МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»  
(Самарский университет)

Институт информатики и кибернетики\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра программных систем\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**  
  
по лабораторной работе №5

«Работа с файлами»

по дисциплине «Логическое программирование»

Обучающаяся в группе 6301-020302D Соколова Алёна

Преподаватель Лобанков Антон Алексеевич

Самара 2025

# ВВЕДЕНИЕ

Целью работы является в процессе выполнения заданий ознакомиться с сохранением, чтением, обновлением данных в текстовых файлах.

Задания:

1. написать предикат, который будет логировать все действия пользователя в файл actions.log в формате «Время – действие»;
2. на основании лабораторной работы №3 разработать структуру для хранения данных в XML-документе. Перенести все данные, используемые для инициализации базы фактов, в XML;
3. написать предикат для загрузки данных из XML-документа и добавления этих данных в базу фактов;
4. написать предикат для сохранения новых данных из базы фактов в XML-документ;
5. залогировать все действия пользователя, включая чтение из файла, сохранение в файл, добавление и удаление фактов, выполнение запросов и вывод их результатов, а также очистку и инициализацию базы фактов.

### Задание 1. Логирование действий

Создаём предикат log\_action/1, который записывает в actions.log строку вида: «2025-04-15 17:42:33 - Действие».

Строчки кода созданного предиката:

log\_action(Action) :- % логирование

get\_time(TimeStamp),

stamp\_date\_time(TimeStamp, DateTime, 'UTC'),

format\_time(atom(TimeStr), '%Y-%m-%d %H:%M:%S', DateTime),

open('actions.log', append, Stream),

format(Stream, '~w - ~w~n', [TimeStr, Action]),

close(Stream).

### Задание 2. Структура данных в XML и перенос инициализации

Необходимо было создать XML-документ на основе данных из лабораторной работы №3.

Все данные перенесены следующим образом:

<cars>

<car>

<seller>AutoWorld</seller>

<brand>Toyota</brand>

<model>Corolla</model>

<price>15000</price>

<color>красный</color>

</car>

...

### Задание 3. Загрузка данных из XML в базу фактов

Был реализован предикат для чтения из XML и добавления фактов. Код представлен ниже:

load\_cars\_from\_xml(File) :- % загрузка из XML

load\_xml\_file(File, [element(cars, \_, Cars)]),

retractall(car(\_, \_, \_, \_, \_)),

forall(member(element(car, \_, [

element(seller, \_, [SellerText]),

element(brand, \_, [BrandText]),

element(model, \_, [ModelText]),

element(price, \_, [PriceText]),

element(color, \_, [ColorText])

]), Cars),

(

atom\_number(PriceText, Price),

assertz(car(SellerText, BrandText, ModelText, Price, ColorText))

)),

log\_action('Загрузка данных из XML').

### Задание 4. Сохранение базы фактов в XML

Необходимо сохранить текущие факты в XML-документ. Код предиката:

save\_cars\_to\_xml(File) :- % сохранение в XML

findall(element(car, [], [

element(seller, [], [Seller]),

element(brand, [], [Brand]),

element(model, [], [Model]),

element(price, [], [PriceAtom]),

element(color, [], [Color])

]),

(car(Seller, Brand, Model, Price, Color),

atom\_number(PriceAtom, Price)), Cars),

XML = [element(cars, [], Cars)],

open(File, write, Stream, [encoding(utf8)]),

xml\_write(Stream, XML, [header(true)]),

close(Stream),

log\_action('Сохранение данных в XML').

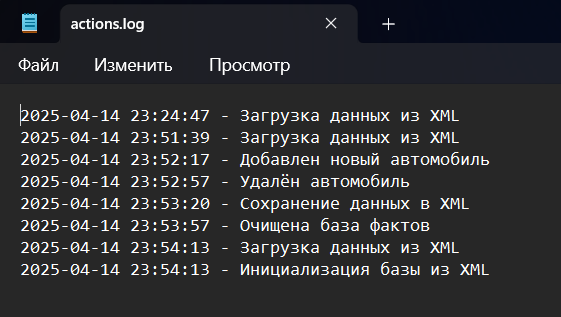
Вызов save\_cars\_to\_xml('cars\_new.xml') сохранит текущие факты в файл и добавит запись в actions.log.

### Задание 5. Логирование всех действий

Реализовано через встраивание log\_action(...) в каждый предикат:

* load\_cars\_from\_xml/1: логирует загрузку;
* save\_cars\_to\_xml/1: логирует сохранение;
* add\_car/5 и delete\_car/2: логируют изменения;
* retractall/1 при загрузке: логирует очистку базы.

На рисунке 1 представлены записи из файла actions.log.

  
Рисунок 1 – Файл actions.log

Использовались модуль use\_module(library(sgml)) для работы с XML и use\_module(library(date)) для времен логов.

После написания кода, загружаем файл в SWI-Prolog и делаем проверку работы (см. рисунки 2-4).

  
  
Рисунок 2 – Добавление в файл и проверка  
Рисунок 3 – Удаление из файла и проверка

  
Рисунок 4 – Сохранение нового файла, очистка и проверка,  
инициализация базы данных

# ЗАКЛЮЧЕНИЕ

В ходе работы была реализована система управления базой данных автомобилей с использованием Prolog и XML. Созданы предикаты для загрузки данных из XML, добавления, удаления и сохранения данных обратно в XML. Реализовано логирование всех действий пользователя в файл actions.log. Система успешно выполняет операции с базой данных и позволяет отслеживать изменения.