**Министерство науки и высшего образования Российской Федерации**

**Новосибирский государственный технический университет**

**Кафедра вычислительной техники**

**Лабораторная работа №4**

**по дисциплине:** Технология программирования

**на тему:** Многопотоковые приложения

Преподаватель: Михайленко Д.А.

Студент: Голиков А.Ю.

Группа: АВТ-808

Новосибирск 2020

**Цель**: изучить особенности реализации классов-коллекций в Java.

**Задание**: изучить особенности реализации и работы потоков в Java, управлением приоритетами потоков и синхронизацией потоков.

1) создать абстрактный класс BaseAI, описывающий «интеллектуальное поведение» объектов. Класс должен создавать поток, обеспечивающий движения объектов коллекции;

2) реализовать класс BaseAI для каждого из видов объекта, включив в него поведение, описанное в индивидуальном задании по варианту;

3) синхронизовать работу потоков расчета интеллекта объектов, их рисования и генерации новых объектов. Рисование должно остаться в основном потоке;

4) добавить в панель управления кнопки для остановки и возобновления работы интеллекта каждого вида объектов. Реализовать через засыпание/пробуждение потоков (методы wait() и notify());

5) добавить в панель управления выпадающие списки для выставления приоритетов каждого из потоков.

Вариант №4

**Ход работы:**

1. Был описан абстрактный класс BaseAI, наследуемый от класса Thread и реализующий интерфейс RabbitAI.

Поля BaseAI:

protected boolean isMoving;

Методы интерфейса RabbitAI:

public void startRabbitBehaviour();  
public void stopRabbitBehaviour();  
public void setTheadPriority(int priority);

1. Был описан интерфейс Behaviour, содержащий метод движения кроликов

Для обычного кролика:

public void move() {  
 if(coordinates.x > 1550 || coordinates.y>650) {  
 coordinates.x = changeDirectionX\*random.nextInt(100);  
 coordinates.y = changeDirectionY\*random.nextInt(100);  
 return;  
 }  
 if(HabitatFrame.*getTime*() - birthTime % changeDirectionTime == 0)  
 {  
  
 var xRandomDirection = 1 + random.nextInt(100);  
 var yRandomDirection = 1 + random.nextInt(100);  
 changeDirectionX = xRandomDirection < 50 ? Direction.*Right*.getDirection(): Direction.*Left*.getDirection();  
 changeDirectionY = yRandomDirection < 50 ? Direction.*Up*.getDirection(): Direction.*Down*.getDirection();  
 }  
 coordinates.x=coordinates.x+speed\*changeDirectionX;  
 coordinates.y=coordinates.y+speed\*changeDirectionY;  
}

Для кролика альбиноса:

public void move() {  
  
 if(1480 - coordinates.x < speed)  
 {  
 changeDirectionX=Direction.*Left*.getDirection();  
 }  
 if(coordinates.x < speed )  
 {  
 changeDirectionX=Direction.*Right*.getDirection();  
 }  
 coordinates.x = coordinates.x + speed\*changeDirectionX;  
}

Для выбора направления используется Enum Direction:

package Models;  
  
public enum Direction {  
 *Right* (1),  
 *Left* (-1),  
 *Up*(1),  
 *Down*(-1);  
  
 private int direction;  
  
 Direction(int direction) {  
 this.direction = direction;  
 }  
  
 public int getDirection() {  
 return direction;  
 }  
}

1. Были добавлены кнопки Включить/Выключть поведение объектов, а также выпадающее меню с приоритетами потоков.

