Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Новосибирский государственный технический университет»

Кафедра вычислительной техники

Лабораторная работа № 2

По дисциплине «Технология программирования»

На тему: «Разработка графического интерфейса приложения»

Факультет: АВТФ

Группа: АВТ-808

Студент: Хайрулина Елизавета

Вариант 12

Преподаватель: Михайленко Дмитрий Анатольевич

Новосибирск 2020

Практические задания

1. Познакомиться с основными графическими библиотеками Java - AWT и Swing и их основными компонентами. Изучить классы менеджеров компоновки, классы создания меню приложения, основных и диалоговых окон и т.д.

2. Для хранения генерируемых объектов использовать динамический массив объектов. Для обеспечения доступа всем элементам приложения и создания только одного экземпляра массива объектов используйте структурный паттерн Singleton.

3. При реализации шаблона Singleton используется класс Singleton. В этом классе определяется закрытый конструктор, имеется закрытая статическая переменная, в которой хранится ссылка на единственный экземпляр данного класса, а также определен статический метод доступа, возвращающий ссылку на этот экземпляр. Так же класс содержит объект коллекции и открытые методы доступа к ней. Таким образом можно в любом месте приложения получить ссылку на объект класса Singleton и доступ к коллекции объектов.

4. Доработать программу, созданную в лабораторной работе № 1:

1) поделить рабочую область окна приложения на 2 части. Визуализация переносится в одну часть окна, панель управления в другую;

2) добавить кнопки «Старт» и «Стоп» в панель управления. Они должны запускать и останавливать симуляцию соответственно. Если симуляция остановлена, то кнопка «Стоп» должна блокироваться. Если симуляция идет, то блокируется кнопка «Старт». Клавиши B и E должны функционировать по-прежнему;

3) добавить переключатель «Показывать информацию», который разрешает отображение модального диалога из 7 пункта задания;

4) добавить группу из 2 исключающих переключателей: «Показывать время симуляции» и «Скрывать время симуляции». Клавиша T должна функционировать по-прежнему;

5) используя различные менеджеры компоновки, реализовать интерфейс пользователя согласно индивидуальному заданию;

6) добавить в программу главное в меню и панель инструментов, в которых продублировать основные команды вашего интерфейса пользователя;

7) при остановке симуляции должно появляться модальное диалоговое окно (при условии, что оно разрешено) с информацией о количестве и типе сгенерированных объектов, а также времени симуляции. Вся информация выводится в элементе TextArea, недоступном для редактирования. В диалоговом окне должно быть 2 кнопки: «ОК» и «Отмена». При нажатии на «ОК» симуляции останавливается, а при нажатии на «Отмена», соответственно продолжается;

8) предусмотреть проверку данных вводимых пользователем. При вводе неверного значения обрабатывать исключительную ситуацию: выставлять значение по умолчанию и выводить диалоговое окно с сообщением об ошибке;

9) Реализовать следующие элементы управления:

- Периоды рождения объектов – текстовые поля;

- Для задания вероятностей рождения объектов комбобокс и список (шаг значений 10%);

- Дополнить интерфейс поясняющими метками.

Описание структуры программы

Классы и интерфейсы, описанные в ЛР № 1:

*interface IBehaviour; public abstract class Bird implements IBehaviour; public class Big extends Bird; public class Small extends Bird; public interface AbstractFactory; public class ConcreteFactory* implem*ents AbstractFactory; public class Habitat extends JFrame; public class ToBeContinued extends TimerTask; public class Depict\_a\_bird extends JPanel; public class AnimalTour extends KeyAdapter.*

public class *Singleton*

Структурный паттерн Singleton: для обеспечения доступа всем элементам приложения и создания только одного экземпляра массива объектов. При реализации шаблона Singleton используется класс Singleton. В этом классе определяется закрытый конструктор (*private Singleton()*), имеется закрытая статическая переменная (*private static Singleton M*), в которой хранится ссылка на единственный экземпляр данного класса, а также определен статический метод доступа, возвращающий ссылку на этот экземпляр (*public static Singleton getM()*). Так же класс содержит объект коллекции и открытые методы доступа к ней. Таким образом можно в любом месте приложения получить ссылку на объект класса Singleton и доступ к коллекции объектов (*public final List <Bird> Bird\_s*).

