# **МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ»**



##### **Факультет управления и информатики в технологических системах**

###### **Кафедра Информационной безопасности**

**Направление подготовки**

**(специальность) 10.05.03 Информационная безопасность автоматизированных систем**

**Отчет**

**по практике по технологиям и методам программирования**

наименование (вид) практики

Выполнил студент гр. УБ-42

Крылов Никита Романович

###### (Ф.И.О.)

###### \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись)*

###### Проверили:

Маслов А.А.

*(Ф.И.О.)*

*(оценка) (подпись)*

*(дата)*

Воронеж - 2025

**Задание 15.**

Продажа автомобилей. Организовать сохранение объектов из коллекции в текстовый файл и загрузку объектов из текстового файла в коллекцию. Изменить пользовательское меню для добавления работы с внешними файлами

**Код программы:**

package Pr7;

import java.io.\*;

import java.time.LocalDate;

import java.util.\*;

import java.nio.file.\*;

class Automobile {

protected String brand;

protected int year;

protected double price;

protected String configuration;

protected String countryOfOrigin;

protected LocalDate saleDate;

protected String buyerName;

public Automobile(String brand, int year, double price, String configuration,

String countryOfOrigin, LocalDate saleDate, String buyerName) {

this.brand = brand;

this.year = year;

this.price = price;

this.configuration = configuration;

this.countryOfOrigin = countryOfOrigin;

this.saleDate = saleDate;

this.buyerName = buyerName;

}

// Convert object to CSV string

public String toCsvString() {

return String.format("%s,%d,%.2f,%s,%s,%s,%s",

brand, year, price, configuration, countryOfOrigin, saleDate, buyerName);

}

// Create object from CSV string

public static Automobile fromCsvString(String line) {

String[] parts = line.split(",");

return new Automobile(

parts[0], // brand

Integer.parseInt(parts[1]), // year

Double.parseDouble(parts[2]), // price

parts[3], // configuration

parts[4], // countryOfOrigin

LocalDate.parse(parts[5]), // saleDate

parts[6] // buyerName

);

}

@Override

public String toString() {

return String.format("Марка: %s\nГод выпуска: %d\nЦена: %.2f\nКомплектация: %s\nСтрана производитель: %s\nДата продажи: %s\nПокупатель: %s",

brand, year, price, configuration, countryOfOrigin, saleDate, buyerName);

}

}

class CarSales {

private List<Automobile> soldCars = new ArrayList<>();

private static final String FILE\_NAME = "car\_sales.txt";

public void addCar(Automobile car) {

soldCars.add(car);

saveToFile();

}

public void sortByBrand() {

soldCars.sort(Comparator.comparing(car -> car.brand));

}

public void sortByPrice() {

soldCars.sort(Comparator.comparingDouble(car -> car.price));

}

public void printSales() {

if (soldCars.isEmpty()) {

System.out.println("Список автомобилей пуст.");

return;

}

for (Automobile car : soldCars) {

System.out.println(car);

System.out.println("----------------------");

}

}

public void saveToFile() {

try {

List<String> lines = new ArrayList<>();

for (Automobile car : soldCars) {

lines.add(car.toCsvString());

}

Files.write(Paths.get(FILE\_NAME), lines);

System.out.println("Данные успешно сохранены в " + FILE\_NAME);

} catch (IOException e) {

System.out.println("Ошибка при сохранении данных: " + e.getMessage());

}

}

public void loadFromFile() {

try {

if (!Files.exists(Paths.get(FILE\_NAME))) {

System.out.println("Файл данных не найден. Создан новый список.");

return;

}

List<String> lines = Files.readAllLines(Paths.get(FILE\_NAME));

soldCars.clear();

for (String line : lines) {

if (!line.trim().isEmpty()) {

soldCars.add(Automobile.fromCsvString(line));

}

}

System.out.println("Данные успешно загружены из " + FILE\_NAME);

} catch (IOException e) {

System.out.println("Ошибка при загрузке данных: " + e.getMessage());

} catch (Exception e) {

System.out.println("Ошибка при обработке данных: " + e.getMessage());

}

}

}

public class Main {

private static final Scanner scanner = new Scanner(System.in);

private static final CarSales sales = new CarSales();

public static void main(String[] args) {

sales.loadFromFile();

while (true) {

System.out.println("\nМеню:");

System.out.println("1. Добавить автомобиль");

System.out.println("2. Вывести список автомобилей (сортировка по марке)");

System.out.println("3. Вывести список автомобилей (сортировка по цене)");

System.out.println("4. Сохранить данные в файл");

System.out.println("5. Загрузить данные из файла");

System.out.println("6. Выход");

