

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  Линь Дмитро Іванович  варіант № 10 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною |

Київ 2021

**Варіант завдання**

**Завдання**: За допомогою Java 2D намалювати картинку з лабораторної роботи №1.

Додатково виконати:

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).

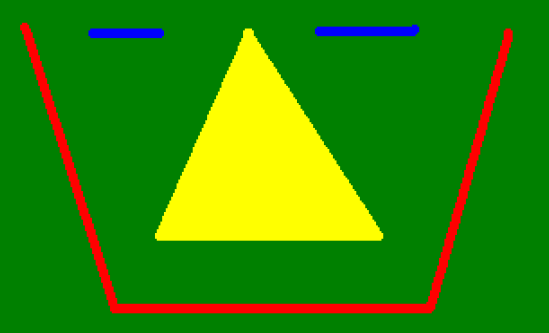
3. На достатній відстані від побудованого малюнку намалювати прямокутну рамку, всередині якої відбуватиметься анімація. Тип лінії рамки задано за варіантом.

4. Виконати анімацію малюнку, за варіантом. При цьому рамка повинна залишатися статичною. Взаємодія з рамкою не обов’язкова, якщо не передбачено варіантом.

**Варіант 10:**

**Тип анімації:** 7, 10

**Тип лінії рамки:** JOIN\_BEVEL



**Лістинг коду програми**

package sample;  
  
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;  
  
@SuppressWarnings("serial")  
public class Main extends JPanel implements ActionListener {  
 Timer timer;  
  
 private static int *maxWidth* = 1600;  
 private static int *maxHeight* = 900;  
  
 private static int *paddingX* = 500;  
 private static int *paddingY* = 250;  
  
 private double angle = 0;  
  
 private double scale = 0.1;  
 private double delta = 0.01;  
  
 private final double centerX = 1;  
 private final double centerY = 1;  
  
 public Main() {  
 timer = new Timer(50, 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(15, 127, 18));  
 g2d.clearRect(0, 0, *maxWidth*, *maxHeight*);  
  
 g2d.setColor(new Color(255, 253, 56));  
 BasicStroke bs1 = new BasicStroke(10, BasicStroke.*CAP\_ROUND*,  
 BasicStroke.*JOIN\_BEVEL*);  
 g2d.setStroke(bs1);  
 g2d.drawRect(0, 0, 1400, 750);  
  
 g2d.translate(*maxWidth* / 2, *maxHeight* / 2);  
 g2d.rotate(angle, centerX, centerY);  
 g2d.scale(scale, scale);  
  
 GradientPaint gp = new GradientPaint(5, 25, Color.*RED*, 20, 2, Color.*GREEN*, true);  
 g2d.setPaint(gp);  
 double points[][] = {  
 { 180, 20 },  
 { 113, 173 },  
 { 280, 173 }  
 };  
 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);  
  
 g2d.setColor(Color.*BLUE*);  
  
 g2d.drawLine(64, 20, 120, 20);  
 g2d.drawLine(233, 20, 302, 20);  
  
 g2d.setColor(Color.*RED*);  
  
 g2d.drawPolyline(new int[]{20, 84, 313, 370}, new int[]{19, 222, 222, 19}, 4);  
 }  
  
 public void actionPerformed(ActionEvent e) {  
  
 if (scale < 0.01) {  
 delta = -delta;  
 } else if (scale > 0.99) {  
 delta = -delta;  
 }  
 angle -= 0.1;  
 scale += delta;  
 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.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 frame.setResizable(false);  
 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 у директорії разом з цим документом**