```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package pointsinplane;

import java.awt.BasicStroke;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.RenderingHints;
import java.awt.geom.Ellipse2D;
import java.awt.geom.Line2D;
import javax.swing.JPanel;


class Drawing extends JPanel implements Constants {
    // x-axis coord constants

    public static final int X_AXIS_FIRST_X_COORD = XO;
    public static final int X_AXIS_SECOND_X_COORD = XO + LENGTH_X;
    public static final int X_AXIS_Y_COORD = FR_HEIGHT - YO;

    // y-axis coord constants
    public static final int Y_AXIS_FIRST_Y_COORD = FR_HEIGHT - YO -
LENGTH_Y;
    public static final int Y_AXIS_SECOND_Y_COORD = FR_HEIGHT - YO;
    public static final int Y_AXIS_X_COORD = XO;

    //arrows of axis are represented with "hipotenuse" of
    //triangle
    // now we are define length of cathetas of that triangle
    public static final int FIRST_LENGHT = 10;
    public static final int SECOND_LENGHT = 5;

    // size of start coordinate lenght
    public static final int ORIGIN_COORDINATE_LENGHT = 10;

    // distance of coordinate strings from axis
    public static final int AXIS_STRING_DISTANCE = 20;

    Ellipse2D AG;
    Ellipse2D BG;
    Line2D AX_Line;
    Line2D AY_Line;
    Line2D BX_Line;
    Line2D BY_Line;

    public Drawing(Point A, Point B) {
        int AgeomX = A.x * RATIO + XO;
        int AgeomY = FR_HEIGHT - A.y * RATIO - YO;
        int BgeomX = B.x * RATIO + XO;
        int BgeomY = FR_HEIGHT - B.y * RATIO - YO;
```

```
        AG = new Ellipse2D.Double(AgeomX - R, AgeomY - R, 2 * R, 2 * R);
        BG = new Ellipse2D.Double(BgeomX - R, BgeomY - R, 2 * R, 2 * R);

        AX_Line = new Line2D.Double(Y_AXIS_X_COORD, AG.getCenterY(),
AG.getCenterX(), AG.getCenterY());
        AY_Line = new Line2D.Double(AG.getCenterX(), AG.getCenterY(),
AG.getCenterX(), X_AXIS_Y_COORD);
        BX_Line = new Line2D.Double(Y_AXIS_X_COORD, BG.getCenterY(),
BG.getCenterX(), BG.getCenterY());
        BY_Line = new Line2D.Double(BG.getCenterX(), BG.getCenterY(),
BG.getCenterX(), X_AXIS_Y_COORD);
    }

    public void movePoint(Point A, Point B) {
        int AgeomX = A.x * RATIO + XO;
        int AgeomY = FR_HEIGHT - A.y * RATIO - YO;
        int BgeomX = B.x * RATIO + XO;
        int BgeomY = FR_HEIGHT - B.y * RATIO - YO;
        AG.setFrame(AgeomX - R, AgeomY - R, 2 * R, 2 * R);
        BG.setFrame(BgeomX - R, BgeomY - R, 2 * R, 2 * R);

        AX_Line = new Line2D.Double(Y_AXIS_X_COORD, AG.getCenterY(),
AG.getCenterX(), AG.getCenterY());
        AY_Line = new Line2D.Double(AG.getCenterX(), AG.getCenterY(),
AG.getCenterX(), X_AXIS_Y_COORD);
        BX_Line = new Line2D.Double(Y_AXIS_X_COORD, BG.getCenterY(),
BG.getCenterX(), BG.getCenterY());
        BY_Line = new Line2D.Double(BG.getCenterX(), BG.getCenterY(),
BG.getCenterX(), X_AXIS_Y_COORD);
    }

    @Override
    public void paintComponent(Graphics g) {

        super.paintComponent(g);

        Graphics2D g2 = (Graphics2D) g;

        g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);

        // x-axis
        g2.drawLine(X_AXIS_FIRST_X_COORD, X_AXIS_Y_COORD,
                X_AXIS_SECOND_X_COORD, X_AXIS_Y_COORD);
        // x-axis negativno
        g2.drawLine(X_AXIS_FIRST_X_COORD, X_AXIS_Y_COORD,
                -X_AXIS_SECOND_X_COORD, X_AXIS_Y_COORD);
        // y-axis
        g2.drawLine(Y_AXIS_X_COORD, Y_AXIS_FIRST_Y_COORD,
                Y_AXIS_X_COORD, Y_AXIS_SECOND_Y_COORD);
        // y-axis
        g2.drawLine(Y_AXIS_X_COORD, Y_AXIS_FIRST_Y_COORD+  LENGTH_Y,
                Y_AXIS_X_COORD, Y_AXIS_SECOND_Y_COORD+ LENGTH_Y);
```

```
        // x-axis arrow
        g2.drawLine(X_AXIS_SECOND_X_COORD - FIRST_LENGHT,
                X_AXIS_Y_COORD - SECOND_LENGHT,
                X_AXIS_SECOND_X_COORD, X_AXIS_Y_COORD);
        g2.drawLine(X_AXIS_SECOND_X_COORD - FIRST_LENGHT,
                X_AXIS_Y_COORD + SECOND_LENGHT,
                X_AXIS_SECOND_X_COORD, X_AXIS_Y_COORD);

