**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

Лабораторная работа № 5

Выполнил: студент группы БФИ2002

Ковачев В.Е.

Проверила:

Москва, 2021

**Задание:**

В данной лабораторной работе генератор фракталов будет расширен двумя новыми функциями. Во-первых, вы добавите поддержку нескольких фракталов и реализуете возможность выбирать нужный фрактал из выпадающего списка. Во-вторых, вы добавите поддержку сохранения текущего изображения в файл.

Поддержка нескольких фракталов

Так как в реализацию была введена абстракция FractalGenerator, добавление нескольких фракталов не будет проблемой. В данной лабораторной работе вы добавите поддержку нескольких фракталов, и пользователь сможет выбирать между ними, используя combo-box. Программный интерфейс Swing (Swing API) предоставляет combo-box через класс javax.swing.JComboBox, а также запускает ActionEvents при выборе новогоэлемента. Необходимо сделать:

• Создать 2 новые реализации FractalGenerator Первым будет фрактал tricorn, который должен находиться в файле Tricorn.java. Для этого нужно создать подкласс FractalGenerator и реализация будет почти идентична фракталу Мандельброта, кроме двух изменений. Вы даже можете скопировать исходный код фрактала Мандельберта и просто внести следующие изменения:

¬ Уравнение имеет вид zn = zn-1 2 + c. Единственное отличие только в том, что используется комплексное сопряжение zn-1 на каждой итерации.

¬ Начальный диапазон для трехцветного фрактала должен быть от (- 2, -2) до (2, 2). Второй фрактал, который необходимо реализовать - это фрактал «Burning Ship», который в реальности не похож на пылающий корабль. Данный фрактал имеет следующие свойства:

¬ Уравнение имеет вид zn = (|Re(zn-1)| + i |Im(zn-1)|)2 + c. Другими словами, вы берете абсолютное значение каждого компонента zn-1 на каждой итерации.

¬ Начальный диапазон для данного фрактала должен быть от (-2, -2.5) до (2, 1.5).

• Сombo-boxe в Swing может управлять коллекцией объектов, но объекты должны предоставлять метод toString(). Убедитесь, что в каждой реализации фракталов tcnm метод toString(), который возвращает имя, например «Mandelbrot», «Tricorn» и «Burning Ship».

• Настроить JComboBox в вашем пользовательском интерфейсе можно с использованием конструктора без параметров, а затем использовать метод addItem(Object) для того, чтобы добавить реализации вашего генератора фракталов. Как указывалось в предыдущем шаге, выпадающий список будет использовать метод toString () в ваших реализациях для отображения генераторов в выпадающем списке.

Необходимо будет также добавить объект label в разрабатываемый пользовательский интерфейс перед выпадающим списком, в качестве пояснения к выпадающему списку. Это можно сделать, создав новый объект Jpanel и добавив в него объекты JLabel и JcomboBox, а затем разместить панель на позиции NORTH на вашем макете окна.

И наконец, необходимо добавить поддержку выпадающего списка в реализацию ActionListener. В случае, если событие поступило от выпадающего списка, вы можете извлечь выбранный элемент из виджета и установить его в качестве текущего генератора фракталов. (Используйте метод getSelectedItem()) При этом не забудьте сбросить начальный диапазон и перерисовать фрактал!

• Во-первых, вам нужно добавить кнопку «Save Image» в ваше окно. Для этого вы можете добавить обе кнопки «Save Image» и «Reset» в новую Jpanel, а затем разместить эту панель в SOUTH части окна.

События от кнопки «Save Image» также должны обрабатываться реализацией ActionListener. Назначьте кнопкам «Save Image» и «Reset» свои значения команд (например, «save» и «reset») для того, чтобы обработчик событий мог отличить события от этих двух разных кнопок.

• В обработчике кнопки «Save Image» вам необходимо реализовать возможность указания пользователем, в какой файл он будет сохранять изображение. Это можно сделать с помощью класса javax.swing.JFileChooser. Указанный класс предоставляет метод showSaveDialog(), который открывает диалоговое окно «Save file», позволяя тем самым пользователю выбрать директорию для сохранения. Метод принимает графический компонент, который является родительским элементом для диалогового окна с выбором файла, что позволяет центрированию окна с выбором относительно его родителя. В качестве родителя используйте окно приложения.

