**Федеральное агентство связи**

**Ордена трудового Красного Знамени федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математическая кибернетика и информационные технологии

Отчет по лабораторной работе № 4

по дисциплине «Объектно-ориентированное программирование»

Выполнил: студент группы БУТ1952

Обухов А.Д.

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020

Цель работы:

Изучить использование фреймворка Swing и Java API для реализации графических интерфейсов.

Задание.

1. Реализовать класс JImageDisplay для отображения графического интерфейса.
2. Реализовать класс Mandelbrot, наследующий от класса FractalGenerator.
3. Реализовать класс FractalExplorer для исследования и отображения различных областей фрактала.

В ходе выполнения задания были реализованы классы JImageDisplay, Mandelbrot и FractalExplorer.  
Исходный код JImageDisplay:

import javax.swing.\*;

import java.awt.\*;

import java.awt.image.BufferedImage;

public class JImageDisplay extends JComponent {

private BufferedImage image;

public JImageDisplay(int h, int w) {

image = new BufferedImage(h, w, BufferedImage.TYPE\_INT\_RGB);

super.setPreferredSize(new Dimension(h, w));

}

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

g.drawImage(image, 0, 0, image.getWidth(), image.getHeight(), null);

}

public void drawPixel(int x, int y, int rgbColor) {

image.setRGB(x, y, rgbColor);

}

public void clearImage() {

for (int i = 0; i < image.getWidth(); i++) {

for (int j = 0; i < image.getHeight(); j++) {

drawPixel(i, j, 0);

}

}

}

}

Исходный код 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) {

int count = 0;

double re = 0;

double im = 0;

double z = 0;

while (count < MAX\_ITERATIONS && z < 4) {

count++;

double nextRe = re\*re - im\*im + x;

double nextIm = 2 \* re \* im + y;

z = nextRe\*nextRe + nextIm\*nextIm;

re = nextRe;

im = nextIm;

}

return count < MAX\_ITERATIONS ? count : -1;

}

}

Исходный код FractalExplorer:

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.awt.geom.Rectangle2D;

public class FractalExplorer {

private int size;

private JImageDisplay display;

private FractalGenerator generator;

private Rectangle2D.Double range;

public FractalExplorer(int size) {

this.size = size;

display = new JImageDisplay(size, size);

generator = new Mandelbrot();

range = new Rectangle2D.Double();

}

public void createAndShowGUI() {

JFrame frame = new JFrame("Explorer");

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

frame.getContentPane().setLayout(new BorderLayout());

frame.getContentPane().add(display, BorderLayout.CENTER);

JButton button = new JButton("Reset");

button.setActionCommand("reset");

button.addActionListener(new ButtonHandler());

frame.getContentPane().add(button, BorderLayout.SOUTH);

frame.getContentPane().addMouseListener(new MouseHandler());

frame.pack();

frame.setVisible(true);

frame.setResizable(false);

}

private void drawFractal() {

float hue;

for (int y = 0; y < size; y++){

for (int x = 0; x < size; x++){

//x - пиксельная координата; xCoord - координата в пространстве фрактала

double xCoord = FractalGenerator.getCoord(range.x, range.x + range.width, size, x);

double yCoord = FractalGenerator.getCoord(range.y, range.y + range.height, size, y);

int iterations = generator.numIterations(xCoord, yCoord);

hue = 0.7f + (float) iterations / 200f;

int rgbColor = Color.HSBtoRGB(hue, 1f, 1f);

display.drawPixel(x, y, rgbColor);

}

}

display.repaint();

}

public class ButtonHandler implements ActionListener {

public void actionPerformed(ActionEvent e) {

String cmd = e.getActionCommand();

if (cmd.equals("reset")){

range = new Rectangle2D.Double();

generator.getInitialRange(range);

drawFractal();

}

}

}

public class MouseHandler extends MouseAdapter {

public void mouseClicked(MouseEvent e) {

double x = FractalGenerator.getCoord(range.x, range.x + range.width, size, e.getX());

double y = FractalGenerator.getCoord(range.y, range.y + range.height, size, e.getY());

generator.recenterAndZoomRange(range, x, y, 0.5);

drawFractal();

}

}

public static void main(String[] args) {

FractalExplorer window = new FractalExplorer(800);

window.createAndShowGUI();

window.drawFractal();

}

}

При запуске программы открывается окно, отображающее исходный фрактал.

