**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра Информатики

**Отчет по лабораторной работе №4**

по дисциплине «ООП»

**«Рисование фракталов»**

Выполнил: студент группы БВТ1901

Осин Д.Ю.

Руководитель:

Мосева Марина Сергеевна

Москва 2020

**Цель:**

Научиться писать алгоритм по вычислению фракталов, а также отрисовывать их в GUI посредством фреймворка Swing и Java API.

**Код программы:**

*public class* Mandelbrot *extends* FractalGenerator {  
 *public static final int* LIMIT = 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) {  
 ComplexNum cmplx = *new* ComplexNum(0, 0);  
 *int* iterator = 0;  
  
 *while* (iterator < LIMIT && cmplx.getSquaredModule() < 4) {  
 cmplx.makeSquaredInPoint(x, y);  
  
 iterator++;  
 }  
  
 *if* (iterator == LIMIT) *return* -1;  
  
 *return* iterator;  
 }  
}

*public class* ComplexNum {  
 *public double* rl;  
 *public double* im;  
  
 *public* ComplexNum(*double* rl, *double* im){  
 *this*.rl = rl;  
 *this*.im = im;  
 }  
  
 *public double* getSquaredModule() {  
 *return* (*this*.rl \* *this*.rl + *this*.im \* *this*.im);  
 }  
  
 *public void* makeSquaredInPoint(*double* x, *double* y) {  
 *double* real = (rl \* rl) - (im \* im) + x;  
 *double* imagine = 2 \* rl \* im + y;  
  
 rl = real;  
 im = imagine;  
 }  
}

*public class* FractalExplorer {  
 *private int* displaySize;  
 *private* JImageDisplay display;  
 *private* FractalGenerator fractal;  
 *private* Rectangle2D.Double range;  
  
 *public* FractalExplorer(*int* size) {  
 displaySize = size;  
  
 fractal = *new* Mandelbrot();  
 range = *new* Rectangle2D.Double();  
  
 fractal.getInitialRange(range);  
 display = *new* JImageDisplay(displaySize, displaySize);  
 }  
  
 *public void* createAndShowGUI() {  
 display.setLayout(*new* BorderLayout());  
 JFrame myframe = *new* JFrame("Fractal Explorer");  
  
 myframe.add(display, BorderLayout.CENTER);  
  
 JButton resetButton = *new* JButton("Reset Display");  
  
 Resetter handler = *new* Resetter();  
 resetButton.addActionListener(handler);  
  
 myframe.add(resetButton, BorderLayout.SOUTH);  
  
 Clicker click = *new* Clicker();  
 display.addMouseListener(click);  
  
 myframe.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
  
 myframe.pack();  
 myframe.setVisible(*true*);  
 myframe.setResizable(*false*);  
 }  
  
 *private void* drawFractal() {  
 *for* (*int* x = 0; x < displaySize; x++) {  
 *for* (*int* y = 0; y < displaySize; y++) {  
  
 *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* iteration = fractal.numIterations(xCoord, yCoord);  
  
 *if* (iteration == -1) {  
 display.drawPixel(x, y, 0);  
 } *else* {  
 *float* hue = 0.5f + (*float*) iteration / 50;  
 *int* rgbColor = Color.HSBtoRGB(hue, 1f, 1f);  
  
 display.drawPixel(x, y, rgbColor);  
 }  
  
 }  
 }  
 display.repaint();  
 }  
  
 *private class* Resetter *implements* ActionListener {  
 *public void* actionPerformed(ActionEvent e) {  
 fractal.getInitialRange(range);  
 drawFractal();  
 }  
 }  
  
 *private class* Clicker *extends* MouseAdapter {  
 *@Override  
 public void* mouseClicked(MouseEvent e) {  
 *int* x = e.getX();  
 *double* xCoord = FractalGenerator.getCoord(range.x,  
 range.x + range.width, displaySize, x);  
  
