Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №7

Выполнила:

Студент 3 курса

Группы ПО-5

Нерода А.А.

Проверил:

Крощенко А.А.

Брест 2021

# Вариант-10

**Цель работы:** освоить возможности языка программирования Java в построении графических приложений.

**Задание 1.** Изобразить в апплете приближающийся издали и удаляющийся шар. Шар должен двигаться с постоянной скоростью.

**Код программы**

import java.awt.\*;

import java.awt.event.ActionEvent;

import javax.swing.\*;

import java.awt.event.ActionListener;

import javax.swing.Timer;

public class Ball extends JPanel implements ActionListener {

    Timer timer = new Timer(10,this);

    int x=100;

    int y=150;

    int z=100;

    int c=100;

    int a = 0, b = 0;

    @Override

    public void paintComponent(Graphics g){

        super.paintComponent(g);

        this.setBackground(Color.WHITE);

        g.setColor(Color.red);

        g.fillOval(x, y, z, c);

        timer.start();

        if (a == 120)

        {

            b =+ 38;

            a = 0 - b;

        }

    }

    public static void main(String[] args) {

        Ball java = new Ball();

        JFrame frame = new JFrame("Bull");

        frame.setSize(500, 500);

        frame.add(java);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setVisible(true);

        frame.setLocationRelativeTo(null);

    }

    @Override

    public void actionPerformed(ActionEvent e) {

        if (a <= 40)

        {

            x++; y++; z--;

            z--; c--; c--;

        }

        else

        {

            x--; y--; z++;

            z++; c++; c++;

        }

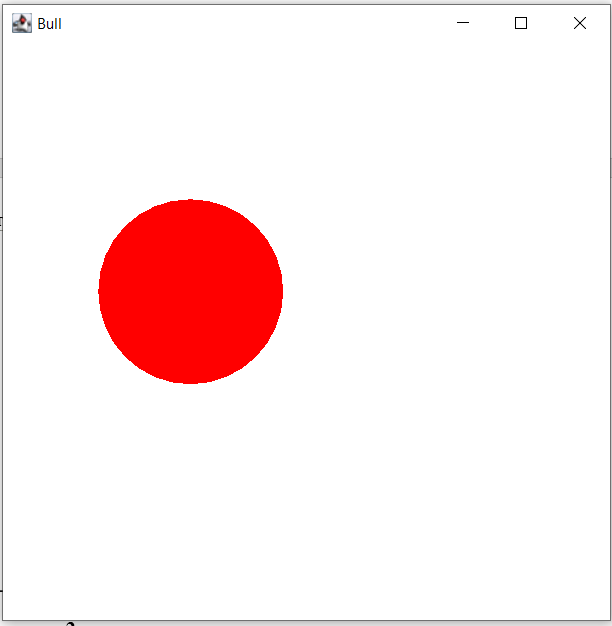
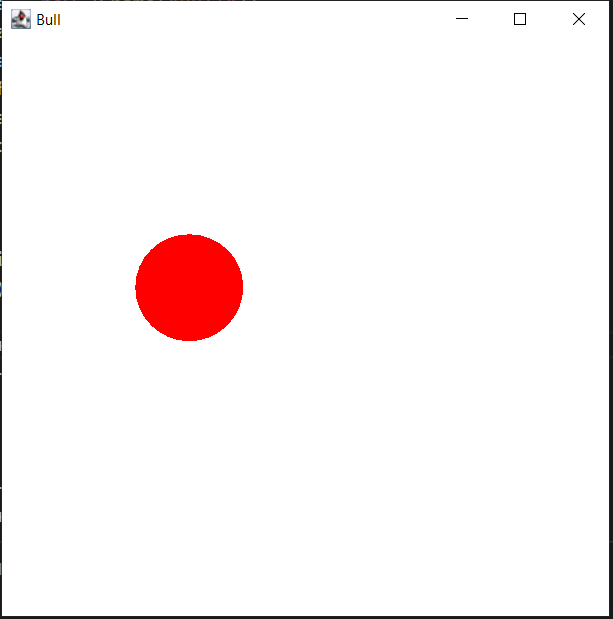
        a++;

        repaint();

