ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ

Ордена Трудового Красного Знамени

федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский Технический Университет Связи и Информатики» (МТУСИ)

Кафедра МКиИТ

Лабораторная работа по технологиям программирования №4

«Фрактал Эксплорер»

Выполнил:

Студент 2 курса

Группы БСТ-1602

Пащенко Алексей

Вариант №12

Москва, 2018

Оглавление

[1. Цель и задачи. 3](#_Toc514605232)

[2. Анализ предметной области и выбор инструментария. 3](#_Toc514605233)

[3. Код программы и объяснение функций. 3](#_Toc514605234)

[3.1. Файл FractalExplorer.java 3](#_Toc514605235)

[3.2. Файл FractalGenerator.java 5](#_Toc514605236)

[3.3. Файл JImageDisplay.java 6](#_Toc514605237)

[3.4. Файл Mandelbrot.java 8](#_Toc514605238)

[3.5. Примеры работы программы. 9](#_Toc514605239)

[4. Заключение. 10](#_Toc514605240)

1. Цель и задачи.

Написать программу, рисующую фрактал Мандельброта.

1. Анализ предметной области и выбор инструментария.

В текущей работе я использовал бесплатно распространяемый пакет разработчика Java Development Kit. Однако, этот пакет не предусматривает интегрированную среду разработки, поэтому для написания непосредственно кода мною была выбрана программа Visual Studio Code, так как она обеспечивает подсветку интерфейса и автоподстановку кода.

1. Код программы и объяснение функций.
   1. Файл FractalExplorer.java

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.awt.geom.Rectangle2D;

public class FractalExplorer{

private JImageDisplay display;

private int dSize;

private FractalGenerator fGenerator;

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

public static void main(String[] args){

FractalExplorer expl = new FractalExplorer(800);

expl.createAndShowGUI();

expl.drawFractal();

}

/\*\*

\* @param size - The height and width of the window

\*/

public FractalExplorer(int size){

dSize = size;

display = new JImageDisplay(dSize, dSize);

fGenerator = new Mandelbrot();

fGenerator.getInitialRange(range);

}

public void createAndShowGUI(){

// Create the Window

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

//Add fractal display

display.addMouseListener(new mouse\_listener());

frame.add(display,BorderLayout.CENTER);

//Add the button

JButton button = new JButton("Reset");

button.addActionListener(new act\_listener());

frame.add(button, BorderLayout.SOUTH);

// Set the default close action to exit

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

//The installation default settings

frame.pack();

frame.setVisible(true);

frame.setResizable(false);

}

public void drawFractal(){

for(int i=0;i<dSize;i++)

for (int j=0;j<dSize;j++){

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

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

// Compute the number of iterations

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

//If the color is not default

if(iterations!=-1){

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

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

display.drawPixel(i, j, color);

}

// If color is default

else display.drawPixel(i, j, 0);

}

// Display the image on the screen

display.repaint();

}

private class act\_listener implements ActionListener {

public void actionPerformed(ActionEvent e)

{

fGenerator.getInitialRange(range);

display.clearImage();

drawFractal();

}

}

private class mouse\_listener extends MouseAdapter {

public void mouseClicked(MouseEvent e) {

// Get mouse's coordinates

super.mouseClicked(e);

int mouseX=e.getX();

int mouseY=e.getY();

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

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

fGenerator.recenterAndZoomRange(range, xCoord, yCoord, 0.5);

display.clearImage();

drawFractal();

}

}

}

Данный класс отвечает за общую работу программы и взаимодействие между другими классами. Старт программы происходит именно с этой функции.

* 1. Файл FractalGenerator.java

import java.awt.geom.Rectangle2D;

/\*\*

\* This class provides the common interface and operations for fractal

\* generators that can be viewed in the Fractal Explorer.

\*/

public abstract class FractalGenerator {

/\*\*

\* This static helper function takes an integer coordinate and converts it

\* into a double-precision value corresponding to a specific range. It is

\* used to convert pixel coordinates into double-precision values for

\* computing fractals, etc.