        // y-axis arrow
        g2.drawLine(Y_AXIS_X_COORD - SECOND_LENGHT,
                Y_AXIS_FIRST_Y_COORD + FIRST_LENGHT,
                Y_AXIS_X_COORD, Y_AXIS_FIRST_Y_COORD);
        g2.drawLine(Y_AXIS_X_COORD + SECOND_LENGHT,
                Y_AXIS_FIRST_Y_COORD + FIRST_LENGHT,
                Y_AXIS_X_COORD, Y_AXIS_FIRST_Y_COORD);

        // draw origin Point
        g2.fillOval(
                Y_AXIS_X_COORD - (ORIGIN_COORDINATE_LENGHT / 2),
                X_AXIS_Y_COORD - (ORIGIN_COORDINATE_LENGHT / 2),
                ORIGIN_COORDINATE_LENGHT, ORIGIN_COORDINATE_LENGHT);

        // draw text "X" and draw text "Y"
        g2.drawString("X", X_AXIS_SECOND_X_COORD - AXIS_STRING_DISTANCE /
2,
                X_AXIS_Y_COORD + AXIS_STRING_DISTANCE);
        g2.drawString("Y", Y_AXIS_X_COORD - AXIS_STRING_DISTANCE,
                Y_AXIS_FIRST_Y_COORD + AXIS_STRING_DISTANCE / 2);
        g2.drawString("(0, 0)", X_AXIS_FIRST_X_COORD -
AXIS_STRING_DISTANCE,
                Y_AXIS_SECOND_Y_COORD + AXIS_STRING_DISTANCE);

        // numerate axis
        int xCoordNumbers = 10;
        int yCoordNumbers = 10;
        int xLength = (X_AXIS_SECOND_X_COORD - X_AXIS_FIRST_X_COORD)
                / xCoordNumbers;
        int yLength = (Y_AXIS_SECOND_Y_COORD - Y_AXIS_FIRST_Y_COORD)
                / yCoordNumbers;

        // draw x-axis numbers
        for (int i = 1; i < xCoordNumbers; i++) {
            g2.drawLine(X_AXIS_FIRST_X_COORD + (i * xLength),
                    X_AXIS_Y_COORD - SECOND_LENGHT,
                    X_AXIS_FIRST_X_COORD + (i * xLength),
                    X_AXIS_Y_COORD + SECOND_LENGHT);
            g2.drawString(Integer.toString(i),
                    X_AXIS_FIRST_X_COORD + (i * xLength) - 3,
                    X_AXIS_Y_COORD + AXIS_STRING_DISTANCE);
        }
        // draw x-axis negative numbers
        for (int i = 1; i < xCoordNumbers; i++) {
            g2.drawLine(X_AXIS_FIRST_X_COORD - (i * xLength),
```

```
                X_AXIS_Y_COORD - SECOND_LENGHT,
                X_AXIS_FIRST_X_COORD - (i * xLength),
                X_AXIS_Y_COORD + SECOND_LENGHT);
            g2.drawString(Integer.toString(-i),
                X_AXIS_FIRST_X_COORD - (i * xLength) - 3,
                X_AXIS_Y_COORD + AXIS_STRING_DISTANCE);
        }

        //draw y-axis numbers
        for (int i = 1; i < yCoordNumbers; i++) {
            g2.drawLine(Y_AXIS_X_COORD - SECOND_LENGHT,
                Y_AXIS_SECOND_Y_COORD - (i * yLength),
                Y_AXIS_X_COORD + SECOND_LENGHT,
                Y_AXIS_SECOND_Y_COORD - (i * yLength));
            g2.drawString(Integer.toString(i),
                Y_AXIS_X_COORD - AXIS_STRING_DISTANCE,
                Y_AXIS_SECOND_Y_COORD - (i * yLength));
        }

        //draw y-axis negative numbers
        for (int i = 1; i < yCoordNumbers; i++) {
            g2.drawLine(Y_AXIS_X_COORD - SECOND_LENGHT,
                Y_AXIS_SECOND_Y_COORD + (i * yLength),
                Y_AXIS_X_COORD + SECOND_LENGHT,
                Y_AXIS_SECOND_Y_COORD + (i * yLength));
            g2.drawString(Integer.toString(-i),
                Y_AXIS_X_COORD - AXIS_STRING_DISTANCE,
                Y_AXIS_SECOND_Y_COORD + (i * yLength));
        }

        //Crta tacke
        g2.setColor(A_COLOR);
        g2.fill(AG);
        g2.setColor(B_COLOR);
        g2.fill(BG);

        //Isprekidana linija
        float[] dash1 = {2f, 0f, 2f};
        BasicStroke bs1 = new BasicStroke(1,
                BasicStroke.CAP_BUTT,
                BasicStroke.JOIN_ROUND,
                1.0f,
                dash1,
                2f);
        g2.setStroke(bs1);
        g2.setColor(A_COLOR);
        g2.draw(AX_Line);
        g2.draw(AY_Line);
         g2.drawString("A", (int)AG.getCenterX()-10, (int)
AG.getCenterY()-10);
         g2.setColor(B_COLOR);
        g2.draw(BX_Line);
        g2.draw(BY_Line);
```

```
        g2.drawString("B", (int)BG.getCenterX()-10, (int)
BG.getCenterY()-10);

    }
}
```