Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Новосибирский государственный технический университет»

Кафедра вычислительной техники

**Отчет**

По лабораторной работе №5

По дисциплине: «Технологии и методы программирования»

Потоки ввода-вывода.

Сериализация объектов в файл

Вариант 8

Выполнили: Павлюк А.С. Преподаватель: Копылова О.А.

Казанцев К.О.

Группа: АВТ-008

Факультет: АВТ

Новосибирск, 2022

**Цель работы:**

Изучить особенности реализации системы ввода-вывода в Java.

**Задание к лабораторной работе:**

Доработать программу, созданную в лабораторной работу № 4:

1. добавить в главное меню команду «Консоль». По этой команде должно появляться немодальное диалоговое окно с многострочным текстовым полем, занимающим всю область окна. В это окно можно вводить команды по варианту. В это же окно выводится реакция программы на команду;
2. для передачи команды в обрабатывающий поток использовать каналы ввода-вывода;
3. создать конфигурационный файл для программы. В конфигурационный файл должны сохраняться все настройки симуляции, т.е. все данные и состояния, которые задаются в панели управления программы. Конфигурационный файл должен читаться при запуске программы и записываться при выходе. Формат файла текстовый;
4. добавить в главное меню пункты «Загрузить» и «Сохранить». Команда «Сохранить» вызывает сериализацию всех «живых» объектов в ней. Команда «Загрузить» останавливает текущую симуляцию (если симуляция запущена) и загружает объекты из выбранного файла. Не забудьте скорректировать время рождения объектов. После открытия симуляцию можно запустить, загруженные объекты должны вести себя естественно;
5. использовать стандартные файловые диалоги.