• Также необходимо настроить средство выбора файлов, чтобы сохранять изображения только в формате PNG, на данном этапе вы будете работать только с данным форматом. вы сможете это настроить с помощью javax.swing.filechooser.FileNameExtensionFilter, как это продемонстрировано ниже: JFileChooser chooser = new JfileChooser();

FileFilter filter = new FileNameExtensionFilter("PNG Images", "png");

chooser.setFileFilter(filter);

chooser.setAcceptAllFileFilterUsed(false);

Последняя строка гарантирует, что средство выбора не разрешит пользователю использование отличных от png форматов.

• Если пользователь успешно выбрал файл, следующим шагом является сохранения изображения фрактала на диск! Для данного рода задач Java включает в себя необходимую функциональность. Класс javax.imageio.ImageIO обеспечивает простые операции загрузки и сохранения изображения. Вы можете использовать метод write(RenderedImage im, String formatName, File output). Параметр formatName будет содержать значение «png”. Тип «RenderedImage» - это просто экземпляр BufferedImage из вашего компонента JimageDisplay. (Используйте для него тип доступа public)

Метод write() может вызвать исключение, поэтому вам необходимо заключить этот вызов в блок try/catch и обработать возможную ошибку. Блок catch должен проинформировать пользователя об ошибке через диалоговое окно. Swing предоставляет класс javax.swing.JoptionPane для того, чтобы упростить процесс создания информационных диалоговых окон или окон, где нужно выбрать да/нет. Для этого вы можете использовать статический метод JoptionPane.showMessageDialog(Component parent, Object message, String title, int messageType), где messageType у вас будет JOptionPane.ERROR\_MESSAGE. В сообщении об ошибке вы можете использовать возвращаемое значение метода getMessage(), а заголовком окна может быть, например, «Cannot Save Image». Родительским компонентом будет окно для того, чтобы диалоговое окно с сообщением об ошибке выводилось относительно центра окна.

**Ход работы:**

Для выполнения первой части задания было необходимо написать два кода для создания новых фракталов.

**Фрактал Tricorn:**

import java.awt.geom.Rectangle2D;

public class Tricorn extends FractalGenerator{

public static final int MAX\_ITERATIONS = 2000;

@Override

public void getInitialRange(Rectangle2D.Double range) {

range.x = -2;

range.y = -2;

range.width = 4;

range.height = 4;

}

@Override

public int numIterations(double x, double y) {

double r = x;

double i = y;

int counter = 0;

while (counter < MAX\_ITERATIONS) {

counter++;

double k = r \* r - i \* i + x;

double m = (-2) \* r \* i + y;

r = k;

i = m;

if (r \* r + i \* i > 4)

break;

}

if (counter == MAX\_ITERATIONS)

return -1;

return counter;

}

@Override

public String toString() {

return "Tricorn";

}

}

**Фрактал BurningShip:**

import java.awt.geom.Rectangle2D;

public class BurningShip extends FractalGenerator {

public static final int MAX\_ITERATIONS = 2000;

@Override

public void getInitialRange (Rectangle2D.Double range){

range.x = -2;

range.y = -2.5;

range.height = 4;

range.width = 4;

}

@Override

public int numIterations(double x, double y) {

double r = x;

double i = y;

int counter = 0;

while ((counter < MAX\_ITERATIONS)) {

counter++;

double k = r \* r - i \* i + x;

double m = Math.abs(2 \* r \* i) + y;

r = k;

i = m;

if ((r \* r + i \* i) > 4)

break;

}

if (counter == MAX\_ITERATIONS)

return -1;

return counter;

}

@Override

public String toString() {

return "Burning Ship";

}

}

**Для выполнения второй части задания был расширен уже существующий класс FractalExplorer:**

import java.awt.\*;

import javax.imageio.ImageIO;

import javax.swing.\*;

import javax.swing.border.Border;

import javax.swing.filechooser.FileNameExtensionFilter;

import java.awt.event.\*;

import java.awt.geom.Rectangle2D;

import java.io.File;

import java.io.FileFilter;

import java.io.IOException;

public class FractalExplorer {

private int displaySize;

private JImageDisplay imageDisplay;

private FractalGenerator fractalGenerator;

private Rectangle2D.Double range;

private JComboBox comboBox;

private FractalExplorer (int displaySize) {

this.displaySize = displaySize;

this.fractalGenerator = new Mandelbrot();

this.range = new Rectangle2D.Double(0,0,0,0);

fractalGenerator.getInitialRange(this.range);

}

**// точка входа**

public static void main(String[] args) {

FractalExplorer fractalExplorer = new FractalExplorer(600);

fractalExplorer.setGUI();

fractalExplorer.drawFractal();

