**Федеральное государственное общеобразовательное учреждение высшего образования**

**Московский государственный технический университет**

**им. Н. Э. Баумана**

Факультет "Информатика и системы управления"

Кафедра "Информационные системы и телекоммуникации"

**Отчет по лабораторной работе №5**

**"Выбор и сохранение фракталов"**

Студент: Задубин А.А.

Группа: ИУ3-42Б

Преподаватель: Руденкова Ю.С.

**Задание:**

Необходимо было добавить: во первых, возможность отображения нескольких фракталов и выбор фрактала с помощью выпадающего списка. Во вторых, возможность сохранить текущее изображение фрактала в файл.

Создали новые реализации FractalGenerator. Первая – фрактал треуголка. Сохранили ее в файле Tricorn.java. Сделали класс наследник FractalGenerator, и реализация его была почти такая же, как реализация Мандельброта, за исключением двух отличий. Фактически, мы могли просто скопировать исходный код фрактала Мандельброта, и внести следующие изменения. Уравнение теперь такое zn = zn-1^2 + c. Разница только в том, что на каждой итерации вычисляется комплексно сопряженное значение zn-1.Начальный диапазон фрактала треуголка должен был быть от (-2, -2) до (2, 2). Второй фрактал, который надо было сделать называется "Горящий Корабль", который в действительности выглядит как горящий корабль. Вот детали его реализации: Уравнение для вычисления фрактала zn = (|Re(zn-1)| + i |Im(zn-1)|) ^2 + c. Начальный диапазон фрактала от (-2, -2.5) до (2, 1.5).

Использовали конструктор без аргументов, и метод addItem(Object) для добавления каждой реализации абстрактного класса FractalGenerator. Выпадающий список использует метод toString() объектов генераторов фракталов для отображения их имен.

Добавили к пользовательскому интерфейсу, перед выпадающим списком, текстовое поле поясняющим роль выпадающего списка. Для этого создали новый объект JPanel, добавили в него объекты JLabel и JComboBox, и затем поместили панель в позицию NORTH содержимого JFrame. Наконец, добавили обработку событий выпадающего списка к реализации ActionListener.

Далее добавили на экран программы кнопку "Сохранить изображение". Обе кнопки Сохранить и Сброс можно было поместить в новый объект JPanel, и затем добавить его в область SOUTH объекта JFrame программы, аналогично тому как раньше мы поступили с текстовым полем и выпадающим списком.

Для кнопки Сохранить также надо было предусмотреть обработку событий в ActionListener. Кнопки Сохранить и Сброс должны были иметь собственные значения команд, для того чтобы обработчик мог их различить. В обработчике кнопки "Сохранить ", надо было добавить выбор имени файла в котором будет сохранено изображение. Это можно было очень просто сделать с помощью класса javax.swing.JFileChooser. Этот класс имеет метод showSaveDialog(), который вызывает диалоговое окно "Сохранить файл " в котором пользователь может указать имя сохраняемого файла. Методу нужно было передать ссылку на объект Component который будет родительским окном диалога выбора файла; это нужно для того чтобы диалог центрировался относительно родительского окна. Подставили в этот параметр ссылку на объект JFrame. Метод возвращает значение типа int, которое определяет результат операции выбора файла. Если метод возвращает константу JFileChooser.APPROVE\_OPTION, можно будет продолжить операцию записи файла; иначе, пользователь отменил запрос, и надо просто вернуть управление.