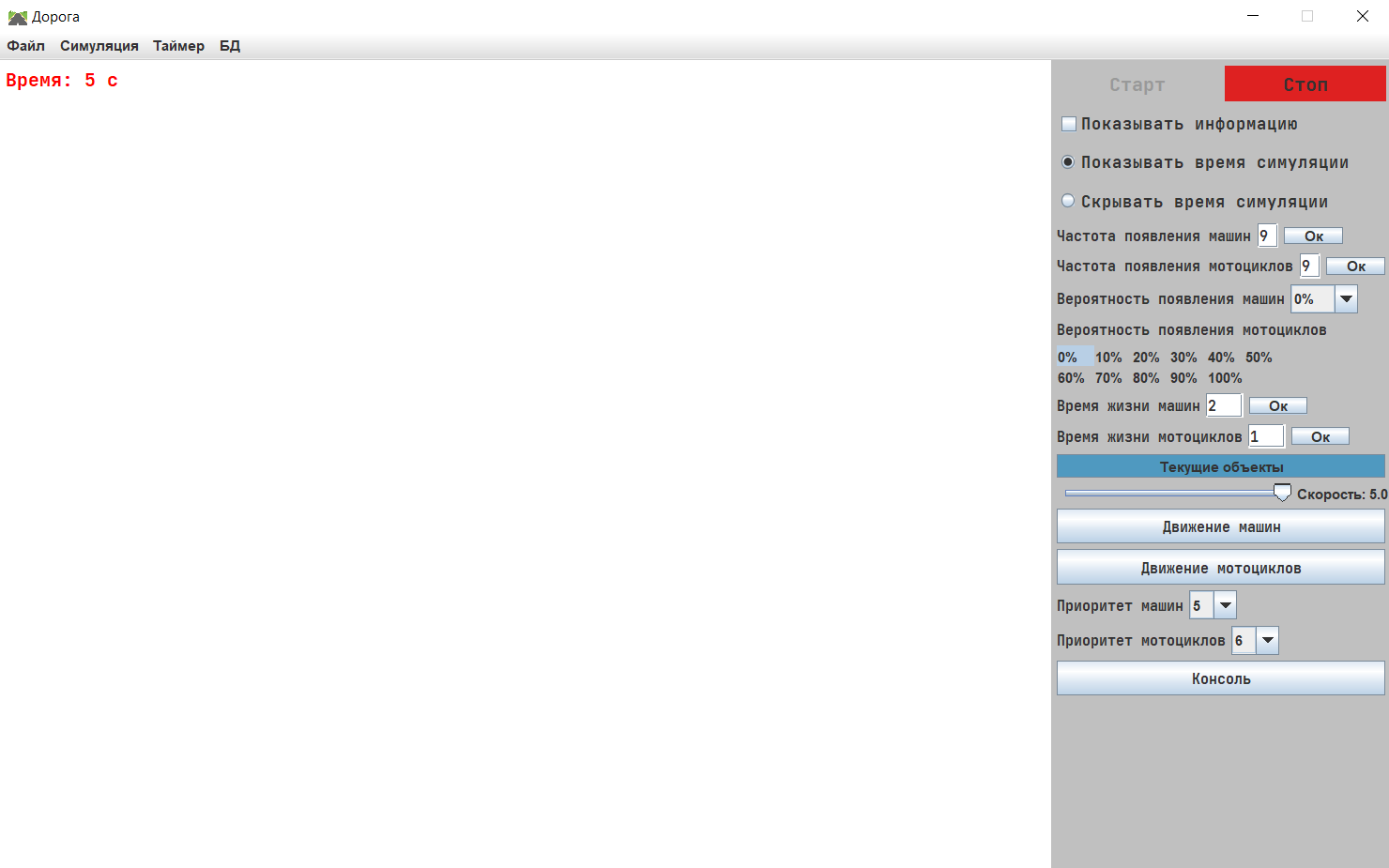
*Вариант 8*

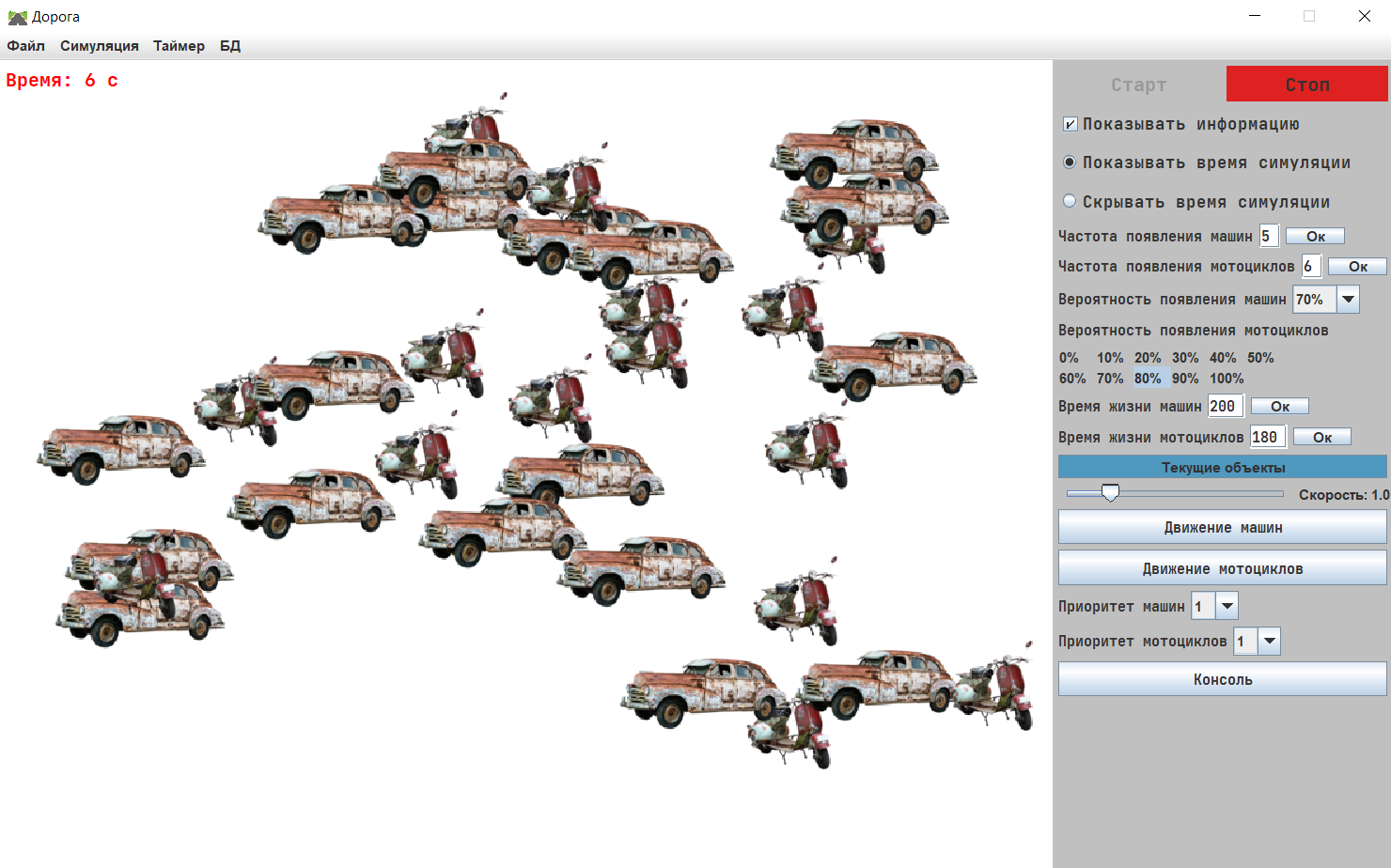
Реализовать в консоли команду «Сократить число мотоциклов на N%». Как параметр в команду должно передаваться значение N%.

