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 №5

по дисциплине «Технологии разработки программного обеспечения»

Выполнил: студент группы БВТ1801

Протасова Елена Сергеевна

Руководитель:

Мосева Марина Сергеевна

Москва 2020

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

Выполнение:

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

public class JImageDisplay extends JComponent {
    private BufferedImage image;
    public JImageDisplay(int width, int height){
        image = new BufferedImage(width,height, BufferedImage.TYPE_INT_RGB);
        Dimension dimension = new Dimension(width,height);
        super.setPreferredSize(dimension); //ваш компонент будет включен в
пользовательский интерфейс, он отобразит на экране все изображение
    }
    public void paintComponent (Graphics g){
        super.paintComponent(g);
        g.drawImage (image, 0, 0, image.getWidth(), image.getHeight(), null);
//нарисовать изображение в компоненте
    }
    public void clearImage (){
        for (int i=0; i<image.getWidth(); i++){
            for (int j=0; j<image.getHeight(); j++){
                image.setRGB(i,j,0);
            }
        }
    }
    public void drawPixel (int x, int y, int rgbColor){
        image.setRGB(x,y,rgbColor);
    }
}
```

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

public class Mandelbrot extends FractalGenerator {
    @Override
    public void getInitialRange(Rectangle2D.Double range) {
        range.x = -2;
        range.y = -1.5;
        range.width = 3;
        range.height = 3;
    }
    public static final int MAX_ITERATIONS = 2000;
    @Override
    public int numIterations(double x, double y) {
        double rez = 0, imz = 0;
        int n = 0;
        while (((rez*rez)+(imz*imz))<4 && n<2000){
            double ri = rez;
            rez = rez*rez - imz * imz + x;
            imz = 2*ri*imz + y;
            n++;
        }
        if (n>=2000) return -1;
        return n;
```

```
    }
}
```

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

public class FractalExplorer {
    private int size;
    private JImageDisplay jImageDisplay;
    private  FractalGenerator fractalGenerator;
    private Rectangle2D.Double rectangle2D;
    public FractalExplorer(int size){
        this.size = size;
        rectangle2D = new Rectangle2D.Double();
        fractalGenerator = new Mandelbrot();
        fractalGenerator.getInitialRange(rectangle2D);
        jImageDisplay = new JImageDisplay(size,size);
    }

    public static void main(String[] args) {
        FractalExplorer fractalExplorer = new FractalExplorer(600);
        fractalExplorer.createAndShowGUI();
        fractalExplorer.drawFractal();
    }
    public void createAndShowGUI() {
        EventButton eventButton = new EventButton();
        EventMouse eventMouse = new EventMouse();

        JFrame jFrame = new JFrame("Fractal");
        jImageDisplay.addMouseListener(eventMouse);
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jFrame.add(jImageDisplay,BorderLayout.CENTER );

        JButton jButton = new JButton("Reset");
        jButton.addActionListener(eventButton);
        jFrame.add(jButton, BorderLayout.SOUTH);

        jFrame.pack ();
        jFrame.setVisible (true);
        jFrame.setResizable (false);
    }
    private void drawFractal() {
        for (int x=0; x<size; x++) {
            for (int y=0; y<size; y++){
                double xCoord = FractalGenerator.getCoord(rectangle2D.x,
rectangle2D.x + rectangle2D.width, size, x);
                double yCoord = FractalGenerator.getCoord(rectangle2D.y,
rectangle2D.y + rectangle2D.height, size, y);
                int numI = fractalGenerator.numIterations(xCoord,yCoord);
                if (numI==-1) jImageDisplay.drawPixel(x,y,0);
                else {
                    float hue = 0.7f + (float)numI / 200f;
                    int rgbColor = Color.HSBtoRGB(hue, 1f, 1f);
                    jImageDisplay.drawPixel(x,y,rgbColor);
                }
            }
```

```
        }
        jImageDisplay.repaint();
    }
    private class EventButton implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            fractalGenerator.getInitialRange(rectangle2D);
            drawFractal();
        }
    }
    private class EventMouse extends MouseAdapter {
        @Override
        public void mouseClicked(MouseEvent e) {
            int x = e.getX();
            int y = e.getY();
            double xCoord = FractalGenerator.getCoord(rectangle2D.x, rectangle2D.x +
rectangle2D.width, size, x);
            double yCoord = FractalGenerator.getCoord(rectangle2D.y, rectangle2D.y +
rectangle2D.height, size, y);
            fractalGenerator.recenterAndZoomRange(rectangle2D,xCoord, yCoord, 0.5);
            drawFractal();
        }
    }
}
```