}

**// задание интерфейса**

public void setGUI() {

JFrame frame = new JFrame("Fractal Generator");

JButton buttonReset = new JButton("Reset");

JButton buttonSave = new JButton("Save image");

JPanel jPanel\_1 = new JPanel();

JPanel jPanel\_2 = new JPanel();

JLabel label = new JLabel("Fractal:");

imageDisplay = new JImageDisplay(displaySize, displaySize);

imageDisplay.addMouseListener(new MouseListener());

**// выпадающий список**

comboBox = new JComboBox();

comboBox.addItem(new Mandelbrot());

comboBox.addItem(new Tricorn());

comboBox.addItem(new BurningShip());

comboBox.addActionListener(new ActionHandler());

**// кнопка reset**

buttonReset.setActionCommand("Reset");

buttonReset.addActionListener(new ActionHandler());

**// кнопка сохранить**

buttonSave.setActionCommand("Save");

buttonSave.addActionListener(new ActionHandler());

jPanel\_1.add(label, BorderLayout.CENTER);

jPanel\_1.add(comboBox, BorderLayout.CENTER);

jPanel\_2.add(buttonReset, BorderLayout.CENTER);

jPanel\_2.add(buttonSave, BorderLayout.CENTER);

frame.setLayout(new BorderLayout());

frame.add(imageDisplay, BorderLayout.CENTER);

frame.add(jPanel\_1, BorderLayout.NORTH);

frame.add(jPanel\_2, BorderLayout.SOUTH);

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

frame.pack();

frame.setVisible(true);

frame.setResizable(false);

}

**// отрисовка фрактала в JImageDisplay**

private void drawFractal() {

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

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

int counter = fractalGenerator.numIterations(FractalGenerator.getCoord(range.x, range.x + range.width, displaySize, x),

fractalGenerator.getCoord(range.y, range.y + range.width, displaySize, y));

if (counter == -1) {

imageDisplay.drawPixel(x, y, 0);

}

else {

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

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

imageDisplay.drawPixel(x, y, rgbColor);

}

}

}

imageDisplay.repaint();

}

**// обработчик кнопок**

public class ActionHandler implements ActionListener {

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equals("Reset")) {

**// перерисовка фрактала**

fractalGenerator.getInitialRange(range);

drawFractal();

} else if (e.getActionCommand().equals("Save")) {

**// сохранение фракталов**

JFileChooser fileChooser = new JFileChooser();

FileNameExtensionFilter fileFilter = new FileNameExtensionFilter("PNG Images", "png");

fileChooser.setFileFilter(fileFilter);

fileChooser.setAcceptAllFileFilterUsed(false);

int t = fileChooser.showSaveDialog(imageDisplay);

if (t == JFileChooser.APPROVE\_OPTION) {

try {

ImageIO.write(imageDisplay.getImage(), "png", fileChooser.getSelectedFile());

} catch (NullPointerException | IOException ee) {

JOptionPane.showMessageDialog(imageDisplay, ee.getMessage(), "Cannot save image", JOptionPane.ERROR\_MESSAGE);

}

}

} else {

fractalGenerator = (FractalGenerator) comboBox.getSelectedItem();

range = new Rectangle2D.Double(0,0,0,0);

fractalGenerator.getInitialRange(range);

drawFractal();

}

}

}

public class MouseListener extends MouseAdapter {

@Override

public void mouseClicked(MouseEvent e) {

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

double y = FractalGenerator.getCoord(range.y, range.y + range.width, displaySize, e.getY());

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

drawFractal();