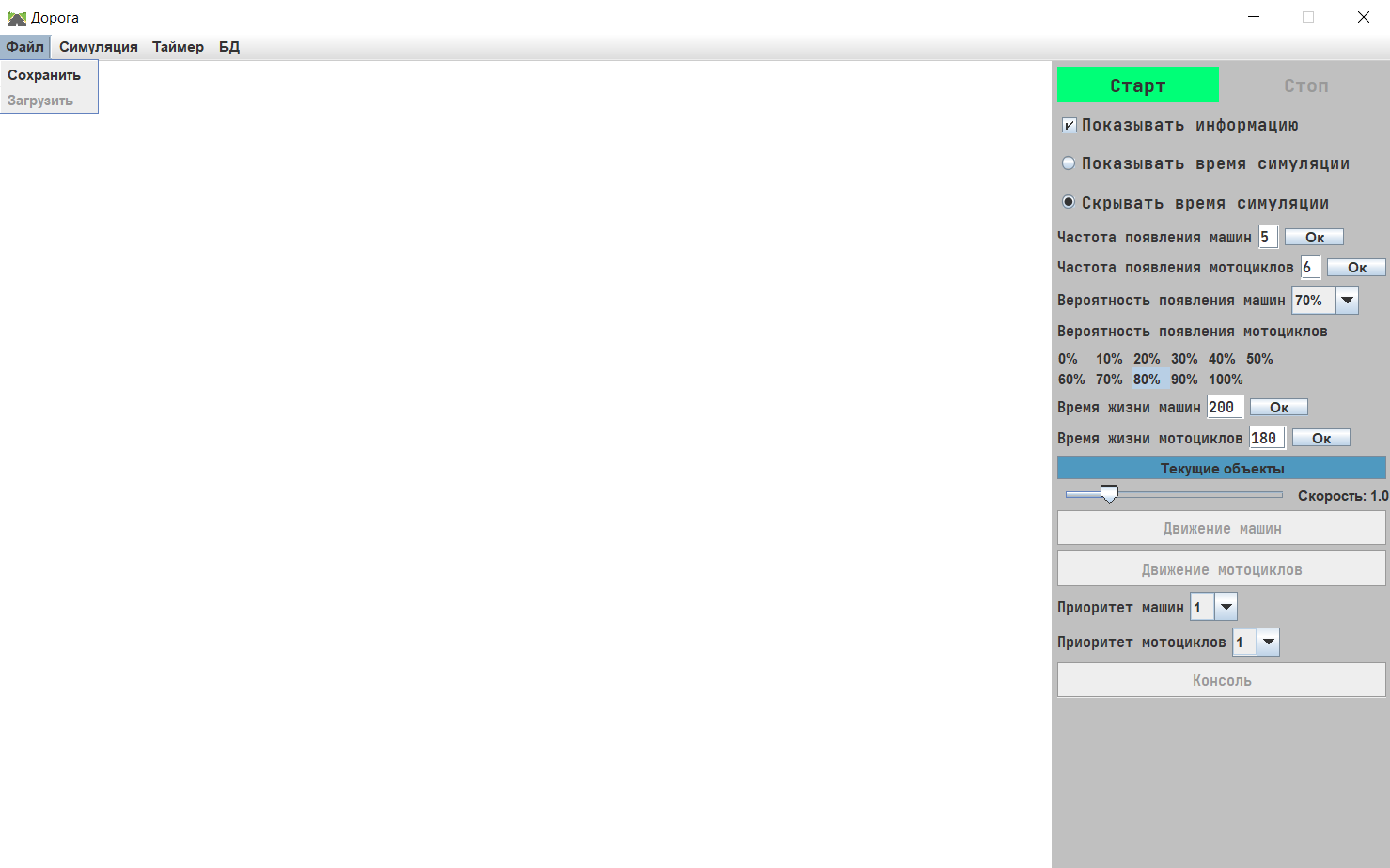
**Выполнение работы:**

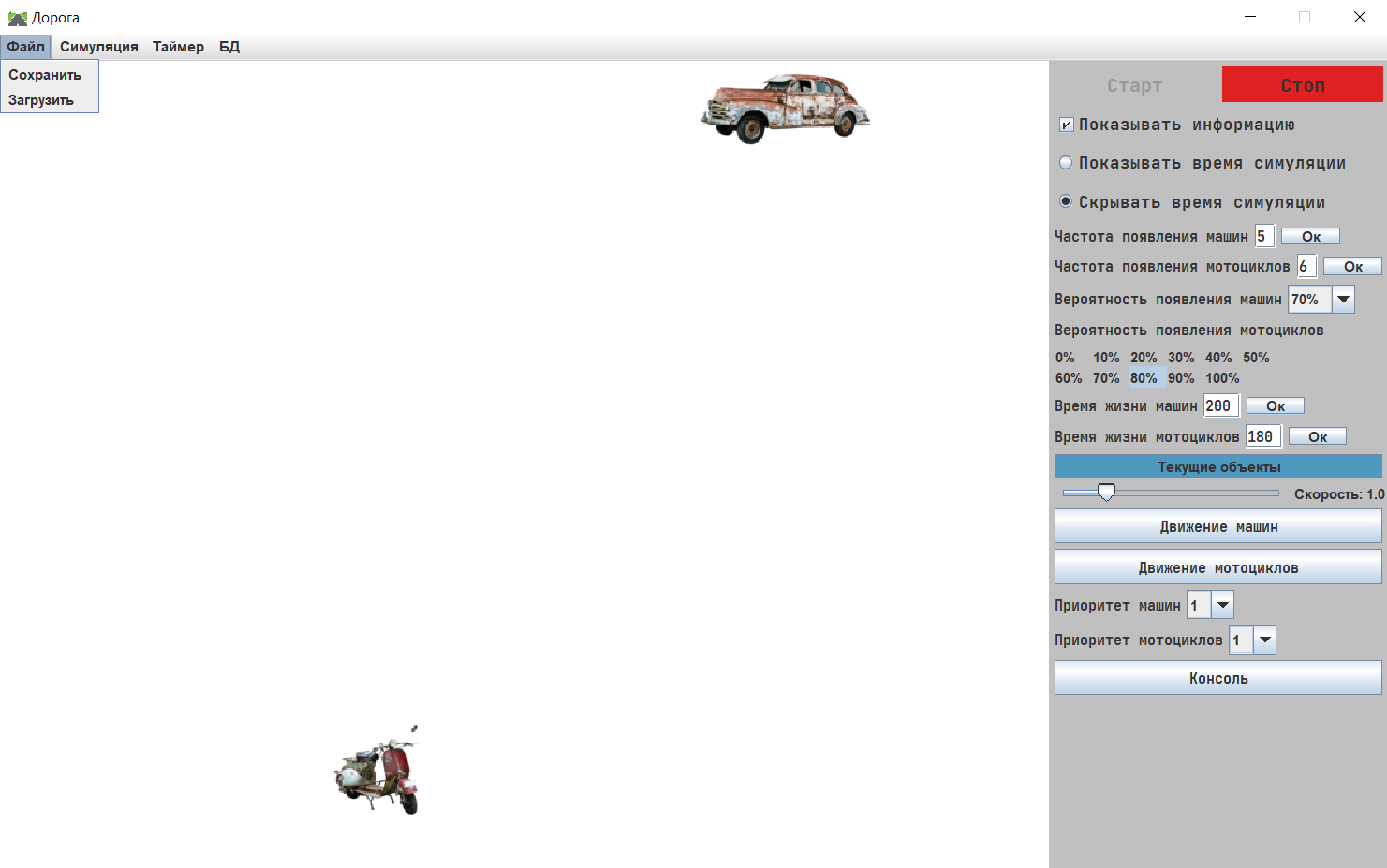
В меню были добавлены кнопки работы с файлом. Теперь пользователю доступна запись всех объектов и параметров симуляции в текстовый файл, а также их чтение. Время появление транспортных средств обнуляется при каждом чтении из файла. В панель управления добавлена кнопка вызова консоли, в которой можно сократить число мотоциклов на n%, введя соответствующую команду (пояснения присутствуют). Всё это выполнено при помощи потоков ввода/вывода. Был создан класс MyConsole, производный от JDialog. В нем прописана консольная логика.

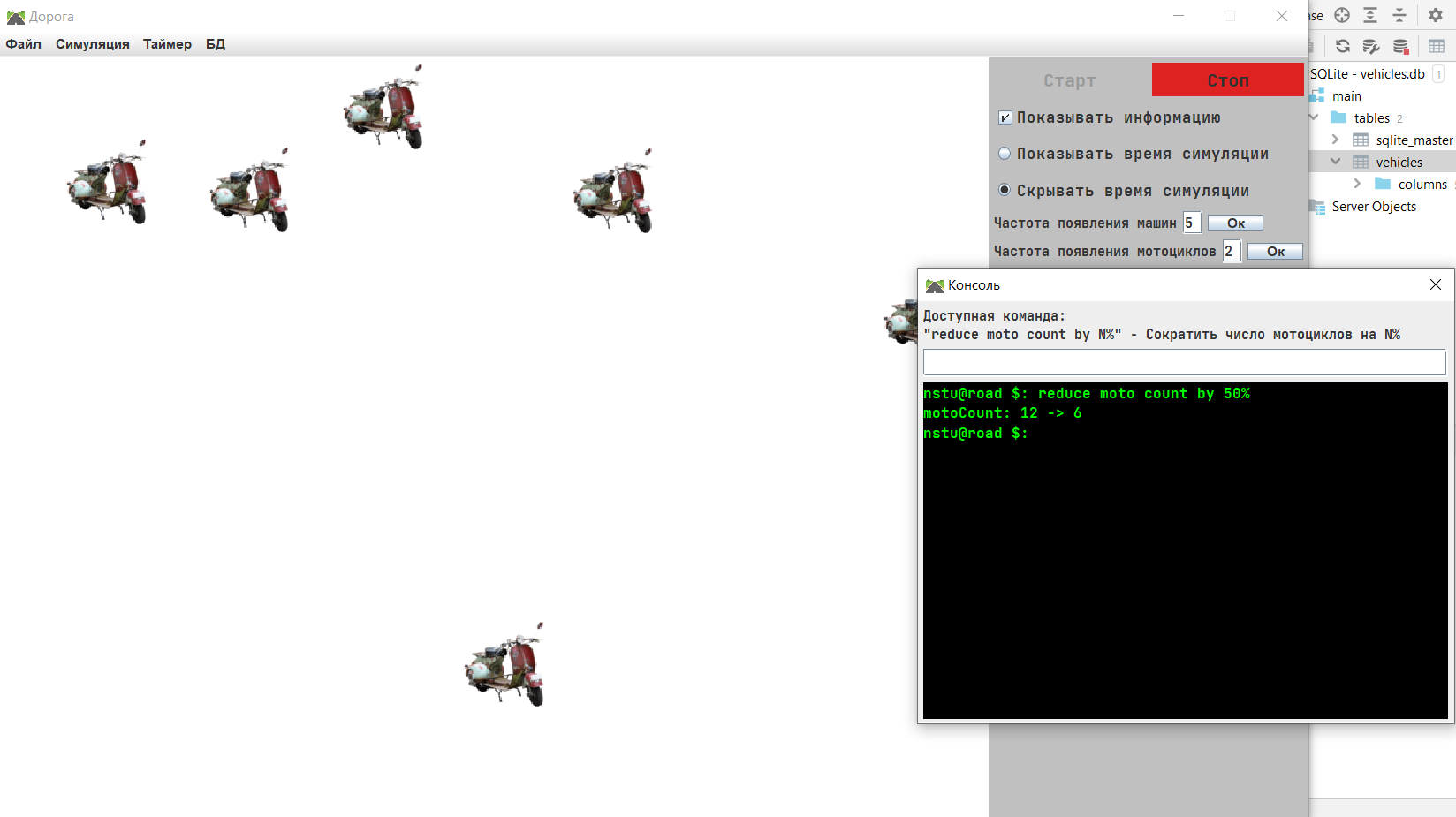
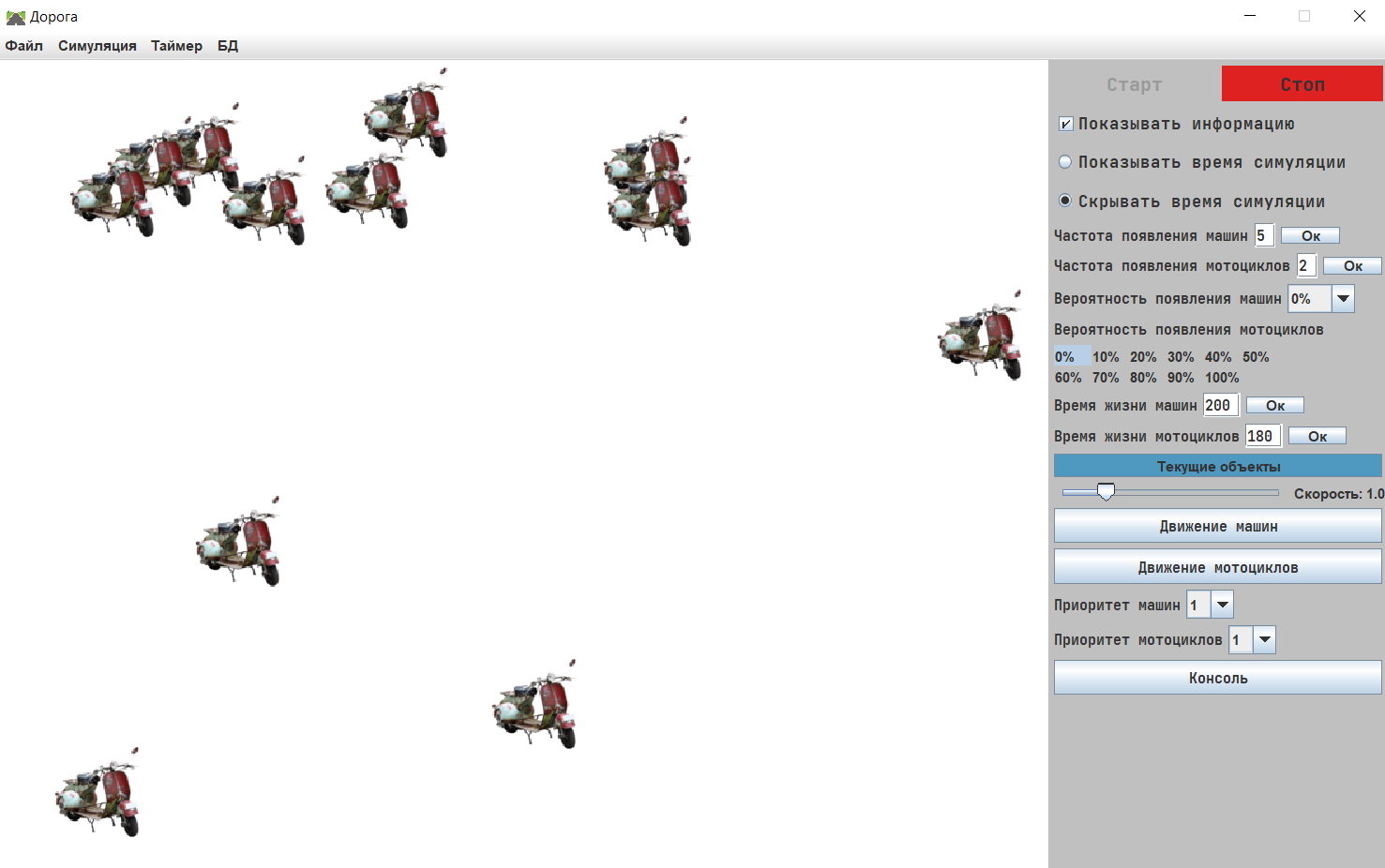
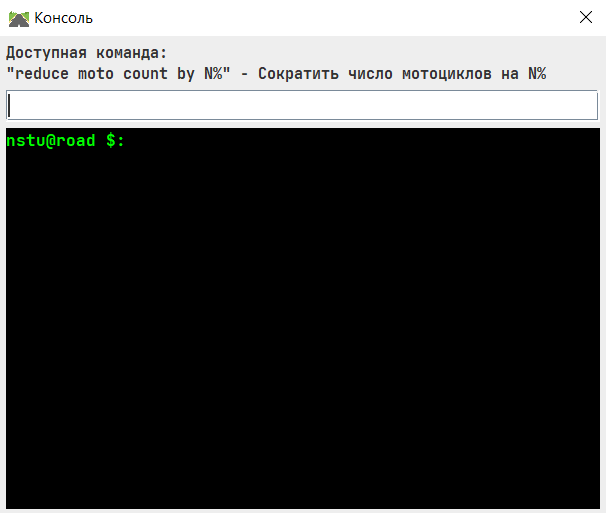
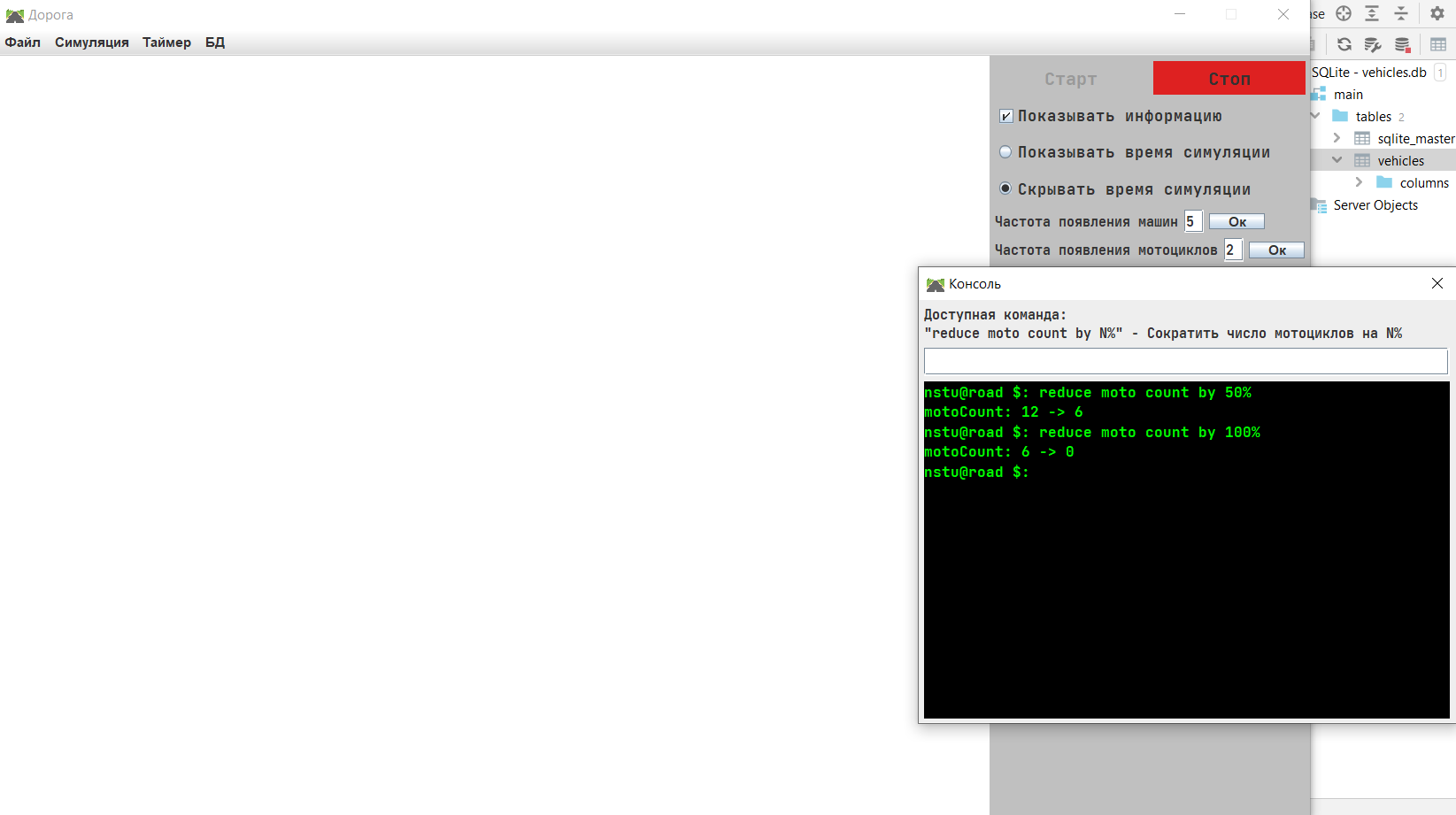
**Пример выполнения работы**









