANCE){
             p.setColor(WHITE);
            }else{
              p.setColor(BLACK);
            }

          }
        }
      }

      /*This method retrieves the R G B values of the passed pixel, and determines its Intensity, which it sends back
       * to the edgeDetection. */
      private double intensity(Color c){

          double B = c.getBlue();
          double R = c.getRed();
          double G = c.getGreen();

          double intensity = (B + R + G)/3.0;

        return intensity;

      }// intensity

    public static void main ( String[] args ) { Edge_Detection s = new Edge_Detection(); };

    }// Edge_Detection
```