Федеральное агентство связи

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

Отчет по лабораторной работе №4

по дисциплине «Технологии разработки программного обеспечения»

Выполнил: студент группы БВТ1803

Маркушин Андрей Васильевич

Руководитель:

Мосева Марина Сергеевна

Москва 2020

Цель работы: создать небольшое JAVA-приложение, которое сможет рисовать фракталы.

Выполнение:

JImageDisplay:

```
import javax.swing.*;
import java.awt.image.*;
import java.awt.*;

public class JImageDisplay extends JComponent{
    private BufferedImage image;

    public JImageDisplay(int w, int h) {
        image = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);//задаем
размер
        Dimension imageDimension = new Dimension(w, h); // создаем
        super.setPreferredSize(imageDimension); //отображение
    }

    //отрисовка
    public void paintComponent (Graphics g){
        super.paintComponent(g);
        g.drawImage (image, 0, 0, image.getWidth(), image.getHeight(), null);
    }

    //очистка
    public void clearImage(){
        for (int x=0;x<image.getWidth();x++)
            for (int y=0;y<image.getHeight();y++)
                image.setRGB(x,y,0);
    }

    //закрашивание 1 пикселя
    public void drawPixel(int x, int y, int rgbColor){
        image.setRGB(x,y,rgbColor);
    }
}
```

Mandelbrot:

```
import java.awt.geom.Rectangle2D;

public class Mandelbrot extends FractalGenerator {
    public static final int MAX_ITERATIONS = 2000;

    @Override
    public void getInitialRange(Rectangle2D.Double range) {
        range.x = -2;
        range.y = -1.5;
        range.width = 3;
        range.height = 3;
    }

    @Override
    public int numIterations(double x, double y) {
        double real = 0; //действительная часть
        double noreal = 0;//мнимая часть
        int i;
        for (i=0; i<MAX_ITERATIONS && real*real+noreal*noreal < 4; i++){
            double real2 = real*real - noreal*noreal + x;
            double notreal2 = 2 * real * noreal + y;
            real = real2;
            noreal = notreal2;
```

```
        }
        if (i == MAX_ITERATIONS) return -1;
        return i;
    }
}
```

FractalExplorer:

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.awt.geom.Rectangle2D;

public class FractalExplorer {
    private int sizeDisplay;
    private JImageDisplay display;
    private FractalGenerator fractal;
    private Rectangle2D.Double range;

    static public void main(String args[]){
        FractalExplorer displayExplorer = new FractalExplorer(700);
        displayExplorer.createAndShowGUI();
        displayExplorer.drawFractal();
    }

    public FractalExplorer(int sizeDisplay) {
        this.sizeDisplay = sizeDisplay;
        range = new Rectangle2D.Double();
        fractal = new Mandelbrot();
        fractal.getInitialRange(range);
        display = new JImageDisplay(sizeDisplay,sizeDisplay);
    }

    public void createAndShowGUI(){
        display.setLayout(new BorderLayout());
        JFrame frame = new JFrame("Fractal");
        frame.add(display, BorderLayout.CENTER);     //дисплей

        EventBtn eventBtn = new EventBtn();          //события кнопок
        EventMouse eventMouse = new  EventMouse();   //события кнопок

        display.addMouseListener(eventMouse);

        JButton btnReset = new JButton();            //кнопка сброса
        btnReset.addActionListener(eventBtn);
        btnReset.setText("Reset");
        frame.add(btnReset, BorderLayout.SOUTH);

        frame.pack ();               //размещение
        frame.setVisible (true);     //видимость
        frame.setResizable (false); //запрет изменения размера окна
        drawFractal();
        frame.repaint();             //обновить дисплей
    }

    private void drawFractal() {
        for (int x = 0; x < sizeDisplay; x++) {
            for (int y = 0; y < sizeDisplay; y++) {
                double xCoord = fractal.getCoord(range.x, range.x +
range.width, sizeDisplay, x);
                double yCoord = fractal.getCoord(range.y, range.y +
range.height, sizeDisplay, y);

                int numIters = fractal.numIterations(xCoord, yCoord);

                if (numIters == -1){
                    display.drawPixel(x, y, 0);
                }
```

```
                else {
                    float hue = 0.7f + (float) numIters / 200f;
                    int rgbColor = Color.HSBtoRGB(hue, 1f, 1f);
                    display.drawPixel(x, y, rgbColor);
                }
            }
        }
        display.repaint();
    }

    private class EventBtn implements ActionListener{
        public void actionPerformed(ActionEvent e)
        {
            fractal.getInitialRange(range);
            drawFractal();
        }
    }

    private class EventMouse extends MouseAdapter {
        public void mouseClicked(MouseEvent e)
        {
            int x = e.getX();
            double xCoord = fractal.getCoord(range.x,range.x + range.width,
sizeDisplay, x);

            int y = e.getY();
            double yCoord = fractal.getCoord(range.y,range.y + range.height,
sizeDisplay, y);

            fractal.recenterAndZoomRange(range, xCoord, yCoord, 0.5);

            drawFractal();
        }
    }
}
```

Скриншоты программы: