# Logic Programming Lab 1: Lists

\*\*Цель работы\*\*: Первоначальное ознакомление с выбранной системой программирования на языке Пролог, реализация предикатов обработки числовых и символьных списков, совершение запросов к реляционному представлению данных о предметной области.

[Отчет](REPORT.md)

## Часть 1: Предикаты работы со списками

1. Ознакомится с одной из систем программирования на языке Пролог на персональной или мини-ЭВМ (J#, GNU Prolog, BinProlog, CProlog, AMZI Prolog, Visual Prolog, TurboProlog, JLog или др.), освоить операции загрузки простейших пролог-программ и формулирования запросов.

2. Проверить наличие в системе программирования встроенных стандартных предикатов обработки списков.

3. Реализовать свои версии стандартных предикатов обработки списков, рассмотренные на занятии (`length`, `member`, `append`, `remove`, `permute`, `sublist`), убедиться в их работоспособности на ряде различных запросов.

4. Реализовать специальный предикат обработки списка в соответствии с вариантом задания двумя способами: на основе стандартных предикатов обработки списков и без их использования.

5. Реализовать предикат обработки числового списка (списков) в соответствии с вариантом задания двумя способами.

6. Привести какой-нибудь содержательный пример совместного использования предикатов, реализованных в пунктах 3 и 4.

\*\*Предикаты обработки списков\*\* (вариант вычислять как (N mod 19)+1)

1. Получение последнего элемента

2. Удаление последнего элемента

3. Удаление трех последних элементов

4. Удаление трех первых элементов

5. Удаление N первых элементов

6. Удаление N последних элементов

7. Усечение списка до указанной длины

8. Добавление элемента в конец списка

9. Получение N-го элемента списка

10. Вставка элемента в список на указанную позицию

11. Удаление элемента с заданным номером

12. Удаление всех элементов списка по значению

13. Нахождение элемента списка, следующего за данным

14. Замена N-го элемента списка на указанный

15. Замена всех элементов списка с указанным значением на другое значение

16. Нахождение номера первого вхождения элемента со заданным значением

17. Отделение хвоста, начиная с элемента с данным значением

18. Подсчет числа вхождений заданного элемента в список

19. Циклический сдвиг списка вправо

\*\*Предикаты обработки числовых списков\*\* (вариант вычислять как (N+5) mod 20 + 1)

1. Вычисление суммы элементов

2. Вычисление произведения элементов

3. Вычисление максимального элемента

4. Вычисление минимального элемента

5. Вычисление скалярного произведения двух векторов-списков (с учетом возможного несовпадения размерностей).

6. Вычисление числа четных элементов

7. Проверка упорядоченности элементов по возрастанию

8. Вычисление среднего арифметического элементов

9. Вычисление числа вхождения 1-го элемента

10. Лексикографическое сравнение 2 списков

11. Вычисление позиции максимального элемента в списке

12. Вычисление позиции минимального элемента в списке

13. Проверка списка на арифметическую прогрессию

14. Проверка списка на геометрическую прогрессию

15. Вычисление позиции первого отрицательного элемента в списке

16. Вычисление суммы двух векторов-списков (с учетом возможного несовпадения размерностей)

17. Слияние двух упорядоченных списков

18. Разделение списка на два по принципу четности элементов

19. Разделение списка на два относительно первого элемента (по принципу "больше-меньше")

20. Разделение списка на два по порядковому принципу (первый-второй)

Реализацию всех предикатов поместите в файл `task1.pl`.

## Часть 2: Реляционное представление предметной области

Представьте себе, что вам необходимо представить информацию о студентах (которые учатся в группах 101..104) и их оценках за сессию. Предикаты Пролога

позволяют нам организовать представление данных в виде \*таблиц\*, или \*отношений\* - поэтому такое представление называется \*реляционным\*.

Для одной и той же предметной области представления могут быть разными. Примеры четырых разных представлений данных о студентах приведены в файлах `one.pl`, `two.pl`, `three.pl` и `four.pl`. Вам необходимо использовать \*\*одно\*\* представление, в соответствии с формулой (N mod 4)+1.

Для данного представления вам нужно выполнить одно из заданий ниже (каждое задание включает в себя три подзадания):

1. Вариант 1

- Получить таблицу групп и средний балл по каждой из групп

- Для каждого предмета получить список студентов, не сдавших экзамен (grade=2)

- Найти количество не сдавших студентов в каждой из групп

2. Вариант 2

- Напечатать средний балл для каждого предмета

- Для каждой группы, найти количество не сдавших студентов

- Найти количество не сдавших студентов для каждого из предметов

3. Вариант 3

- Для каждого студента, найти средний балл, и сдал ли он экзамены или нет

- Для каждого предмета, найти количество не сдавших студентов

- Для каждой группы, найти студента (студентов) с максимальным средним баллом

Для решения задачи вам скорее всего понадобится использовать предикаты `findall` (или `bagof`/`setof`) для поиска списка всех решений некоторого

предиката.

Решение разместите в файле `task2.pl`.

## Оформление отчета

Отчет необходимо оформить в формате Markdown, заполнив все необходимые пункты в файле [REPORT.md](REPORT.md). Оценка в табличке в начале отчета проставляется

преподавателем после проверки.

Удачной работы!