public class *menu* extends *JPanel*

Служит для сборки следующих элементов управления на панели:

*/////////*

*checkB = new CheckB(); // Показывать информацию*

*add(checkB);*

*/////////*

*periods = new Periods(); // Периоды появления птиц*

*add(periods);*

*/////////*

*clock = new Clock(habitat); // Показывать/скрывать таймер*

*add(clock);*

*////////*

*combob = new ComboB(); // Вероятности*

*add(combob);*

*////////*

Так же содержит методы, возвращающие необходимые элементы и служащие связующими звеньями между *Habitat* или *ConcreteFactory* и элементами управления, находящимися в пакете *Contr*.

package *Contr*

Содержит все элементы управления, информационные окна, … , - основные составляющие интерфейса пользователя.

public class *Butt* extends *JPanel*

Кнопки «Старт» и «Стоп», которые запускают и останавливают симуляцию соответственно. К данным кнопкам присоединяются слушатели, в которых реализуется тот или иной метод из класса *StarT\_StoP*.

public class *StarT\_StoP*

Здесь запускается/останавливается симуляция. Если симуляция остановлена, то кнопка «Стоп» блокируется. Если симуляция идет, то блокируется кнопка «Старт». Клавиши *B* и *E* функционируют по-прежнему, как описано в ЛР № 1.

public class *CheckB* extends *JPanel*

Переключатель «Показывать информацию», который разрешает отображение модального диалога *Inf*  (см. далее).

public class *Clock* extends *JPanel*

Группа из 2 исключающих переключателей: «Показывать время симуляции» и «Скрывать время симуляции». Регулирует отображение строки *Line* . Клавиша *T* функционирует по-прежнему, как описано в ЛР № 1.

public class *ComboB* extends *JPanel*

Комбобокс и список (шаг значений 10%) - для задания вероятностей рождения объектов (взрослых птиц и птенцов).

public class *Inf* extends *JDialog*

При остановке симуляции появляется модальное диалоговое окно (при условии, что оно разрешено) с информацией о количестве и типе сгенерированных объектов (взрослых птиц и птенцов), а также времени симуляции. Вся информация выводится в элементе TextArea, недоступном для редактирования. В диалоговом окне предусмотрены 2 кнопки: «ОК» и «Отмена». При нажатии на «ОК» симуляция останавливается, а при нажатии на «Отмена» - продолжается;

public class *Line* extends *JPanel*

Строка с информацией о времени симуляции и количестве сгенерированных объектов (птиц всего). Управляется классом *Clock* и клавишей *T*.

public class *Periods* extends *JPanel*

Текстовые поля - периоды рождения объектов. Предусмотрена проверка данных вводимых пользователем. При вводе неверного значения обрабатывается исключительная ситуация: выставляется значение по умолчанию и выводится диалоговое окно с сообщением об ошибке;

В остальные классы и их методы внесены изменения, сопутствующие появлению классов, описанных выше.

