ФГБОУ ВО

Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет) «МГТУ им. Н.Э. Баумана»

Факультет ИУ «Информатика и системы управления»

Кафедра ИУ-3 «Информационные системы и телекоммуникации»

Отчет по лабораторной работе 4

«**Программа для изучения фракталов**»

По курсу «Технология программирования»

Выполнил: Козлов П.В.

Группа: ИУ3-42Б

Проверила: Руденкова Ю.С.

Москва 2021

**Цель:** Ознакомиться с библиотеками Jswing, awt и создать программу для изучения фракталов Мандельброта.

**Задание:** Реализация UI и окна графического приложения, создание классов JimageDisplay, Mandelbrot (дочерний класс FractalGenerator)

**Решение:**

**Изменение в классе JImageDisplay:**

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

**Изменение в классе 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.**height** = 3;  
 range.**width** = 3;  
 range.**x** = -2;  
 range.**y** = -1.5;  
 }  
  
 @Override  
 **public int** numIterations(**double** x, **double** y) {  
  
 **int** count = 0;  
  
 **double** real = x;  
 **double** imagine = y;  
 **double** z\_multiplyZ = 0;  
  
 **while** (count<***MAX\_ITERATIONS***&&z\_multiplyZ<4){  
 count++;  
 **double** newReal= real\*real-imagine\*imagine+x;  
 **double** newImagine = 2\*real\*imagine+y;  
 z\_multiplyZ=newReal\*newReal+newImagine\*newImagine;  
 real=newReal;  
 imagine=newImagine;  
 }  
 **if**(***MAX\_ITERATIONS***==count){**return** -1;}  
 **else**{**return** count;}  
 }  
}

**Класс FractalExplorer:**

**import** javax.swing.\*;  
**import** javax.swing.plaf.basic.BasicTreeUI;  
**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;  
**import** java.awt.Color;  
**import** javax.swing.JButton;  
**import** javax.swing.JFrame;  
  
**public class** FractalExplorer {  
 **private int displaySize**;  
 **private** JImageDisplay **imageDisplay**;  
 **private** FractalGenerator;  
 **private** Rectangle2D.Double **range**;  
  
 **public** FractalExplorer(**int** displaySize) {  
  
 **this**.**displaySize** = displaySize;  
 **this**.**fractalGenerator** = **new** Mandelbrot();  
 **this**.**range** = **new** Rectangle2D.Double(0,0,0,0);  
 **fractalGenerator**.getInitialRange(**this**.**range**);  
 }  
  
 **public void** createAndShowGUI() {  
 JFrame frame = **new** JFrame(**"Fractal Explorer"**);  
 **imageDisplay** = **new** JImageDisplay(**displaySize**, **displaySize**);  
 JButton button = **new** JButton(**"Reset Display"**);  
 *//button.setPreferredSize(new Dimension(200,100));* ActionHandler aHandler = **new** ActionHandler();  
 MouseHandler mHandler = **new** MouseHandler();  
 button.addActionListener(aHandler);  
 **imageDisplay**.addMouseListener(mHandler);  
  
 frame.setLayout(**new** java.awt.BorderLayout());  
 frame.add(**imageDisplay**, BorderLayout.***CENTER***);  
 frame.add(button, BorderLayout.***SOUTH***);  
 frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);  
  
 frame.pack();  
 frame.setVisible(**true**);  
 frame.setResizable(**false**);  
 }  
  
 **public void** drawFractal() {  
 **for** (**int** i = 0; i < **displaySize**; i++) {  
 **for** (**int** j = 0; j < **displaySize**; j++) {  
 **double** xCoord = FractalGenerator.*getCoord*(**range**.**x**, **range**.**x** + **range**.**width**, **displaySize**, i);  
 **double** yCoord = FractalGenerator.*getCoord*(**range**.**x**, **range**.**x** + **range**.**width**, **displaySize**, j);  
 **double** count = **fractalGenerator**.numIterations(xCoord,yCoord);  
  
 **if**(count==-1){  
 **imageDisplay**.drawPixel(i,j,0);  
 }  
 **else**{  
 **float** hue = 0.7f + (**float**) count / 200f;  
 **int** rgbColor = Color.*HSBtoRGB*(hue, 1f, 1f);  
 **imageDisplay**.drawPixel(i,j,rgbColor);  
 }  
 }  
 }  
 **imageDisplay**.repaint();  
 }  
  
 **public class** ActionHandler **implements** ActionListener{  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 **fractalGenerator**.getInitialRange(**range**);  
 drawFractal();  
 }  
 }  
  
