**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

«Рисование фракталов»

по дисциплине «Объектно-ориентированное программирование»

Выполнила студентка

группы БУТ 1952

Рюмина К.С.

Москва, 2020

Целью данной лаборторной работы является создание Java-приложения, которое сможет рисовать фраклаты с помощью Swing и Java API, которые позволяют создавать графисекие пользовательсткие интерфейсы.

**Задания**

1. Создать класс JImageDisplay, а также реализовать его для отображения графического интерфейса.
2. Реализовать класс Mandelbrot, наследующий класса FractalGenerator.
3. Создать и реализовать класс FractalExplorer для того, чтобы исследовать и отобразить различные области фрактала.

Для выполнения данной работы были созданы и реализованы классы:

1. JimageDisplay:

mport javax.swing.JComponent;  
import java.awt.image.BufferedImage;  
import java.awt.Dimension;  
import java.awt.Graphics;  
  
public class JImageDisplay extends JComponent{  
  
 private BufferedImage imgBuf;  
 static final long serialVersionUID = 0;  
  
 public JImageDisplay(int width, int height){  
 imgBuf = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_RGB);  
  
 Dimension size = new Dimension(width, height);  
 super.setPreferredSize(size);  
 }  
  
 public void paintComponent(Graphics g){  
 super.paintComponent(g);  
 g.drawImage(imgBuf, 0, 0, imgBuf.getWidth(), imgBuf.getHeight(), null);  
 }  
  
 public void drawPixel(int x, int y, int rgbColor) {  
 imgBuf.setRGB(x, y, rgbColor);  
 }  
  
 public void clearImage() {  
 for (int i = 0; i < imgBuf.getWidth(); i++)  
 for (int j = 0; i < imgBuf.getHeight(); j++)  
 drawPixel(i, j, 0);  
 }  
}

1. Mandelbrot:

import java.awt.geom.Rectangle2D;  
  
public class Mandelbrot extends FractalGenerator {  
 public static final int MAX\_ITERATIONS = 2000;  
  
 public void getInitialRange(Rectangle2D.Double range) {  
 range.x = -2;  
 range.y = -1.5;  
  
 range.width = 3;  
 range.height = 3;  
 }  
  
 public int numIterations(double x, double y) {  
 int count = 0;  
  
 double re = 0;  
 double im = 0;  
 double z\_n2 = 0;  
  
 while (count < MAX\_ITERATIONS && z\_n2 < 4) {  
 count++;  
  
 double nextRe = re\*re - im\*im + x;  
 double nextIm = 2 \* re \* im + y;  
  
 z\_n2 = nextRe\*nextRe + nextIm\*nextIm;  
  
 re = nextRe;  
 im = nextIm;  
 }  
  
 return count < MAX\_ITERATIONS ? count : -1;  
 }  
}

1. FractalExplorer:

import java.awt.BorderLayout;  
import java.awt.Color;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.MouseEvent;  
import java.awt.event.MouseAdapter;  
import java.awt.geom.Rectangle2D;  
  
import javax.swing.\*;  
  
public class FractalExplorer {  
 private int displaySize;  
 private JImageDisplay img;  
 private FractalGenerator gen;  
 private Rectangle2D.Double range;  
  
 public FractalExplorer(int displaySize){  
 this.displaySize = displaySize;  
 img = new JImageDisplay(displaySize, displaySize);  
 gen = new Mandelbrot();  
 range = new Rectangle2D.Double();  
 }  
  
 public void createAndSHowGUI() {  
 JFrame frame = new JFrame("Fractal Explorer");  
 frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 frame.getContentPane().setLayout(new BorderLayout());  
 frame.getContentPane().add(img, 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 < displaySize; y++){  
 for (int x = 0; x < displaySize; x++){  
 // x is the pixel-coordinate; xCoord is the coordinate in the fractal's space  
 double xCoord = FractalGenerator.getCoord(range.x, range.x + range.width, displaySize, x);  
 double yCoord = FractalGenerator.getCoord(range.y, range.y + range.height, displaySize, y);  
 int iters = gen.numIterations(xCoord, yCoord);  
 hue = 0.7f + (float) iters / 200f;  
 int rgbColor = Color.HSBtoRGB(hue, 1f, 1f);  
 img.drawPixel(x, y, rgbColor);  
 }  
 }  
 img.repaint();  
 }  
  
 public class ButtonHandler implements ActionListener{  
 public void actionPerformed(ActionEvent e) {  
 String cmd = e.getActionCommand();  
 if (cmd.equals("reset")){  
 range = new Rectangle2D.Double();  
 gen.getInitialRange(range);  
 drawFractal();  
 }  
 }  
 }  
  
 public class MouseHandler extends MouseAdapter{  
 public void mouseClicked(MouseEvent e) {  
 double xCoord = FractalGenerator.getCoord(range.x, range.x + range.width, displaySize, e.getX());  
 double yCoord = FractalGenerator.getCoord(range.y, range.y + range.height, displaySize, e.getY());  
 gen.recenterAndZoomRange(range, xCoord, yCoord, 0.5);  
 drawFractal();  
 }  
 }  
  
 public static void main(String[] args){  
 FractalExplorer window = new FractalExplorer(600);  
 window.createAndSHowGUI();  
 window.drawFractal();  
 }  
}

При запуске программы открывается окно (рис.1) с отображением фрактала.