Результат работы программы

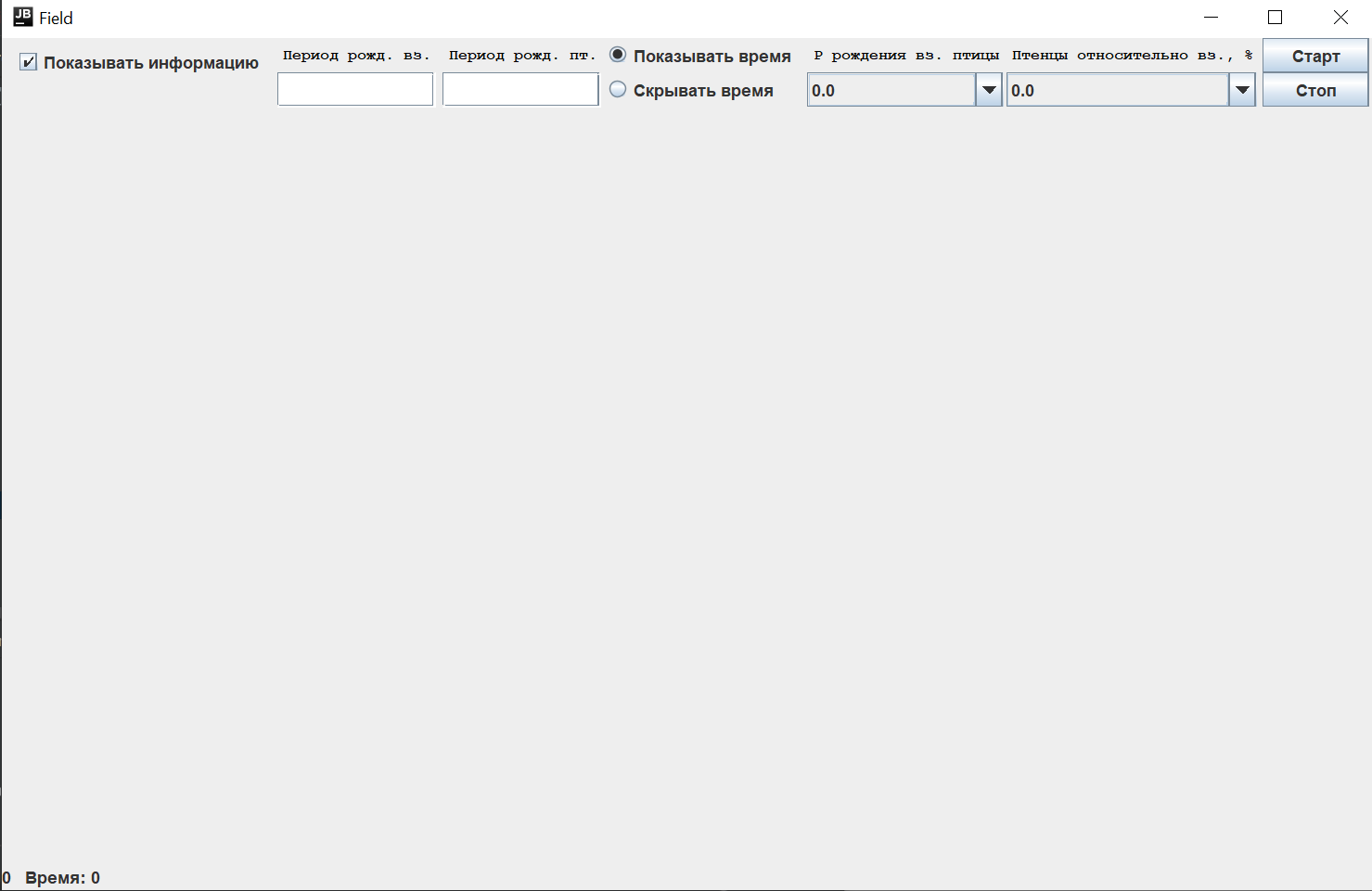


Рис. 1 – Рабочая область до ввода данных пользователем

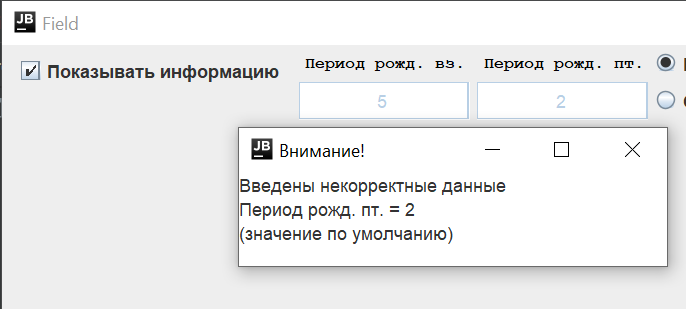


Рис. 2 – Результат ввода некорректных данных

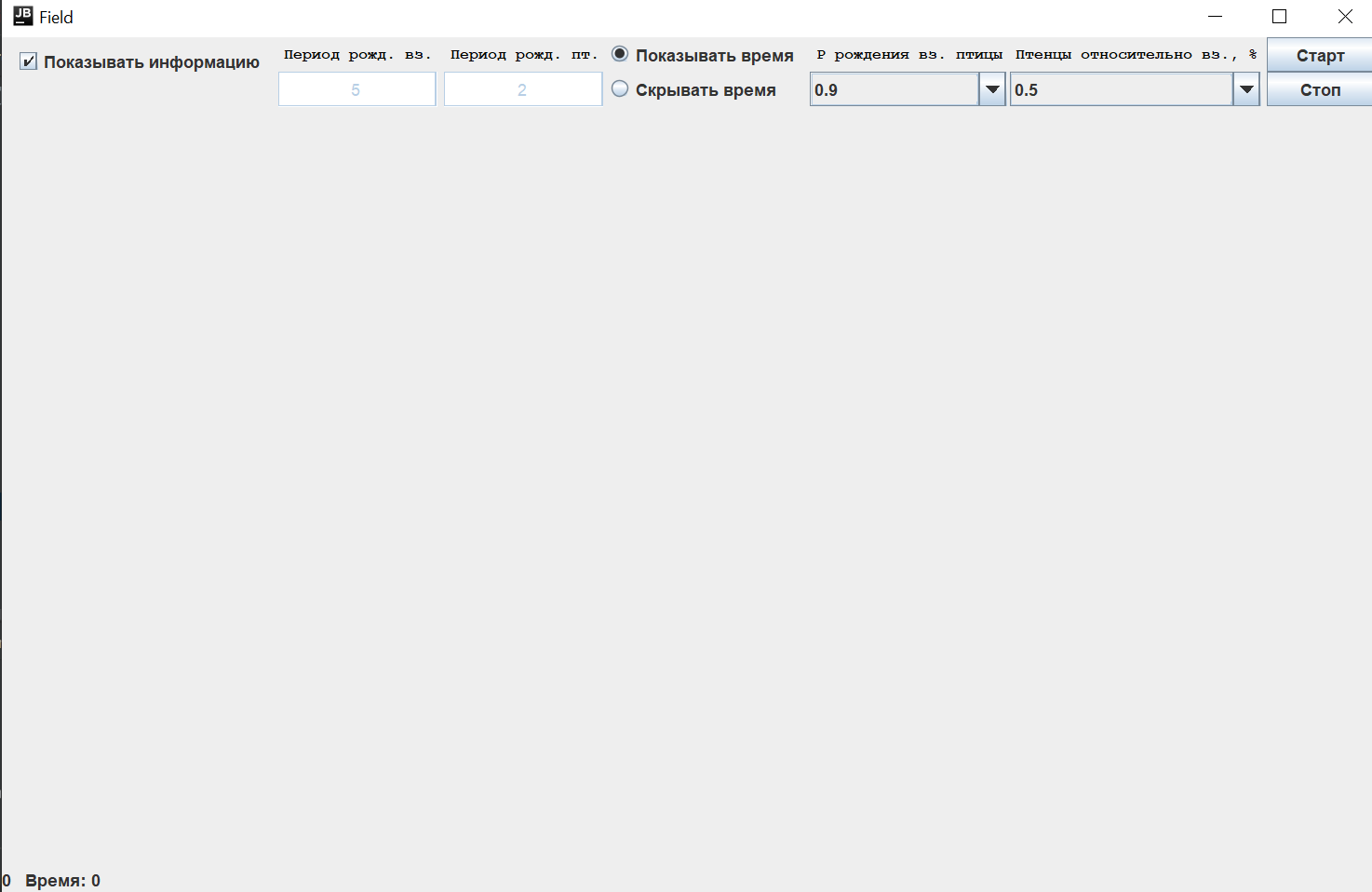


Рис. 3 – Рабочая область после ввода данных пользователем

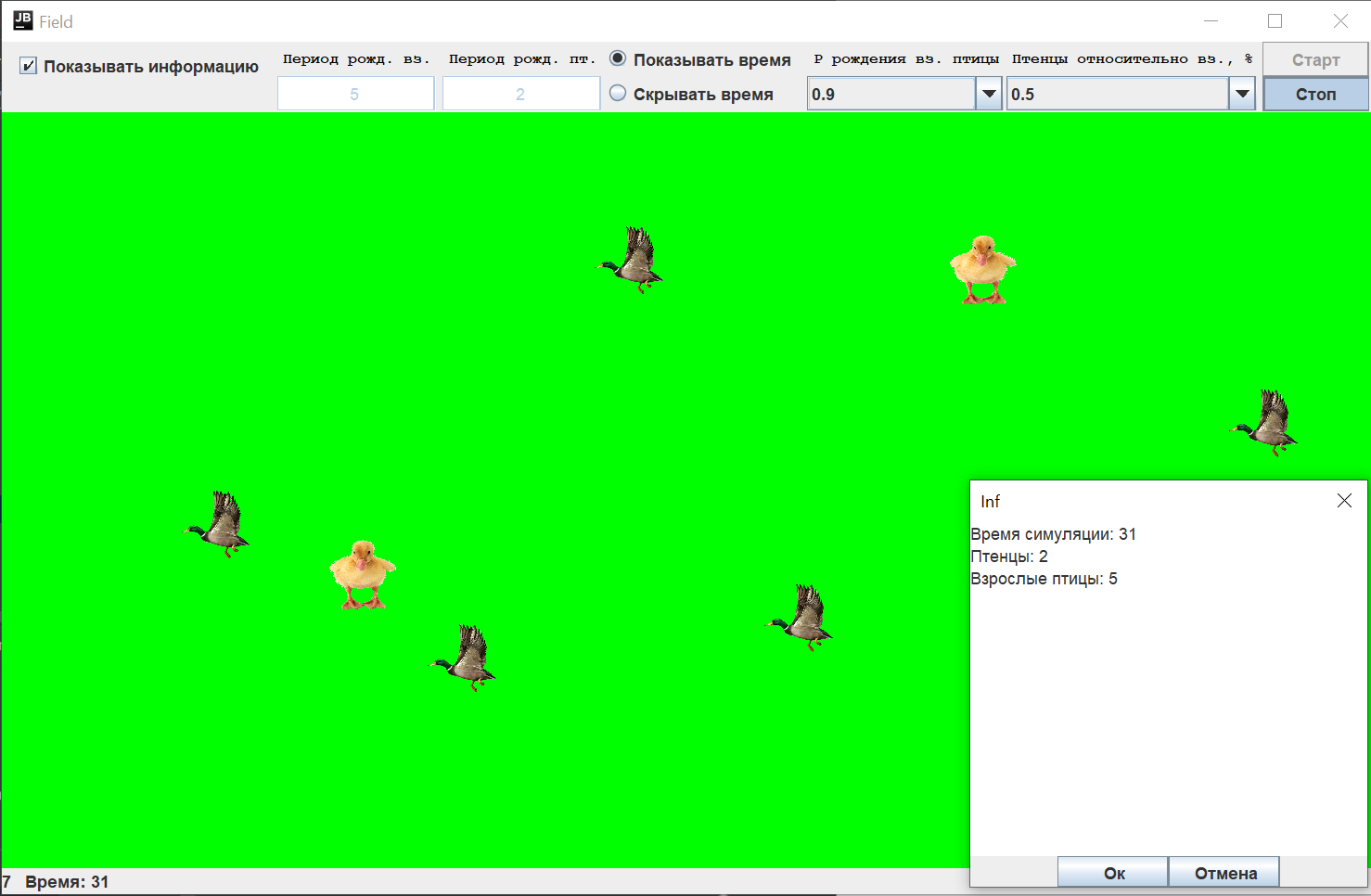


Рис. 4 – Результат работы программы



Рис. 5 – Управление видимостью времени и выводом итоговой информации

Выводы

В ходе выполнения данной лабораторной работы изучена тема «Разработка графического интерфейса приложения». В качестве закрепления изученного материала была доработана программа, созданная в лабораторной работе № 1. А именно, была поделена рабочая область окна приложения на 2 части. Визуализация перенеслась в одну часть окна, панель управления в другую. На панель управления были помещены кнопки, переключатели и другие элементы управления симуляцией. Интерфейс был дополнен поясняющими метками и появляющимся предупреждением о вводе неверных данных.

Таким образом, благодаря внесенным доработкам, пользователь сможет самостоятельно задавать начальные параметры, корректировать некоторые из них во время симуляции и самостоятельно определять видимость итоговой информации, которая ему может быть необходима.