

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-02  Щербатюк Ярослав Сергійович  (*прізвище, ім’я, по батькові*)  варіант № 17 | Зарахована “ ” “ ” 20 р.  викладачем  Шкурат Оксаною Сергіївною  (*прізвище, ім’я, по батькові*) |

Київ 2023

# 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.

1. 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

|  |
| --- |
| Lab2.java |
| package lab2;  import java.awt.\*;  import java.awt.event.ActionEvent;  import java.awt.event.ActionListener;  import javax.swing.\*;  @SuppressWarnings("serial")  public class lab2 extends JPanel implements ActionListener {      private double angle = 0;      private double scale = 1;      private double delta = 0.01;        private static int maxWidth;      private static int maxHeight;        public lab2() {          Timer timer =  new Timer(10, this);          timer.start();      }          public void paint(Graphics g) {          super.paint(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(255, 255, 128));          g2d.clearRect(0, 0, maxWidth + 1, maxHeight + 1);            drawFrame(g2d);          g2d.translate(maxWidth/2, maxHeight/2);            g2d.rotate(angle);          g2d.setComposite(AlphaComposite.getInstance(AlphaComposite.SRC\_OVER,                  (float) scale));          Ship ship = new Ship();          ship.drawShip(g2d);        }      private void drawFrame(Graphics2D g2d) {          g2d.setColor(Color.green);          BasicStroke bs2 = new BasicStroke(16, BasicStroke.CAP\_ROUND,                  BasicStroke.JOIN\_MITER);          g2d.setStroke(bs2);          g2d.drawRect(50, 15, maxWidth - 100, maxHeight - 30);      }      public static void main(String[] args) {          JFrame frame = new JFrame("LAB 2");          frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);          frame.setSize(900, 700);          frame.setLocationRelativeTo(null);          frame.setResizable(false);          frame.add(new lab2());          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;      }      @Override      public void actionPerformed(ActionEvent arg0) {          if (scale < 0.01) {              delta = -delta;          } else if (scale > 0.99) {              delta = -delta;          }          scale += delta;          angle += 0.01;          repaint();      }  } |

|  |
| --- |
| Ship.java |
| package lab2;  import java.awt.\*;  import java.awt.geom.GeneralPath;  public class Ship {      private void drawBasis(Graphics2D g2d) {          g2d.setColor(new Color(128, 64, 0));          double[][] points = {                  {-300, -10}, {-60, 70}, {140, 70}, {290, 0},                  {165, 170}, {-80, 170}, {-300, -10}          };          GeneralPath basis = new GeneralPath();          basis.moveTo(points[0][0], points[0][1]);          int length = points.length;          for(int i = 0; i < length; i++) {              basis.lineTo(points[i][0], points[i][1]);          }          basis.closePath();          g2d.fill(basis);      }        private void drawMast(Graphics2D g2d) {          g2d.setColor(Color.black);          g2d.fillRect(0,  -60,  13,  130);      }        private void drawSail(Graphics2D g2d) {          GradientPaint gp = new GradientPaint(10, 20, new Color(192, 192, 192), 20, 10, Color.blue, true);          g2d.setPaint(gp);          double[][] points = {                  {-90, -130}, {-60, -230}, {130, -230}, {90, -110},                  {130, -20}, {-70, -30}, {-90, -130}          };          GeneralPath sail = new GeneralPath();          sail.moveTo(points[0][0], points[0][1]);          int length = points.length;          for(int i = 0; i < length; i++) {              sail.lineTo(points[i][0], points[i][1]);          }          sail.closePath();          g2d.fill(sail);      }        void drawShip(Graphics2D g2d) {          drawBasis(g2d);          drawMast(g2d);          drawSail(g2d);      }  } |

# Результат