    }

}

**Результаты работы**



**Задание 2.** Кривая Пеано.

**Код программы**

import java.awt.Color;

import java.awt.Graphics;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import java.util.Scanner;

import javax.imageio.ImageIO;

import java.awt.Dimension;

import java.awt.Graphics;

import java.awt.image.BufferedImage;

import javax.swing.JFrame;

public class PeanoCurve {

    private static boolean debug = false;

    private static int xi = 100;

    private static int yi = 100;

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out

                .println("Enter the iteration of the Peano Curve you want to generate (n >= 1)");

        if (scanner.hasNextInt()) {

            UserInterface ui = new UserInterface();

            while (scanner.hasNextInt()) {

                int iteration = scanner.nextInt();

                if (iteration < 0)

                    break;

                DrawPeanoCurve(ui, iteration);

            }

        }

        scanner.close();

        System.exit(0);

    }

    public static void DrawPeanoCurve(UserInterface ui, int iteration) {

        int width = ui.getWidth();

        int height = ui.getHeight();

        int hsl = getPeanoSize(iteration);

        if (iteration > 5) {

            width = hsl \* 2 + 2 \* xi;

            height = hsl \* 2 + 2\*yi;

            System.out.println("Generating image at size "+width+"x"+height);

        }

        BufferedImage img = new BufferedImage(width, height, BufferedImage.TYPE\_BYTE\_BINARY);

        Graphics g = img.getGraphics();

        g.setColor(Color.WHITE);

        g.fillRect(0, 0, width, height);

        g.setColor(Color.BLACK);

        int state = 0;

        int x = xi;

        int y = height - yi;

        int d = (width < height) ? (width - 2\*xi) / hsl : (height - 2\*yi) / hsl;

        int dx = x;

        int dy = y;

        String curve = generateHilbertCurve(iteration);

        for (int i = 0; i < curve.length(); i++) {

            switch (curve.charAt(i)) {

            case '+':

                state -= 90;

                if (state < 0) state += 360;

                break;

            case '-':

                state += 90;

                state %= 360;

                break;

            case 'F':

                if (debug) System.out.println("state = "+state);

                switch(state) {

                case 0:

                    dx = x + d;

                    dy = y;

                    break;

                case 90:

                    dx = x;

                    dy = y - d;

                    break;

                case 180:

                    dx = x - d;

                    dy = y;

                    break;

                case 270:

                    dx = x;

                    dy = y + d;

                    break;

                }

                if (debug) System.out.println("drawing line from ("+x+","+y+") to ("+dx+","+dy+")");

                g.drawLine(x, y, dx, dy);

                x = dx;

                y = dy;

                break;

            }

        }

        File outputfile = new File("peano-"+iteration+".png");

        try {

            ImageIO.write(img, "png", outputfile);

            System.out.println("Finished writing image.");

        } catch (IOException e) {

            e.printStackTrace();

        }

        if (iteration <= 5) {

            ui.setImage(img);

            ui.repaint();

            ui.setVisible(true);

        } else {

            ui.setVisible(false);

        }

    }

    public static String generateHilbertCurve(int iteration) {

        String curve = "-A";

        String axiomA = "CFDFC+F+DFCFD-F-CFDFC";

        String axiomB = "DFCFD-F-CFDFC+F+DFCFD";

        for (int i = 0; i < iteration; i++) {

            curve = curve.replace("A", axiomA);

            curve = curve.replace("B", axiomB);

            curve = curve.replace("C", "A");

            curve = curve.replace("D", "B");

            if (debug)

                System.out.println("curve at iteration #" + (i + 1) + " = "

                        + curve);

        }

        System.out.println("Generated curve is "+curve.length()+" characters long.");

        return curve;

    }

    private static int getPeanoSize(int iteration) {

        int result = 2;

        for (int i = 1; i < iteration; i++) {

            result \*= 3;

            result += 2;

        }

        return result;