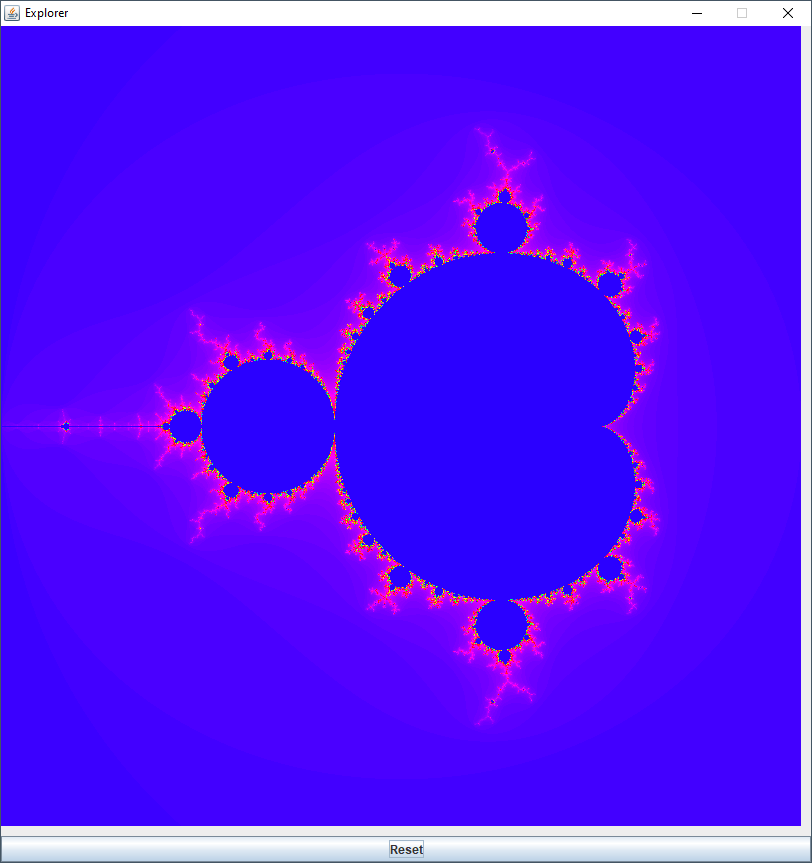
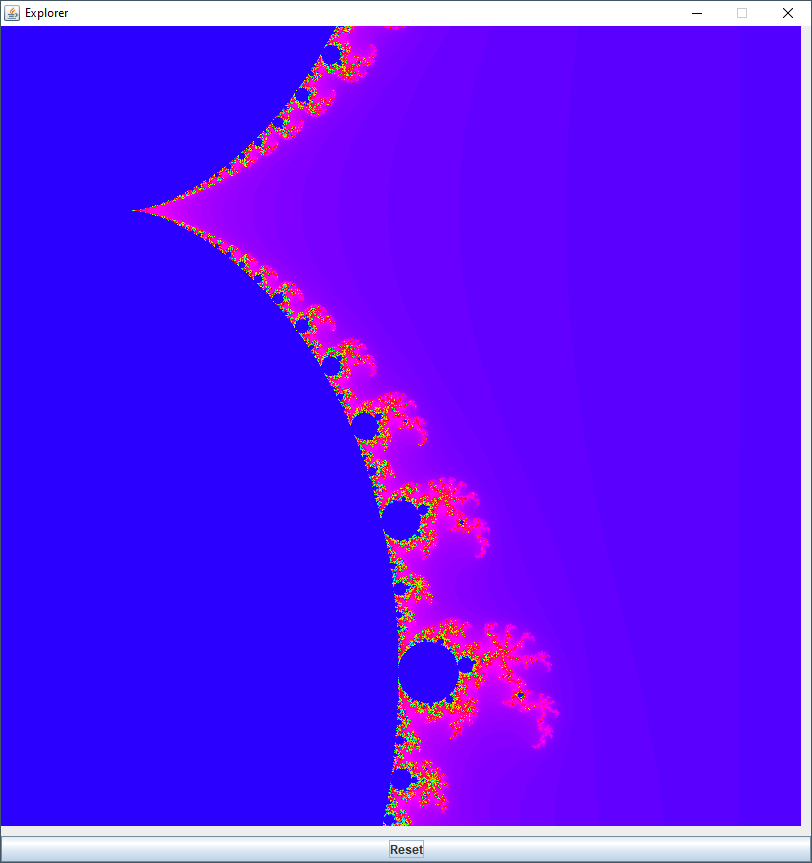
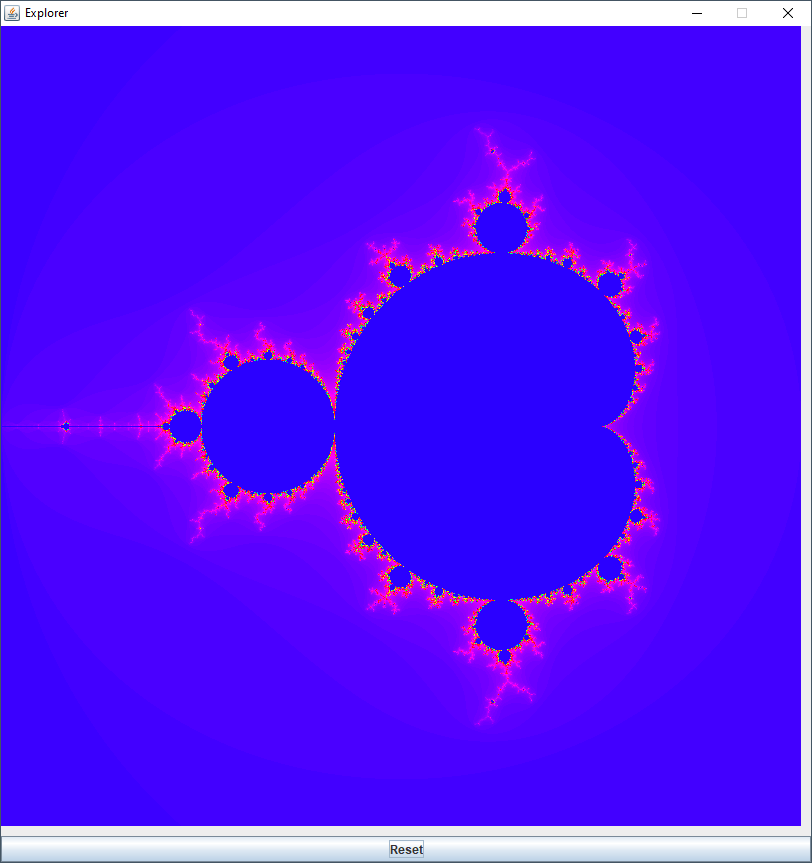


Рис. 1 – окно программы.

При нажатии кнопкой мыши на окно происходит масштабирование и перерисовка фрактала.

Рис. 2 – увеличенный фрактал.

При нажатии на кнопку Reset изображение возвращается к исходному.

  
Рис. 3 – исходное изображение после нажатия кнопки Reset

Заключение.

В ходе выполнения этой лабораторной работы была изучена работа с фреймворком Swing и Java API для реализации графических приложений.