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”

Виконала
студентка III курсу
групи КП-82
Морщак Каріна Юріївна
(*прізвище, ім’я, по батькові*)

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

варіант № 14

Київ 2021

Варіант завдання

Тема: Побудова та анімація зображень за допомогою Java2D

Завдання: За допомогою Java 2D намалювати картинку з лабораторної роботи №1 (за варіантом). Додатково виконати: 1. Хоча б 1 стандартний примітив, та хоча б 1 фігуру, побудовану по точкам (ламаною). 2. Хоча б 1 фігуру залити градієнтною фарбою за вибором (в цьому випадку колір може не співпадати з варіантом із лабораторної роботи № 1).11 3. На достатній відстані від побудованого малюнку намалювати прямокутну рамку, всередині якої відбуватиметься анімація. Тип лінії рамки задано за варіантом. 4. Виконати анімацію малюнку, за варіантом. При цьому рамка повинна залишатися статичною. Взаємодія з рамкою не обов'язкова, якщо не передбачено варіантом.

Завдання за варіантом:

Тип лінії рамки: JOIN_MITER

Типи анімації:

1. Рух по колу проти годинникової стрілки
2. Обертання навколо центру малюнка проти годинникової стрілки

Лістинг коду програми

Animation2D.java

```
package main;

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



public class Animation2D extends JPanel implements ActionListener {
    Timer timer;

    private double angle = 0;
    private double scale = 1;
    private double delta = 0.01;
    private double tx = 1;
    private double ty = 1;
    private static int maxHeight;
    private static int maxWidth;
    private int radius = 100;

    public void paint(Graphics graph) {
      Graphics2D graph2d = (Graphics2D) graph;

        RenderingHints rh = new
RenderingHints(RenderingHints.KEY_ANTIALIASING,
                  RenderingHints.VALUE_ANTIALIAS_ON);
        rh.put(RenderingHints.KEY_RENDERING,
                  RenderingHints.VALUE_RENDER_QUALITY);
        graph2d.setRenderingHints(rh);

        graph2d.setBackground(new Color(100,150,238));
        graph2d.clearRect(0, 0, maxWidth, maxHeight);
        graph2d.translate(maxWidth/2, maxHeight/2);

        BasicStroke bs1 = new BasicStroke(8, BasicStroke.CAP_ROUND,
BasicStroke.JOIN_MITER);
        graph2d.setStroke(bs1);
```

```
        graph2d.drawRect(-(radius + 150),-(radius + 150),(radius +
150)*2,(radius + 150)*2);

        graph2d.translate(tx, ty);

        graph2d.setColor(Color.WHITE);
        int[] xRect = new int[] {150, 350, 350, 150};
          int[] yRect = new int[] {100, 100, 150, 150};
          Polygon rect = new Polygon(xRect,yRect,4);
          graph2d.rotate(angle, rect.getBounds2D().getCenterX(),
rect.getBounds2D().getCenterY());
          graph2d.drawPolygon(rect);
          graph2d.fillPolygon(rect);

          GradientPaint gp = new GradientPaint(
          30, 60,
          new Color(255, 180, 13),
          50, 20,
          new Color(255, 0, 0),
          true
          );
          graph2d.setPaint(gp);
        int[] x = new int[] {350, 380, 350};
        int[] y = new int[] {100, 125, 150};
        Polygon p = new Polygon(x, y, 3);
        graph2d.drawPolygon(p);
        graph2d.fillPolygon(p);

        graph2d.setColor(new Color(255,128,64));
        double points1[][] = {
              {150, 70}, {180, 70}, {195, 100}, {165, 100}
            };
          GeneralPath sideBlock1 = new GeneralPath();
          sideBlock1.moveTo(points1[0][0],points1[0][1]);
          for (int k = 1; k < points1.length; k++)
            sideBlock1.lineTo(points1[k][0], points1[k][1]);
          sideBlock1.closePath();
          graph2d.fill(sideBlock1);

        double points2[][] = {
                {165, 150}, {195, 150}, {180, 180}, {150, 180}
            };
```

```
          GeneralPath sideBlock2 = new GeneralPath();
          sideBlock2.moveTo(points2[0][0],points2[0][1]);
          for (int k = 1; k < points2.length; k++)
            sideBlock2.lineTo(points2[k][0], points2[k][1]);
          sideBlock2.closePath();
          graph2d.fill(sideBlock2);


        graph2d.fillRect(150, 120, 70, 8);
    }

    public Animation2D() {
        timer = new Timer(10, this);
        timer.start();
    }

    public static void main(String[] args){
        JFrame frame = new JFrame("lab2");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1000, 1000);
        frame.setLocationRelativeTo(null);
        frame.setResizable(false);
        frame.add(new Animation2D());
        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;
    }

    public void actionPerformed(ActionEvent e) {
        if (tx <= 0 && ty < 0){
            tx -= 1;
            ty = (-1) * Math.abs(Math.sqrt(Math.pow(radius, 2) -
Math.pow(tx, 2)));
        }else if(tx > 0 && ty <= 0){
            tx -= 1;
            ty = (-1) * Math.abs(Math.sqrt(Math.pow(radius, 2) -
Math.pow(tx, 2)));
        }else if(tx >= 0 && ty > 0){
            tx += 1;
            ty = Math.abs(Math.sqrt(Math.pow(radius, 2) - Math.pow(tx, 2)));
```

```
        }else if(tx < 0 && ty >= 0){
            tx += 1;
            ty = Math.abs(Math.sqrt(Math.pow(radius, 2) - Math.pow(tx, 2)));
        }

        angle -= 0.01;

        repaint();
    }
}
```

Результат