****

**Вывод:** в ходе выполнения лабораторной работы были изучены особенности реализации системы ввода-вывода в Java. Мы научились сериализовывать(десериализовывать) свои объекты, познакомились с маркерными интерфейсами.

## Класс MyConsole:

**package** nstu.client;  
  
**import** nstu.client.vehicles.Motorbike;  
**import** nstu.client.vehicles.Vehicle;  
  
**import** javax.swing.\*;  
**import** java.awt.\*;  
**import** java.io.IOException;  
**import** java.io.PipedInputStream;  
**import** java.io.PipedOutputStream;  
**import** java.io.PrintStream;  
**import** java.util.Scanner;  
  
**public class** MyConsole **extends** JDialog **implements** Runnable {  
 **private** Habitat **habitat**;  
 **private final** Scanner **fieldInput**;  
 **private final** JTextField **inputField**;  
 **private final** JTextArea **outputArea**;  
  
 MyConsole(MyFrame owner, Habitat habitat) **throws** IOException {  
 **super**(owner, **"Консоль"**, **false**);  
 **this**.**habitat** = habitat;  
 setLayout(**new** FlowLayout(FlowLayout.***LEFT***));  
 setDefaultCloseOperation(***DISPOSE\_ON\_CLOSE***);  
 setSize(500, 420);  
 setBounds(1000, 300, 500, 420);  
 setVisible(**false**);  
 requestFocus();  
  
 JLabel commands = **new** JLabel(**"<html>Доступная команда: <br>\"reduce moto count by N%\" - Сократить число мотоциклов на N%"**);  
 commands.setFont(**new** Font(**"JetBrains Mono"**, Font.***BOLD***, 12));  
  
 **outputArea** = **new** JTextArea(**"nstu@road $: "**);  
 **outputArea**.setEditable(**false**);  
 **outputArea**.setPreferredSize(**new** Dimension(475, 305));  
 **outputArea**.setBackground(Color.***BLACK***);  
 **outputArea**.setForeground(Color.***GREEN***);  
 **outputArea**.setFont(**new** Font(**"JetBrains Mono"**, Font.***BOLD***, 13));  
  
 **inputField** = **new** JTextField();  
 **inputField**.setPreferredSize(**new** Dimension(475, 25));  
 **inputField**.setFont(**new** Font(**"JetBrains Mono"**, Font.***BOLD***, 13));  
  
 add(commands);  
 add(**inputField**);  
 add(**outputArea**);  
  
 PipedOutputStream outputFromField = **new** PipedOutputStream();  
 PipedInputStream inputFromField = **new** PipedInputStream(outputFromField);  
  
 **fieldInput** = **new** Scanner(inputFromField);  
 PrintStream fieldOutput= **new** PrintStream(outputFromField);  
  
 **inputField**.addActionListener(e -> {  
 String text = **inputField**.getText();  
 fieldOutput.println(text);  
 **inputField**.setText(**""**);  
 });  
  
 **new** Thread(**this**).start();  
 }  
  
 @Override  
 **public void** run() {  
 **while** (**fieldInput**.hasNextLine()) {  
 String line = **fieldInput**.nextLine();  
 **if** (line.contains(**"reduce moto count by "**) && line.contains(**"%"**)) {  
 **int** motoCountPrev = **habitat**.**motoCount**;  
 **int** prs = Integer.*parseInt*(line.replace(**"reduce moto count by "**, **""**).replace(**"%"**, **""**));  
 **if** (prs > 100 || prs < 0) {  
 JOptionPane.*showMessageDialog*(**null**, **"N = [0; 100]"**, **"Ошибка!"**, JOptionPane.***ERROR\_MESSAGE***);  
 } **else** {  
 **habitat**.**motoCount** -= **habitat**.**motoCount** \* prs / 100;  
 **outputArea**.append(line + **"\nmotoCount: "** + motoCountPrev + **" -> "** + **habitat**.**motoCount** + **"\n"**);  
 **while** (motoCountPrev > **habitat**.**motoCount**) {  
 **for** (**int** i = 0; i < **habitat**.*vehicles*.size(); i++) {  
 Vehicle v = **habitat**.*vehicles*.get(i);  
 **if** (v **instanceof** Motorbike) {  
 **habitat**.*vehicles*.remove(i);  
 System.***out***.println(**"-motorbike{"** + v.getX() + **"; "** + v.getY() + **"; "** + v.getId() + **"}"**);  
 **break**;  
 }  
 }  
 motoCountPrev--;  
 }  
 **outputArea**.append(**"nstu@road $: "**);  
 }  
 } **else** {  
 JOptionPane.*showMessageDialog*(**null**, **"Неверная команда!"**, **"Ошибка!"**, JOptionPane.***ERROR\_MESSAGE***);  
 }  
 }  
 }  
}

**Изменения в классе MyFrame:**

**public** JMenuItem **saveSimilation**;  
**public** JMenuItem **loadSimilation**;

JMenu fileMenu = **new** JMenu(**"Файл"**);  
**saveSimilation** = **new** JMenuItem(**"Сохранить"**);  
**loadSimilation** = **new** JMenuItem(**"Загрузить"**);  
**loadSimilation**.setEnabled(**false**);  
  