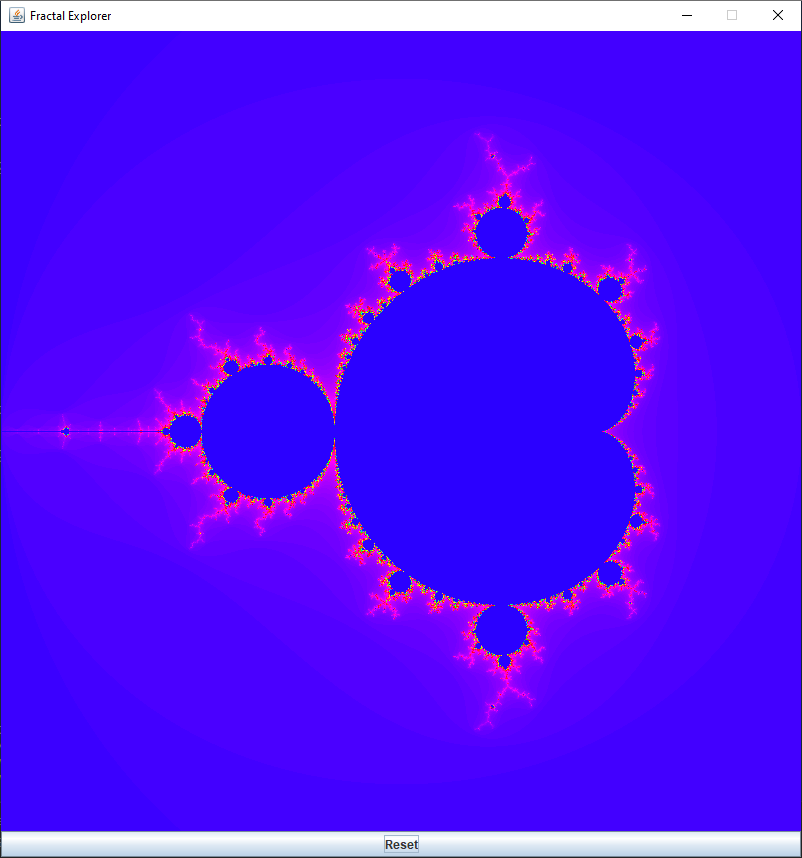


Рисунок 1 - окно программы

После нажатия на кнопку мыши в какой-либо стороне окна происходит масштабирование (рис.2).

При нажатии кнопки «Reset» изображение фрактала возвращается к исходному масштабу.

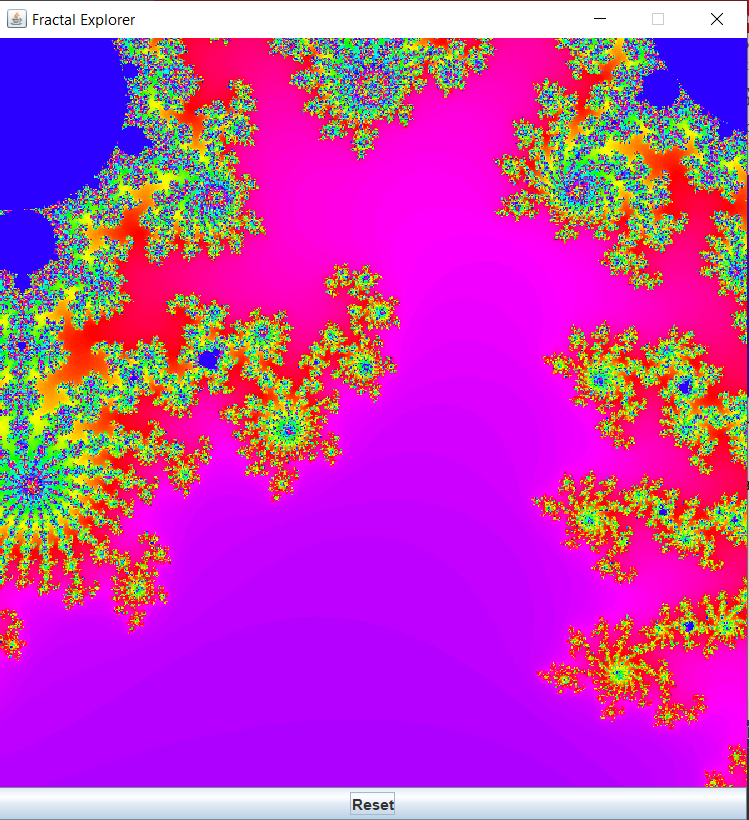


Рисунок 2 – увеличение области фрактала

**Заключение**

При выполнения данной работы я изучила работу Swing и Java API для реализации графических изображений.