**Код программы:**

BurningShip.java

**public** **class** **BurningShip** **extends** FractalGenerator {

**public** **static** **final** **int** MAX\_ITERATIONS = **500**;

**@Override**

**public** **void** **getInitialRange**(Double range) {}

**@Override**

**public** **int** **numIterations**(**double** x, **double** y) {}

**@Override**

**public** String **toString**() {}

}

FractalExplorer.java

**public** **class** **FractalExplorer** {

**private** **int** side;

**private** JImageDisplay jpgImg;

**private** FractalGenerator geni = **new** BurningShip();

**private** Rectangle2D.Double vsblRange;

**public** **FractalExplorer**(**int** x) {}

**class** **ActiveLisner** **implements** ActionListener{

**@Override**

**public** **void** **actionPerformed**(ActionEvent e) {}

}

**class** **MouseClass** **extends** MouseAdapter{

**public** **void** **mouseClicked**(MouseEvent e) {}

}

**private** **void** **drawFractal**() {}

**private** **void** **createAndShowGUI**() {}

**public** **static** **void** **main**(String[] args) {}

}

FractalGenerator.java

**public** **abstract** **class** **FractalGenerator** {

**public** **static** **double** **getCoord**(**double** rangeMin, **double** rangeMax,

**int** size, **int** coord) {}

**public** **abstract** **void** **getInitialRange**(Rectangle2D.Double range);

**public** **void** **recenterAndZoomRange**(Rectangle2D.Double range,

**double** centerX, **double** centerY, **double** scale) {}

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

}

JImageDisplay.java

**public** **class** **JImageDisplay** **extends** JComponent {

**private** BufferedImage img;

**public** **JImageDisplay**(**int** width, **int** height) {}

**public** BufferedImage **getImg**() {}

**@Override**

**protected** **void** **paintComponent**(Graphics g) {}

**public** **void** **clearImage**() {}

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

}

Mandelbrot.java

**public** **class** **Mandelbrot** **extends** FractalGenerator {

**@Override**

**public** **void** **getInitialRange**(Double range) {}

**@Override**

**public** **int** **numIterations**(**double** x, **double** y) {}

**@Override**

**public** String **toString**() {}

}

Tricorn.java

**public** **class** **Tricorn** **extends** FractalGenerator {

**public** **static** **final** **int** MAX\_ITERATIONS = **1000**;

**@Override**

**public** **void** **getInitialRange**(Double range) {}

**@Override**

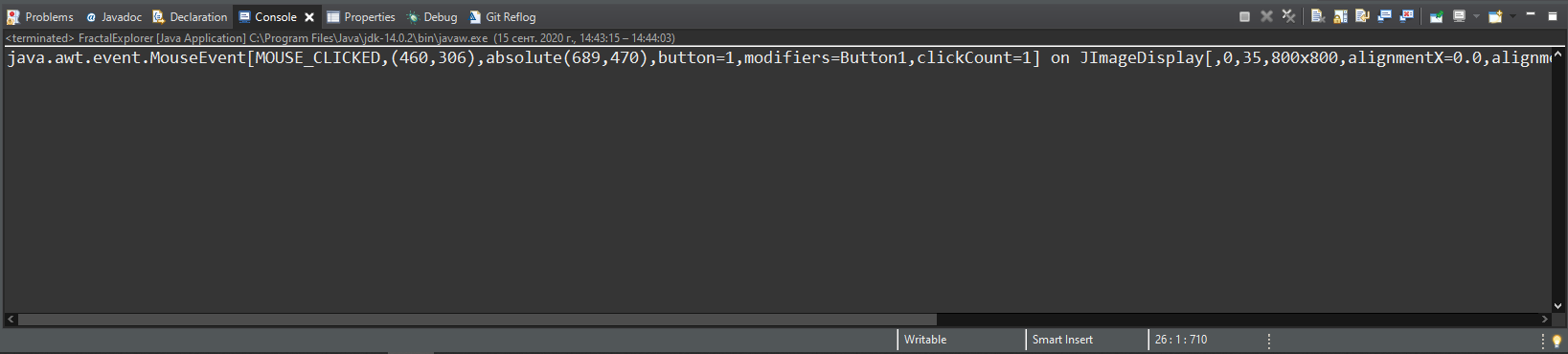
**public** **int** **numIterations**(**double** x, **double** y) {}