}

}

}

**Результат работы программы**

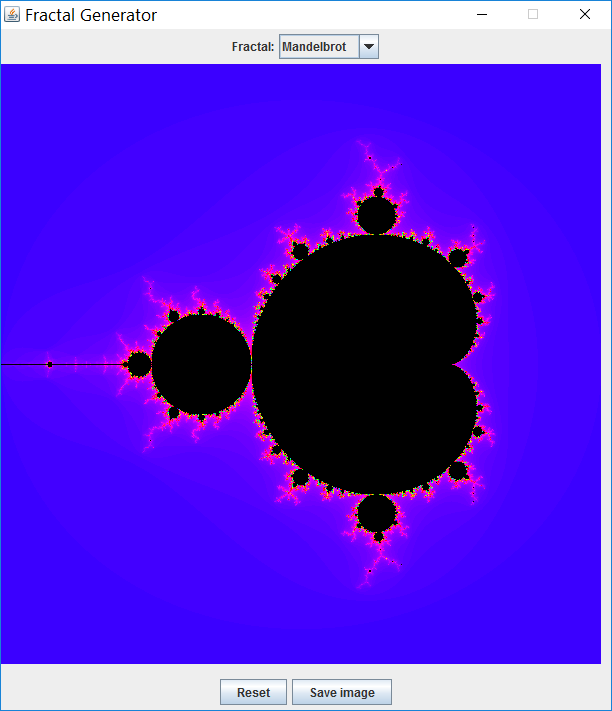


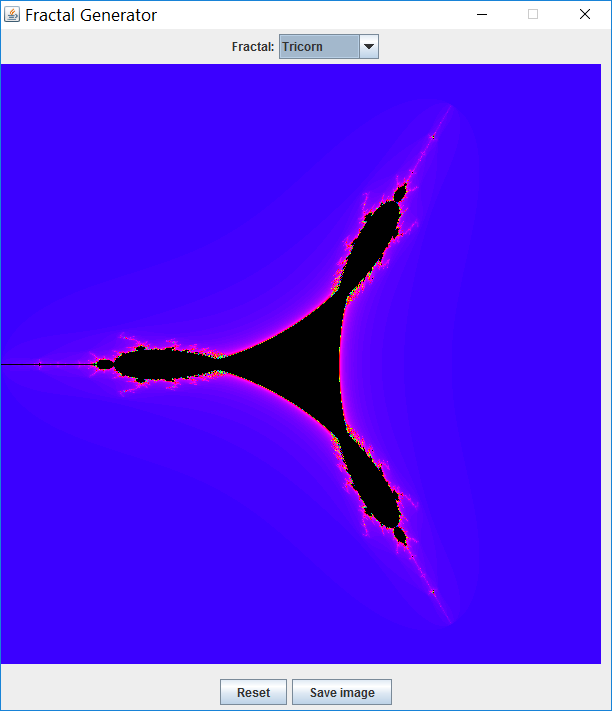
Рисунок №1- Фрактал Mandellbrot

Рисунок №2- Фрактал Tricorn

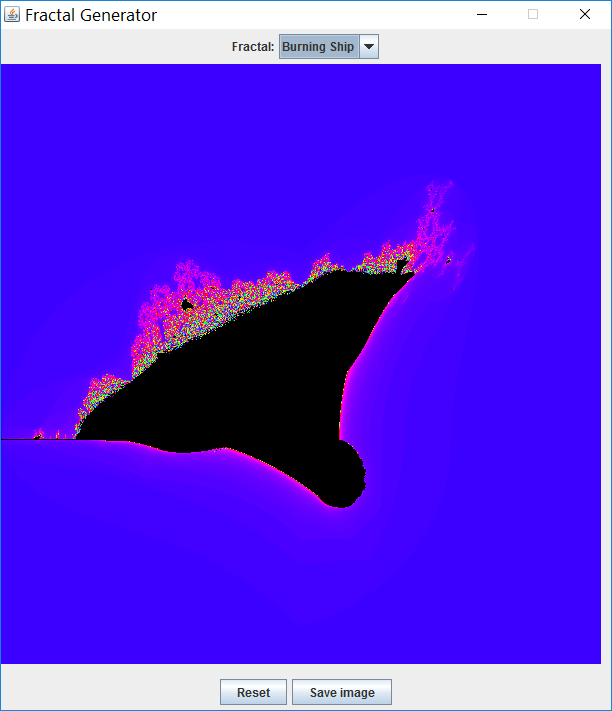


Рисунок №3- Фрактал Burning Ship

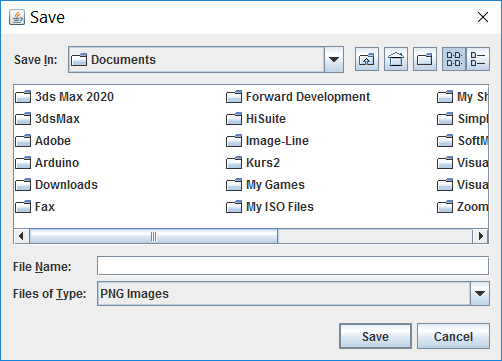


Рисунок №3- Диалоговое окно для сохранения фрактала

**Вывод**

В результат е выполнения заданий было мы научились использовать диалоговые окна на языке java, а также выпадающие списки. Была реализована возможность переключения между фракталами.

**Список использованной литературы**

1. Камаев В.А., Костерин В.В. Технологии программирования. М.: Высшая школа,

2006.

2. Жоголев Е.А.Технология программирования. – М.: Научный мир, 2004.