System.out.print("Выберите опцию: ");

try {

int choice = Integer.parseInt(scanner.nextLine());

switch (choice) {

case 1:

addCar();

break;

case 2:

sales.sortByBrand();

sales.printSales();

break;

case 3:

sales.sortByPrice();

sales.printSales();

break;

case 4:

sales.saveToFile();

break;

case 5:

sales.loadFromFile();

break;

case 6:

System.out.println("Программа завершена.");

return;

default:

System.out.println("Неверный ввод, попробуйте снова.");

}

} catch (NumberFormatException e) {

System.out.println("Пожалуйста, введите число.");

}

}

}

private static void addCar() {

try {

System.out.print("Введите марку автомобиля: ");

String brand = scanner.nextLine();

System.out.print("Введите год выпуска: ");

int year = Integer.parseInt(scanner.nextLine());

System.out.print("Введите цену: ");

double price = Double.parseDouble(scanner.nextLine());

System.out.print("Введите комплектацию: ");

String configuration = scanner.nextLine();

System.out.print("Введите страну производитель: ");

String country = scanner.nextLine();

System.out.print("Введите дату продажи (ГГГГ-ММ-ДД): ");

LocalDate saleDate = LocalDate.parse(scanner.nextLine());

System.out.print("Введите ФИО покупателя: ");

String buyer = scanner.nextLine();

sales.addCar(new Automobile(brand, year, price, configuration, country, saleDate, buyer));

System.out.println("Автомобиль успешно добавлен!");

} catch (Exception e) {

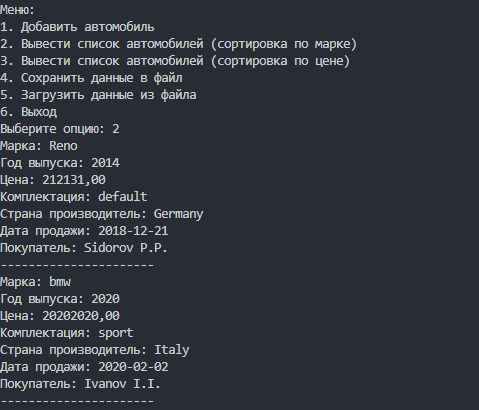
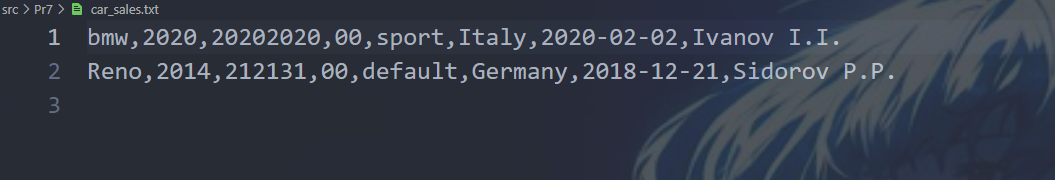
System.out.println("Ошибка при вводе данных: " + e.getMessage());

}

}

}

**Результат программы.**



**Вывод**

В ходе работы я изучил систему ввода-вывода Java, включая чтение и запись файлов, обработку исключений и управление данными с помощью потоков. Программа для учета продаж автомобилей наглядно продемонстрировала, как эффективно работать с файлами, сериализацией и коллекциями.

1. Работа с файлами и потоками

Программа использует классы Files, Path и методы write()/readAllLines() для сохранения и загрузки данных в текстовый файл. Это иллюстрирует ключевые аспекты ввода-вывода:

Простота – стандартные методы Java позволяют легко читать и записывать данные без низкоуровневого кода.

Надежность – обработка исключений (IOException) гарантирует стабильность программы при ошибках доступа к файлам.

Гибкость – данные хранятся в CSV-формате, что обеспечивает совместимость с другими приложениями.

2. Сериализация и преобразование объектов

Класс Automobile поддерживает методы toCsvString() и fromCsvString(), что демонстрирует принципы сериализации:

Структурированное хранение – объекты преобразуются в строки CSV, что упрощает их запись в файл.

Обратимость – данные можно восстановить из файла в объекты, сохраняя их состояние.

Удобство – такой подход позволяет избежать сложных форматов (например, бинарной сериализации), сохраняя читаемость данных.

3. Практическое применение

Программа охватывает несколько важных аспектов:

Интерактивный ввод – с помощью Scanner пользователь может добавлять новые автомобили.

Сортировка данных – методы sortByBrand() и sortByPrice() используют Comparator для упорядочивания коллекций.

Работа с файлами – автоматическое сохранение при добавлении данных и загрузка при запуске.

**Итог**

Система ввода-вывода Java предоставляет мощные инструменты для работы с файлами и потоками данных. Использование CSV-формата, обработка исключений и методы сериализации объектов позволяют создавать надежные приложения для хранения и управления данными.

Эта работа помогла мне закрепить навыки работы с файлами, коллекциями и пользовательским вводом, а также лучше понять важность правильной обработки ошибок ввода-вывода.