    }

}

class UserInterface extends JFrame {

    private static final long serialVersionUID = 5051321847886284731L;

    private final static String title = "Space Filling Curves";

    private final int width = 1000;

    private final int height = 1000;

    private BufferedImage img;

    public UserInterface() {

        super(title);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setSize(new Dimension(width, height));

        setVisible(true);

    }

    public void paint(Graphics g) {

        g.drawImage(img, 0, 0, null);

    }

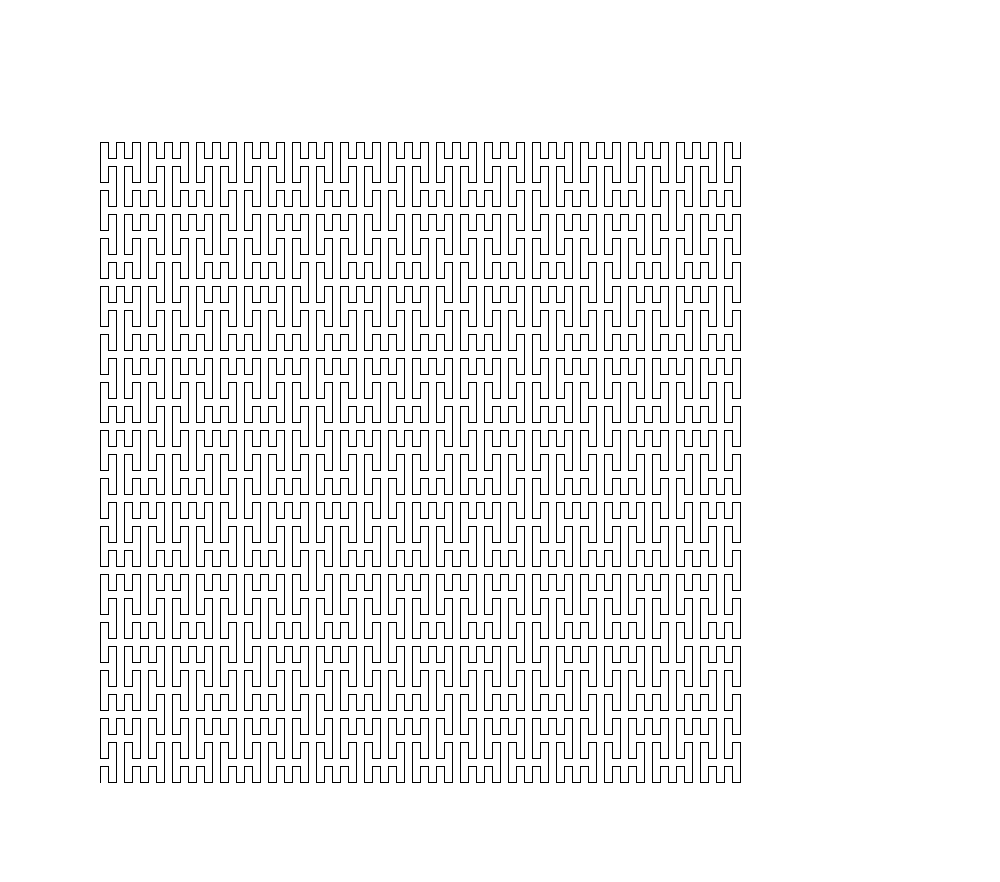
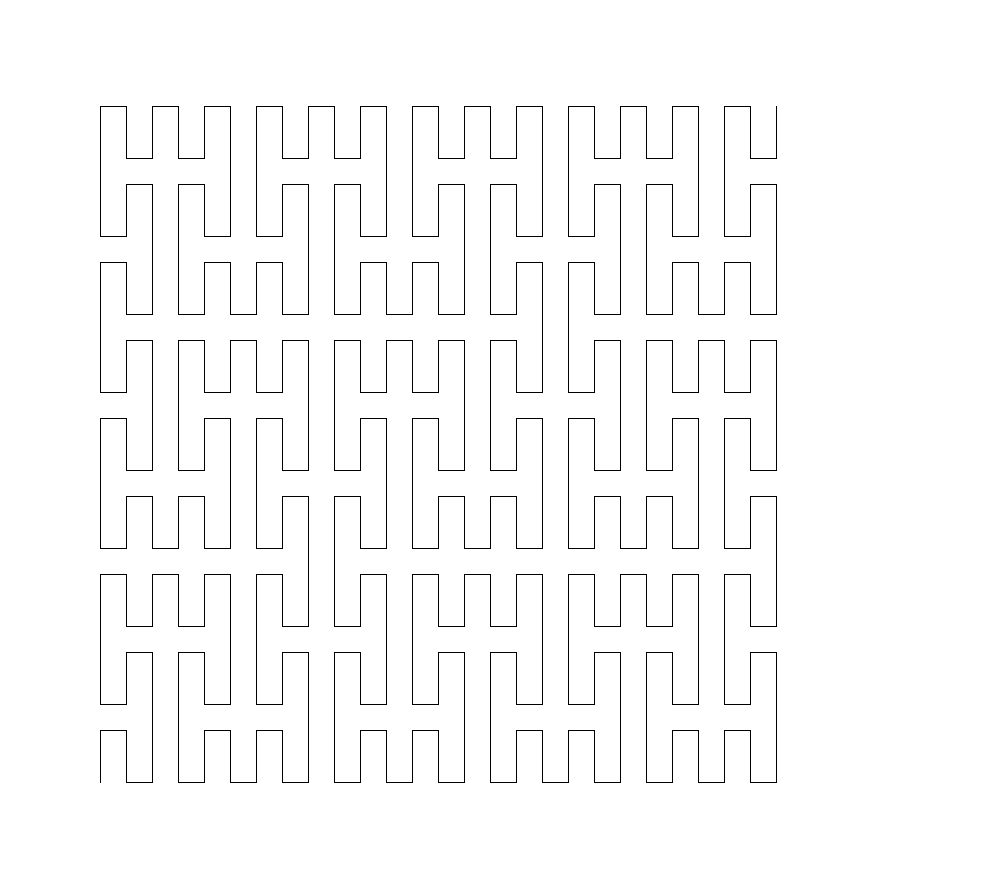