 **public class** MouseHandler **extends** MouseAdapter {  
 @Override  
 **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**.**width**, **displaySize**, e.getY());  
 **fractalGenerator**.recenterAndZoomRange(**range**, xCoord, yCoord, 0.5);  
 drawFractal();  
 }  
 }  
  
 **public static void** main(String[] args) {  
 FractalExplorer fracExp = **new** FractalExplorer(700);  
 fracExp.createAndShowGUI();  
 fracExp.drawFractal();  
 }  
}

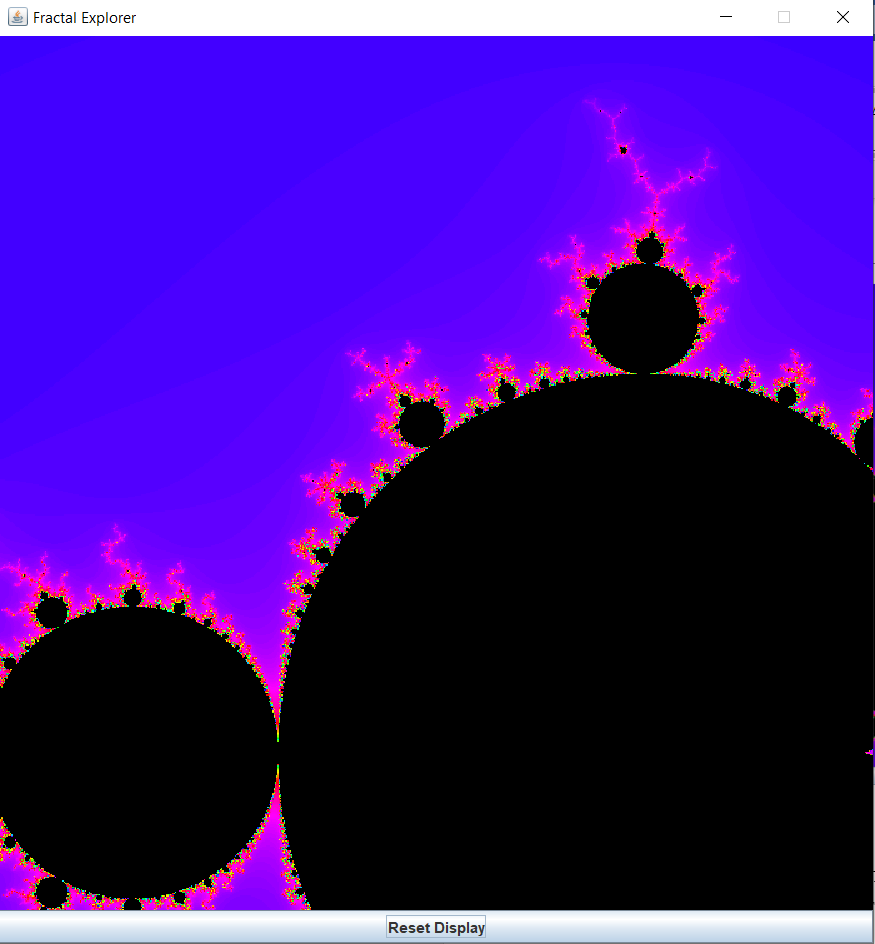
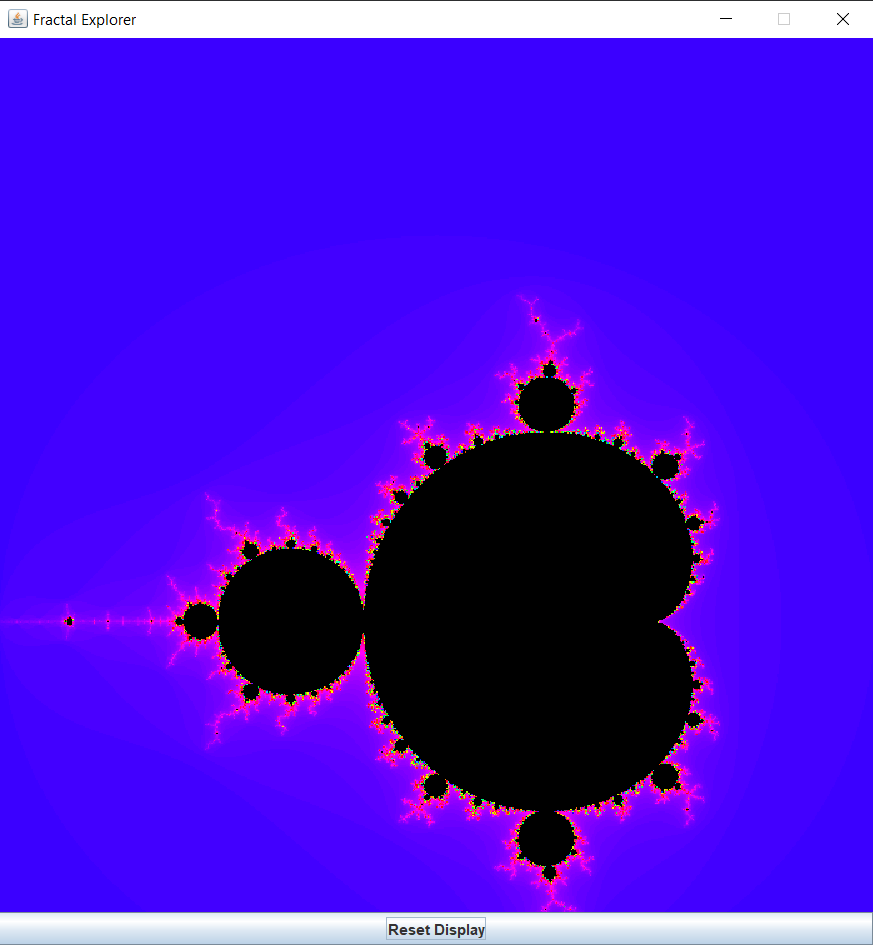