\*

\* @param rangeMin the minimum value of the floating-point range

\* @param rangeMax the maximum value of the floating-point range

\*

\* @param size the size of the dimension that the pixel coordinate is from.

\* For example, this might be the image width, or the image height.

\*

\* @param coord the coordinate to compute the double-precision value for.

\* The coordinate should fall in the range [0, size].

\*/

public static double getCoord(double rangeMin, double rangeMax,

int size, int coord) {

assert size > 0;

assert coord >= 0 && coord < size;

double range = rangeMax - rangeMin;

return rangeMin + (range \* (double) coord / (double) size);

}

/\*\*

\* Sets the specified rectangle to contain the initial range suitable for

\* the fractal being generated.

\*/

public abstract void getInitialRange(Rectangle2D.Double range);

/\*\*

\* Updates the current range to be centered at the specified coordinates,

\* and to be zoomed in or out by the specified scaling factor.

\*/

public void recenterAndZoomRange(Rectangle2D.Double range,

double centerX, double centerY, double scale) {

double newWidth = range.width \* scale;

double newHeight = range.height \* scale;

range.x = centerX - newWidth / 2;

range.y = centerY - newHeight / 2;

range.width = newWidth;

range.height = newHeight;

}

/\*\*

\* Given a coordinate <em>x</em> + <em>iy</em> in the complex plane,

\* computes and returns the number of iterations before the fractal

\* function escapes the bounding area for that point. A point that

\* doesn't escape before the iteration limit is reached is indicated

\* with a result of -1.

\*/

public abstract int numIterations(double x, double y);

}

Этот класс отвечает за генерацию фракталов. Здесь определены функции изменения центра фрактала и получения координат.

* 1. Файл JImageDisplay.java

import javax.swing.JComponent;

import java.awt.image.BufferedImage;

import java.awt.Dimension;

import java.awt.Graphics;

public class JImageDisplay extends JComponent{

private BufferedImage im;

/\*\*

\* @param width - Picture width

\* @param height - Picture height

\*/

JImageDisplay(int width, int height){

im = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_RGB);

super.setPreferredSize(new Dimension(width, height));

}

/\*\*

\* @param g - Displaying an image on the screen

\*/

protected void paintComponent(Graphics g){

super.paintComponent(g);

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

}

// Image cleanup

public void clearImage(){

/\*int arrLength;

if(im.getHeight()>im.getWidth()) arrLength=im.getHeight();

else arrLength=im.getWidth();

int[] rgbArray = new int[arrLength];\*/

int[] rgbArray = new int[im.getWidth()];

im.setRGB(0, 0, im.getWidth()-1, im.getHeight()-1, rgbArray, 0, 0);

}

// Setting the color to pixel

/\*\*

\* @param x - The abscissa of the point

\* @param y - The ordinate of the point

\* @param rgbColor - The color of the point

\*/

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

im.setRGB(x, y, rgbColor);

}

}

Этот класс наследует класс JComponent и содержит функции отрисовки компонента, очистки изображения и закрашивания пикселя.

* 1. Файл Mandelbrot.java

import java.awt.geom.Rectangle2D;

public class Mandelbrot extends FractalGenerator{

// Maximum iterations:

public static final int MAX\_ITERATIONS = 2000;

/\*\*

\* @param range - The range borders of the fractal

\*/

public void getInitialRange (Rectangle2D.Double range){

range.x=-2;

range.y=-1.5;

range.width=range.height=3;

}

//Use formulas for fractal Mandelbrot

/\*\*

\* @param x - The abscissa of the point

\* @param y - The ordinate of the point

\*/

public int numIterations(double x, double y){

double re=0;

double im=0;

for (int i=0; i<MAX\_ITERATIONS; i++){

double re1=re\*re-im\*im+x;

double im1=2\*re\*im+y;

//If we have reached the Mandelbrot condition, we will return the number of iterations

if ((im\*im+re\*re) > 4)

return i;

//The change the real and imaginary parts for the next iteration

re=re1;

im=im1;