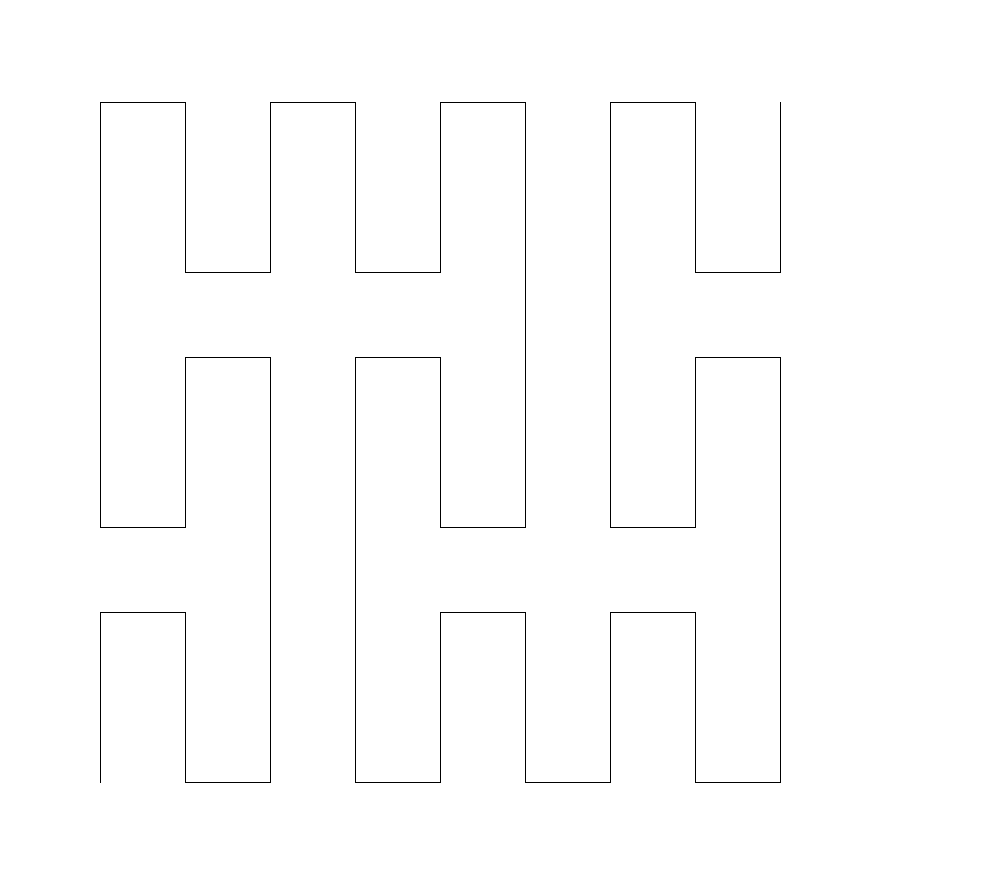
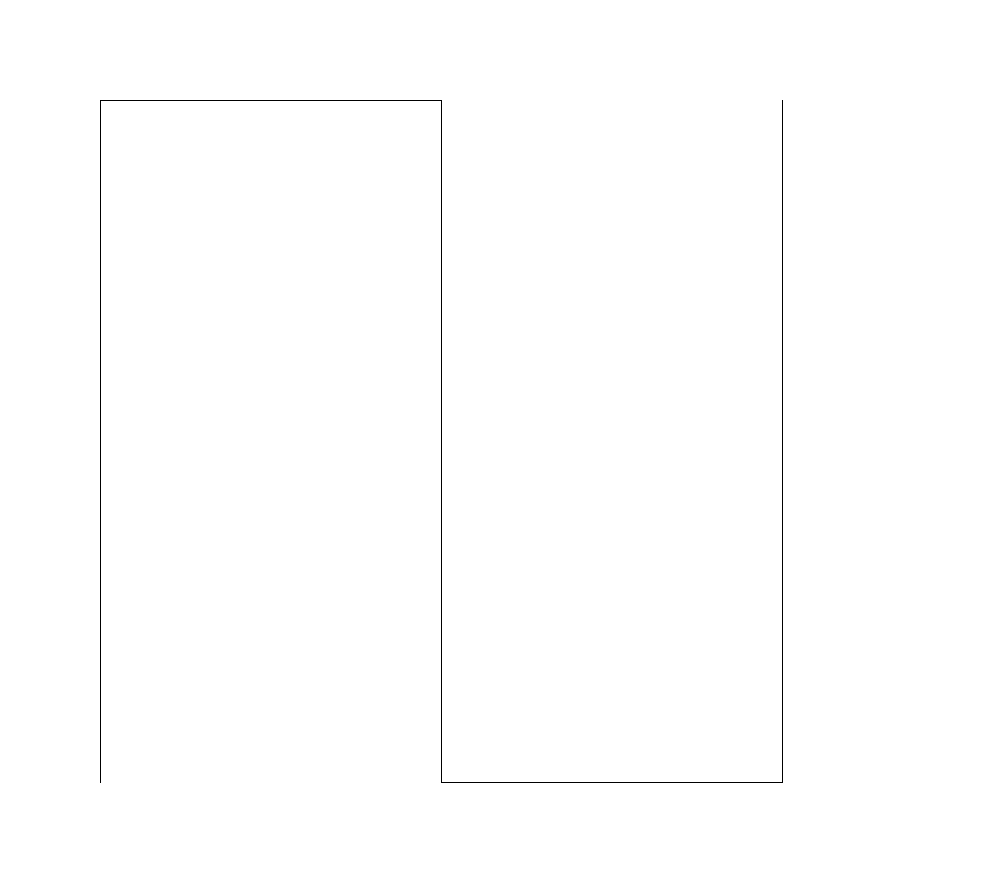
    public void setImage(BufferedImage img) {

        this.img = img;

    }

}

**Результаты работы**

****

**Вывод:** в ходе работы освоила возможности языка программирования Java в построении графических приложений.