**@Override**

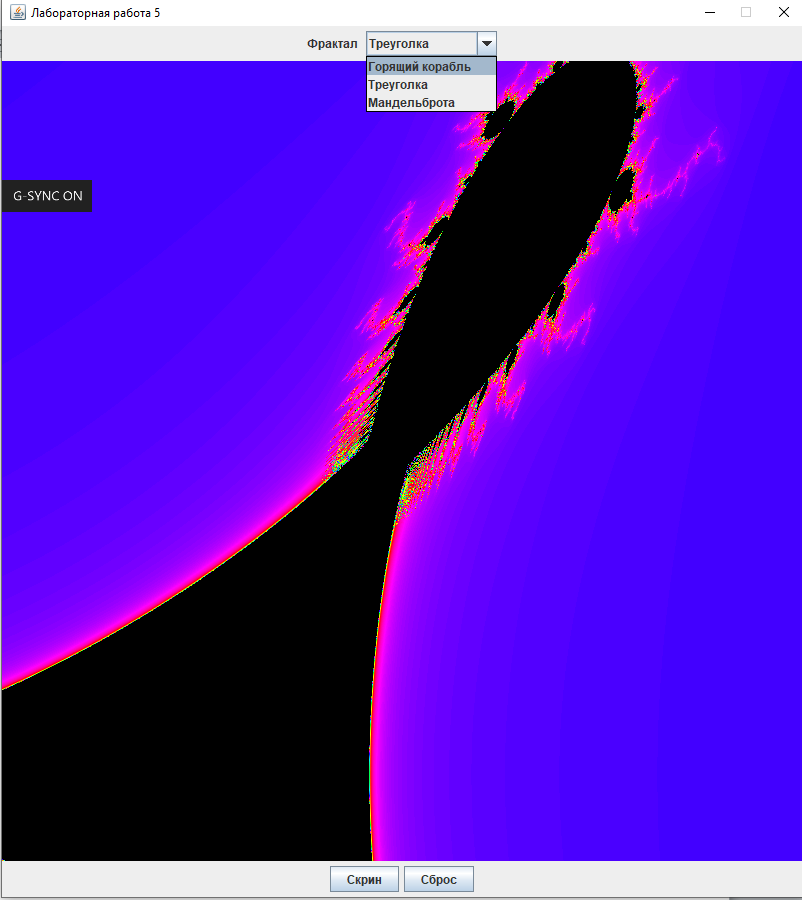
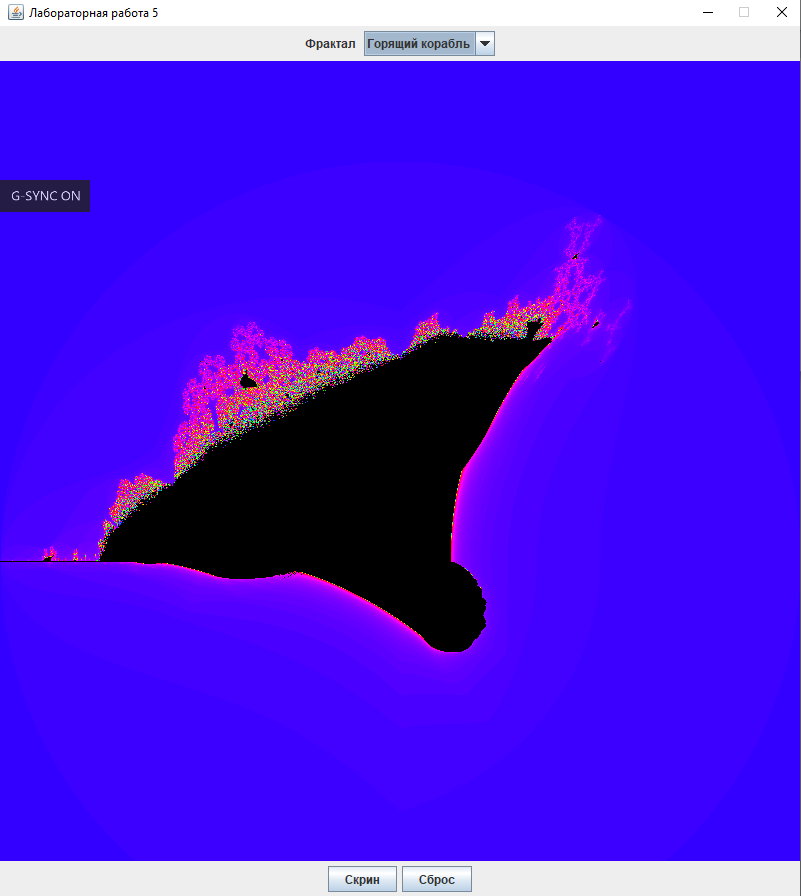
**public** String **toString**() {}

}

**Вывод в консоль:**

****

**Результат работы программы:**



**Вывод**

В данной лабораторной работе мы изучили разные типы фракталов. Программа была модернизирована. Были добавлены возможности выбора любого из трёх фракталов и сохранения скриншотов на диск.