 *int* y = e.getY();  
 *double* yCoord = FractalGenerator.getCoord(range.y,  
 range.y + range.height, displaySize, y);  
  
 fractal.recenterAndZoomRange(range, xCoord, yCoord, 0.5);  
  
 drawFractal();  
 }  
 }  
  
 *public static void* main(String[] args) {  
 FractalExplorer displayExplorer = *new* FractalExplorer(800);  
 displayExplorer.createAndShowGUI();  
 displayExplorer.drawFractal();  
 }  
}

**Примеры работы программы:**

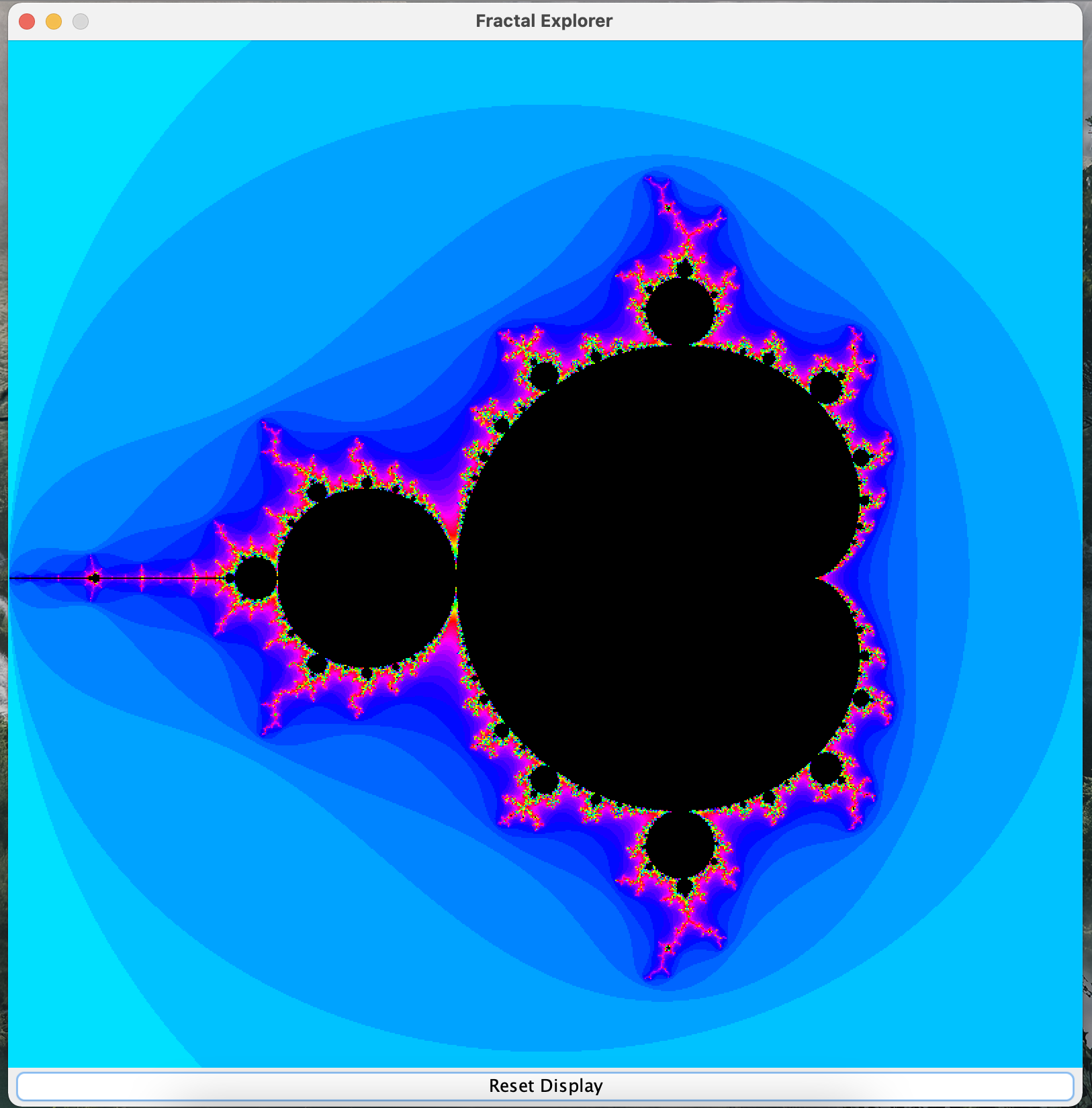
****

Рисунок 1 - Вывод фрактала

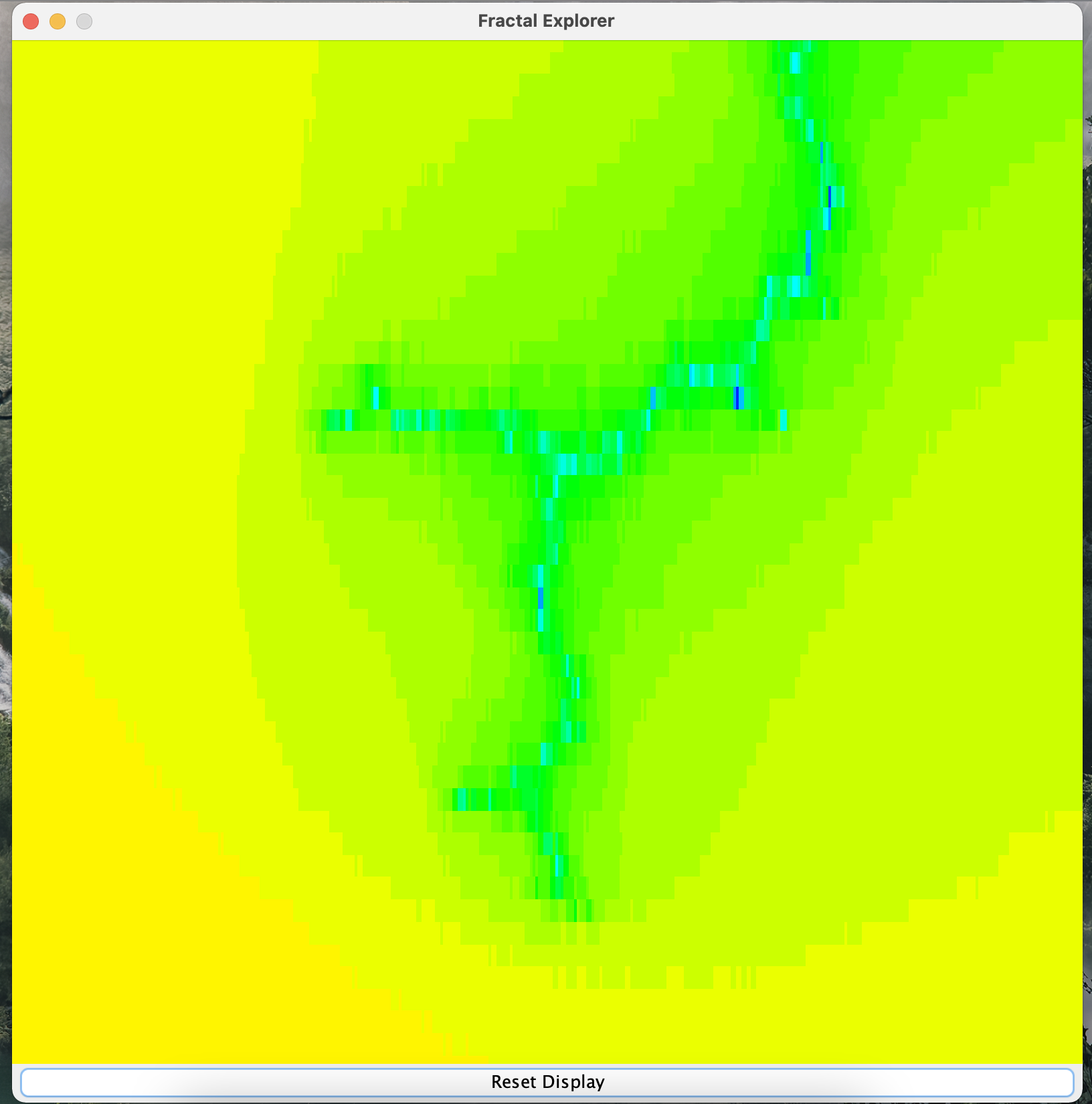


Рисунок 2 - Максимальное приближение (предел double)