**Результат:**

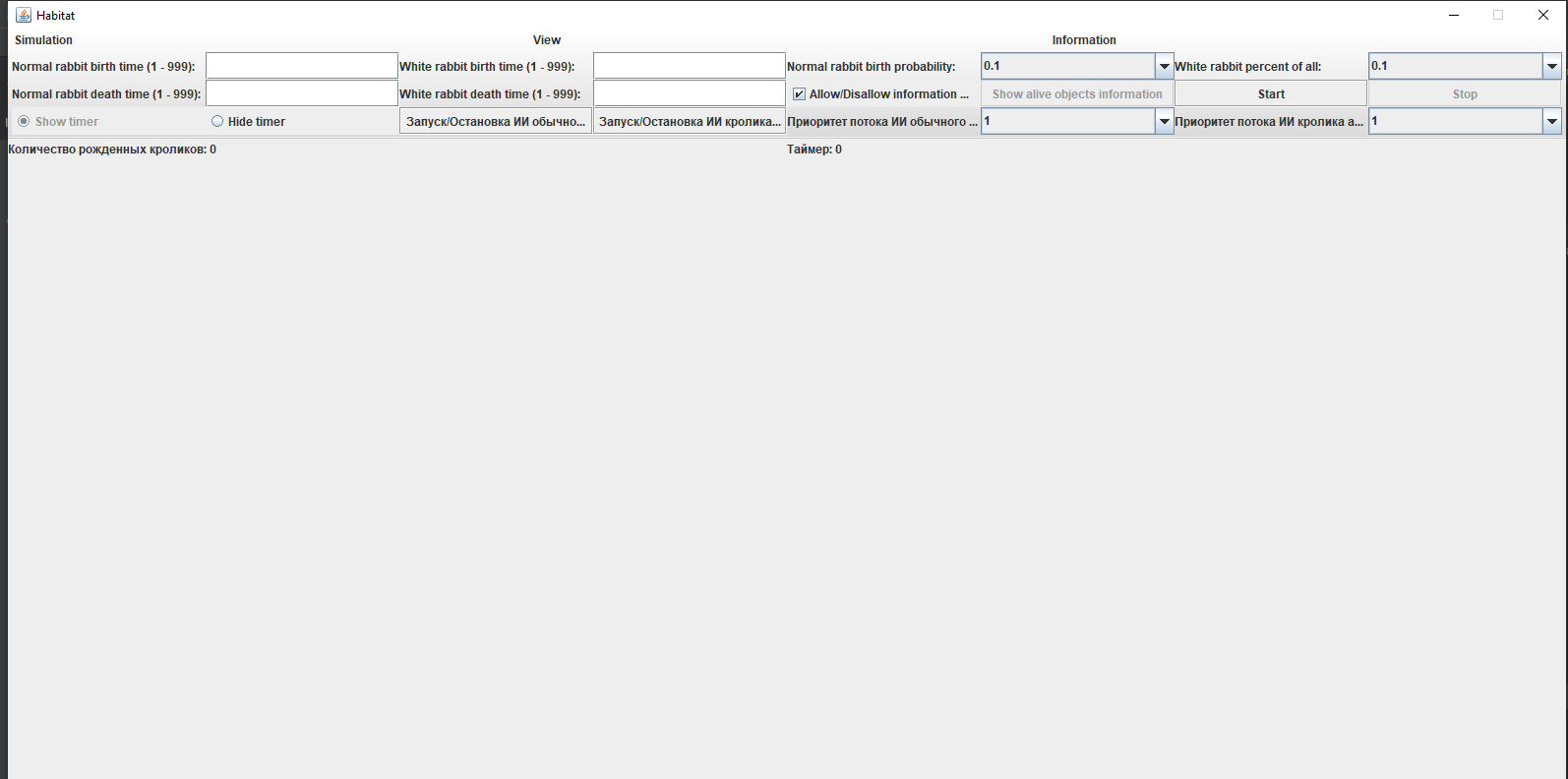


Рис. 1 – Перед симуляцией

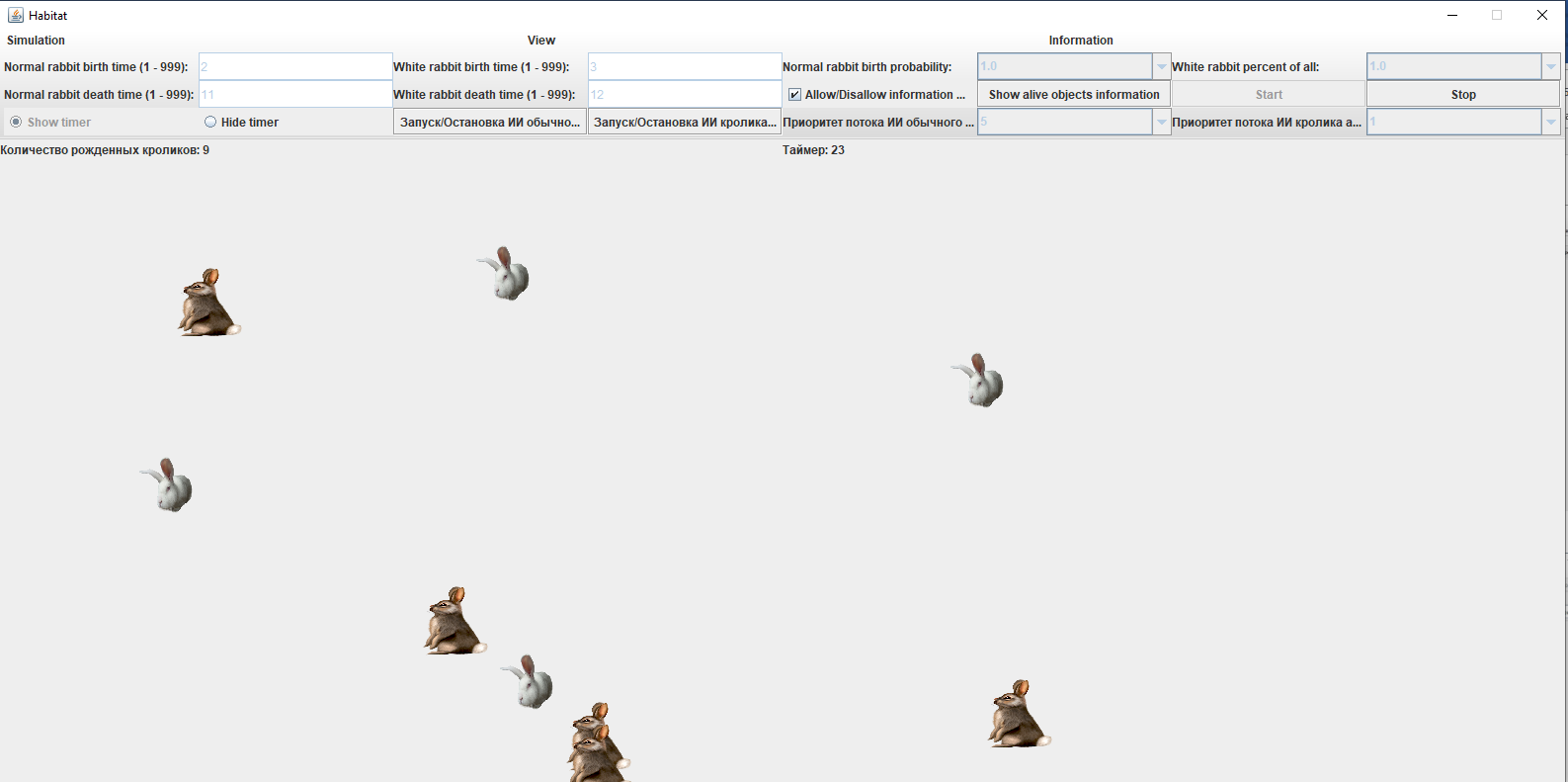


Рис. 2 – Во время симуляции

**Вывод:** в данной работе была улучшена программа симуляции рождения кроликов, использованы дополнительные потоки для задания поведения объектов.

Код расположен на ресурсе: <https://github.com/AVT-808/AVT-808/tree/Golikov_Alexey_var4/Lab4>