Рис.1Рис.2

На рис.1 можно увидеть прорисованный фрактал, на рис.2 мы видим приближенный прорисованный фрактал.

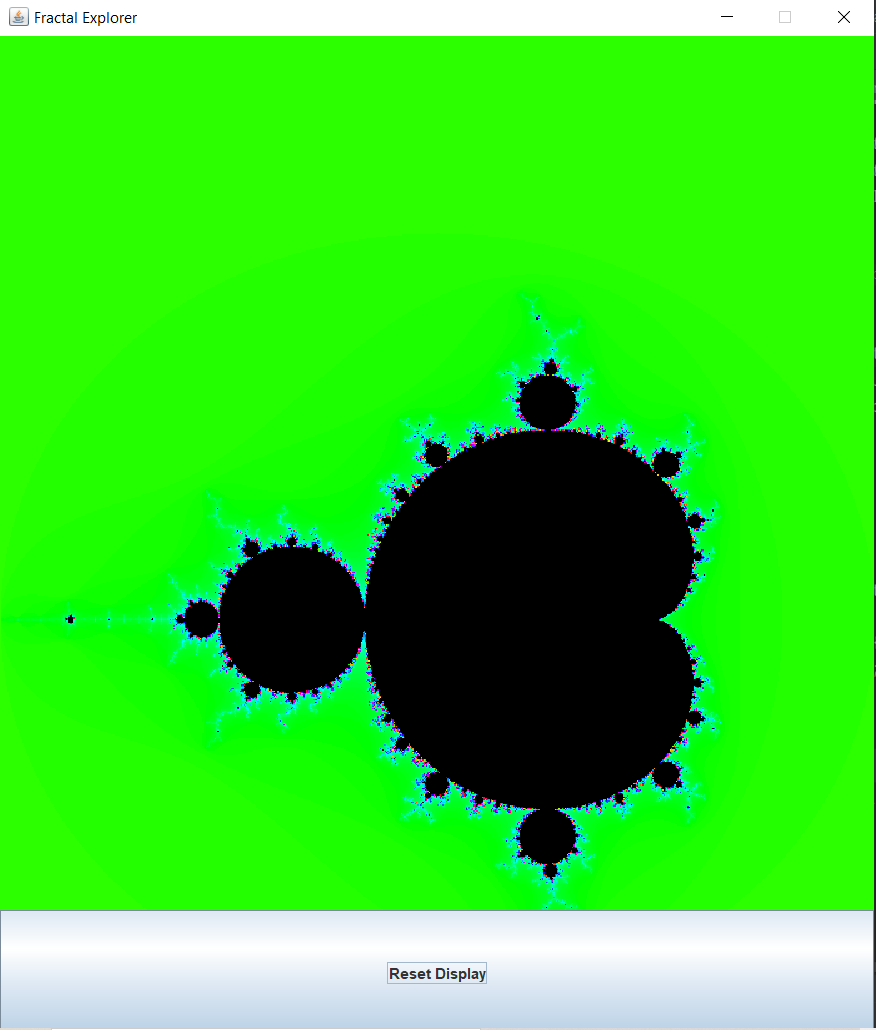


Рис.3

На рис.3 выполнено дополнительное задание, то есть измененный цвет фона, а также увеличенная в высоте кнопка “Reset Display”

**Вывод:**  В ходе выполнения лабораторной работы были получены теоретические знания в области работы с различными классами и интерфейсами. Были изучены основные методы, связывающие их, а также была реализовано переопределение методов в дочерних классах.