}

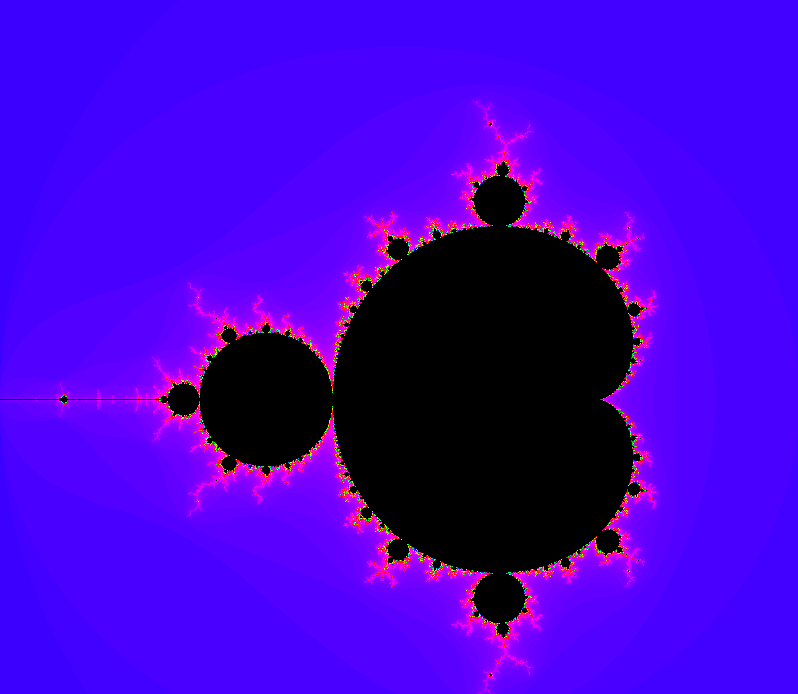
return -1;

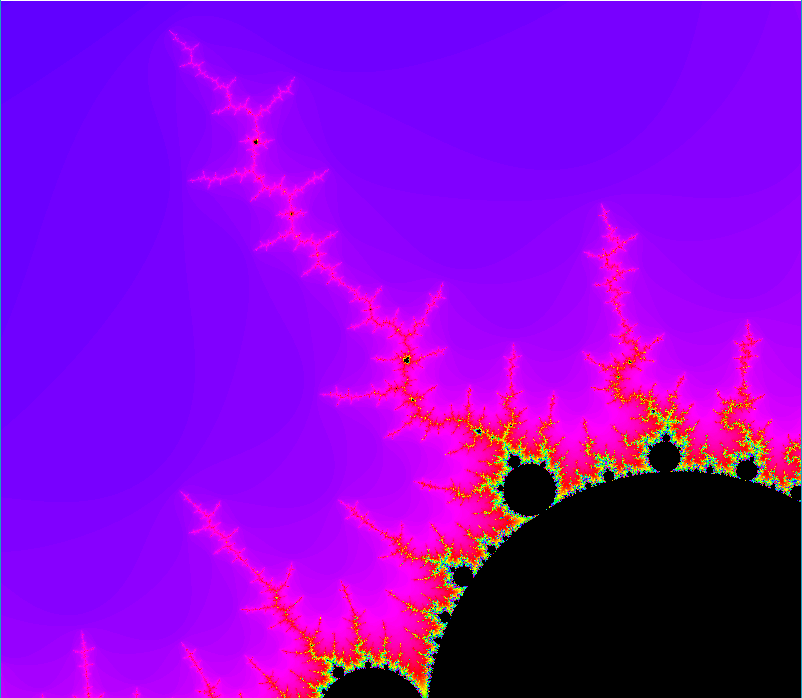
}

}

Этот класс отвечает за расчёт фрактала Мандельброта и позволяет найти количество итераций для расчёта фрактала Мандельброта. Он наследует класс FractalGenerator.

* 1. Примеры работы программы.





1. Заключение.

В ходе работы была написана программа, позволяющая представлять графически фрактал Мандельброта, используя при этом наследование классов.