**saveSimilation**.addActionListener(e -> {  
 **try** (FileOutputStream outputStream = **new** FileOutputStream(**"C:\\Users\\Александр\\Java\\Labs-4-sem\\JavaLabs\\src\\nstu\\client\\config.txt"**);  
 ObjectOutputStream objectOutputStream = **new** ObjectOutputStream(outputStream)) {  
 objectOutputStream.writeObject(**h**.*vehicles*);  
 objectOutputStream.writeObject(**h**.*ids*);  
 objectOutputStream.writeObject(**h**.*times*);  
  
 **int** isInfo = **showInfo**.isSelected() ? 1 : 0;  
 **int** showT = **showTimer**.isSelected() ? 1 : 0;  
 **int** hideT = **hideTimer**.isSelected() ? 1 : 0;  
 outputStream.write(isInfo);  
 outputStream.write(showT);  
 outputStream.write(hideT);  
  
 outputStream.write(**h**.*N1*);  
 outputStream.write(**h**.*N2*);  
 outputStream.write(**h**.*P1*);  
 outputStream.write(**h**.*P2*);  
  
 outputStream.write((**int**) Car.*getTimeLifeCar*());  
 outputStream.write((**int**) Motorbike.*getTimeLifeMoto*());  
  
 outputStream.write((**int**) (**h**.*speed* \* 10));  
 outputStream.write(**carMoving**.getPriority());  
 outputStream.write(**motoMoving**.getPriority());  
  
 } **catch** (IOException exp) {  
 System.***out***.println(**"OutputException: "** + exp.getMessage());  
 }  
});  
  
**loadSimilation**.addActionListener(e -> {  
 **try** (FileInputStream inputStream = **new** FileInputStream(**"C:\\Users\\Александр\\Java\\Labs-4-sem\\JavaLabs\\src\\nstu\\client\\config.txt"**);  
 ObjectInputStream objectInputStream = **new** ObjectInputStream(inputStream)) {  
 *time* = 0;  
 **h**.**carCount** = 0;  
 **h**.**motoCount** = 0;  
 **h**.*vehicles*.clear();  
 **h**.*ids*.clear();  
 **h**.*times*.clear();  
 **h**.*vehicles* = (List<Vehicle>) objectInputStream.readObject();  
 **h**.*ids* = (Set<Integer>) objectInputStream.readObject();  
 **h**.*times* = (Map<Integer, Long>) objectInputStream.readObject();  
  
 **boolean** isInfoB = inputStream.read() == 1;  
 **boolean** showTB = inputStream.read() == 1;  
 **boolean** hideTB = inputStream.read() == 1;  
 **showInfo**.setSelected(isInfoB);  
 **showInfoItem**.setSelected(isInfoB);  
 **showTimerItem**.setSelected(showTB);  
 **showTimer**.setSelected(showTB);  
 **hideTimer**.setSelected(hideTB);  
 **hideTimerItem**.setSelected(hideTB);  
 **timeLabel**.setVisible(showTB);  
  
 **h**.*N1* = inputStream.read();  
 **h**.*N2* = inputStream.read();  
 **h**.*P1* = inputStream.read();  
 **h**.*P2* = inputStream.read();  
  
 Car.*setTimeLifeCar*(inputStream.read());  
 Motorbike.*setTimeLifeMoto*(inputStream.read());  
  
 **h**.*speed* = (**float**) (inputStream.read() / 10);  
 **carMoving**.setPriority(inputStream.read());  
 **motoMoving**.setPriority(inputStream.read());  
  
 **for** (**int** i = 0; i < **h**.*vehicles*.size(); i++) {  
 Vehicle v = **h**.*vehicles*.get(i);  
 v.setTimeAppear(0);  
 **h**.*times*.put(v.getId(), 0L);  
 **if** (v **instanceof** Car) {  
 **h**.**carCount**++;  
 } **else** {  
 **h**.**motoCount**++;  
 }  
 }  
 **carsFreqText**.setText(**""** + **h**.*N1*);  
 **motoFreqText**.setText(**""** + **h**.*N2*);  
 **carProbability**.setSelectedItem(**""** + **h**.*P1* + **"%"**);  
 **motoProbability**.setSelectedIndex(**h**.*P2* / 10);  
 **carsTimeText**.setText(**""** + Car.*getTimeLifeCar*());  
 **motoTimeText**.setText(**""** + Motorbike.*getTimeLifeMoto*());  
 **speedSlider**.setValue((**int**) (**h**.*speed* \* 10));  
 **carPriority**.setSelectedItem(**""** + **carMoving**.getPriority());  
 **motoPriority**.setSelectedItem(**""** + **motoMoving**.getPriority());  
  
  
 System.***out***.println(**h**.*N1* + **" "** + **h**.*N2* + **" "** + **h**.*P1* + **" "** + **h**.*P2* + **" "** + **h**.*speed* + **" "** + **carMoving**.getPriority() + **" "** +  
 **motoMoving**.getPriority() + **" "**);  
  
 } **catch** (IOException ex1) {  
 ex1.printStackTrace();  
 } **catch** (ClassNotFoundException ex2) {  
 System.***out***.println(**"ClassNotFoundException: "** + ex2.getMessage());  
 }  
});

fileMenu.add(**saveSimilation**);  
fileMenu.add(**loadSimilation**);