МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 2

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

Виконав
студент III курсу
групи КП-83
Мричко Богдан
(*прізвище, ім’я, по батькові*)

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною
(*прізвище, ім’я, по батькові*)

варіант № 15

Київ 2020

Варіант завдання

Завдання: Побудова та анімація зображень за допомогою Java2D

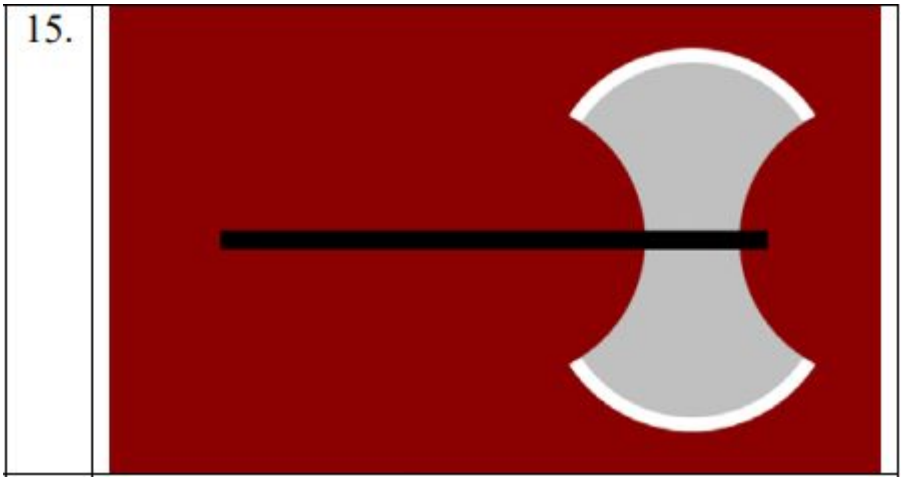
Короткі теоретичні відомості: Є два види комп'ютерної графіки – растрова та векторна. Растрова графіка представляє зображення як набір пікселів. Векторна графіка використовує геометричні примітиви – точки, лінії, полігони, дуги. Java 2D надає можливість працювати як з растровою, так і з векторною графікою.

Java 2D це API для створення двовимірних зображень за допомогою мови програмування Java. Java 2D API має наступні властивості:

- широкий вибір геометричних примітивів
- визначення перетину фігур, тексту, зображень
- контроль якості рендерингу
- друк документів
- уніфікована модель рендерингу для дисплеїв та принтерів

Java 2D досить потужна технологія, за допомогою якої можна створювати потужні та красиві інтерфейси користувача, ігри, анімації, мультимедійні програми.

Завдання за варіантом:



Код програми

Main.java

```
package sample;


import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.geom.GeneralPath;

public class Main extends JPanel implements ActionListener {

    Timer timer;

    private static int maxWidth = 1024;
    private static int maxHeight = 768;


    public int positionX = 300;
    public int positionY = 200;

    public static int corners[][] = {
            { 300, 150 },
            { 800, 500 }
    };

    private double angle = 0;

    private int deltaX = 1;
    private int deltaY = 1;


    private final double center_x = 1;
    private final double center_y = 1;

    public Main() {
        timer = new Timer(10, this);
        timer.start();
    }

    public void paint(Graphics g) {
        Graphics2D g2d = (Graphics2D) g;
        RenderingHints rh = new
RenderingHints(RenderingHints.KEY_ANTIALIASING,
                RenderingHints.VALUE_ANTIALIAS_ON);
        rh.put(RenderingHints.KEY_RENDERING,
RenderingHints.VALUE_RENDER_QUALITY);
        g2d.setRenderingHints(rh);

        g2d.setBackground(new Color(138, 0, 0));
        g2d.clearRect(0, 0, maxWidth, maxHeight);


        g2d.setColor(new Color(255, 253, 56));
        BasicStroke bs1 = new BasicStroke(16, BasicStroke.CAP_ROUND,
```

```
                BasicStroke.JOIN_ROUND);
        g2d.setStroke(bs1);
        g2d.drawRect(20, 20, maxWidth-40, maxHeight-40);


        g2d.rotate(angle, positionX-20, positionY+55);
        GradientPaint gp = new GradientPaint(5, 25, Color.RED, 20, 2,
Color.WHITE, true);
        g2d.setPaint(gp);
        g2d.fillOval(positionX, positionY, 70, 110);
        g2d.setColor(new Color(142, 142, 142));
        g2d.fillOval(positionX+5, positionY+5, 60, 100);
        g2d.setColor(new Color(138, 0, 0));
        g2d.fillOval(positionX-45, positionY+15, 70, 80);
        g2d.fillOval(positionX+45, positionY+15, 70, 80);

        int corrector  = 55;
        GradientPaint gp1 = new GradientPaint(5, 25, Color.BLACK, 40, 2, new
Color(80, 55, 0), true);
        g2d.setPaint(gp1);
        double points[][] = {
                { positionX-90, positionY-6+corrector },
                { positionX-90, positionY+6 +corrector},
                { positionX+65, positionY+6 +corrector},
                { positionX+65, positionY-6 +corrector}
        };
        GeneralPath rect = new GeneralPath();
        rect.moveTo(points[0][0], points[0][1]);
        for (int k = 1; k < points.length; k++)
            rect.lineTo(points[k][0], points[k][1]);
        rect.closePath();
        g2d.fill(rect);

    }

    public void actionPerformed(ActionEvent e) {

        int velocity = 5;

        if (positionX <= corners[0][0] && positionY<= corners[1][1]) {
            deltaX = 0;
            deltaY = velocity;
        } else if (positionX <= corners[1][0] && positionY>= corners[1][1]) {
            deltaX = velocity;
            deltaY = 0;
        } else if (positionX >= corners[1][0] && positionY>= corners[0][1]) {
            deltaX = 0;
            deltaY = -velocity;
        }else if (positionX >= corners[0][0] && positionY<= corners[0][1]) {
            deltaX = -velocity;
            deltaY = 0;
        }
        positionX+=deltaX;
        positionY+=deltaY;
        angle -= 0.1;
        repaint();
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame("lab2");
        frame.add(new Main());
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(maxWidth, maxHeight);
```

```
        frame.setResizable(false);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
        Dimension size = frame.getSize();
        Insets insets = frame.getInsets();
        maxWidth = size.width - insets.left - insets.right - 1;
        maxHeight = size.height - insets.top - insets.bottom - 1;
    }

}
```

Результат