



МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО МГТУ «СТАНКИН»)

Институт
информационных технологий

Кафедра
информационных систем

Отчет по лабораторной работе №5

по дисциплине **«Интеллектуальные и экспертные системы»**
на тему: «Работа со списками в среде Prolog»

Студент
группа ИДБ–22–06

Мустафаева П.М.

подпись

Руководитель
старший преподаватель

Быстрикова В. А.

подпись

ВВЕДЕНИЕ

Целью работы является изучение возможностей представлений и обработки списков в программах на языке Пролог с использованием рекурсии.

ХОД РАБОТЫ

ЗАДАНИЕ 1

Вариант 11.

1. Реализовать основные операции со списками (проверка принадлежности конкретного элементу к списку, добавление элемента в начало списка, удаление элемента из списка, соединение двух списков в один).

2. Согласно своему варианту, написать программу вычисления предиката: Нахождение последнего элемента списка.

3. Множества заданы в виде списков. Реализовать операцию исключения из списка повторяющихся элементов.

4. Согласно варианту, написать программу вычисления предиката: Пересечение двух множеств.

Листинг 1 – Программный код

```
% Вывод списка на экран
вывести_список([]) :- !.
вывести_список(Список) :-
    Список = [Голова|Хвост], /* разделить список */
    write(Голова),           /* напечатать голову */
    write(' '),              /* напечатать пробел */
    вывести_список(Хвост). /* повторить для хвоста */

%удаление первого элемента
удалить([],_,[]):-!.
удалить(Список,Элемент,Результат):-
    Список=[Голова|Хвост], /* разделить список */
    Голова = Элемент, /* если удаляемый элемент – голова */
    Результат=Хвост, !. /* вернуть хвост списка */
удалить(Список,Элемент,Результат):-
    Список=[Голова|Хвост], /* а иначе удалить */
    удалить(Хвост,Элемент,Результат1), /* элемент из хвоста */
    Результат = [Голова|Результат1]. /* собрать список */

% Удаление всех вхождений элемента из списка
удалить_всех(Список, _, Результат) :-
    Список = [],
    Результат = [].
удалить_всех(Список, Элемент, Результат) :-
    Список = [Голова|Хвост],
    Голова = Элемент,
```

```

        удалить_всех(Хвост, Элемент, Результат), !.
удаить_всех(Список, Элемент, Результат) :-
    Список = [Голова|Хвост],
    удалить_всех(Хвост, Элемент, ХвостРезультата),
    Результат = [Голова|ХвостРезультата].

% Проверка принадлежности элемента к списку
принадлежит(Список, Элемент) :-
    Список = [Голова|_],
    Голова = Элемент, !.
принадлежит(Список, Элемент) :-
    Список = [_|Хвост],
    принадлежит(Хвост, Элемент).

% Добавление элемента в начало списка
добавить_в_начало(Элемент, Список, Результат) :-
    Результат = [Элемент|Список].

% Соединение двух списков
соединить(Список1, Список2, Результат) :-
    Список2 = [],
    Результат = Список1.
соединить(Список1, Список2, Результат) :-
    Список1 = [],
    Результат = Список2.
соединить(Список1, Список2, Результат) :-
    Список1 = [Голова|Хвост],
    соединить(Хвост, Список2, Временный),
    Результат = [Голова|Временный].

% Нахождение последнего элемента списка
последний_элемент(Список, Последний) :-
    Список = [Последний], !.
последний_элемент(Список, Последний) :-
    Список = [_|Хвост],
    последний_элемент(Хвост, Последний).

%Пересечение двух множеств
% Проверка принадлежности элемента списку
принадлежит(Список, Элемент) :-
    Список = [Голова|Хвост],
    (Голова = Элемент; принадлежит(Хвост, Элемент)).

% Проверка непринадлежности элемента списку
не_принадлежит([], _) :- !.
не_принадлежит(Список, Элемент) :-
    Список = [Голова|Хвост],

```

```
Элемент \== Голова,  
не_принадлежит(Хвост, Элемент).
```

```
пересечение_множеств([], _, []) :- !.  
пересечение_множеств(Список1, Список2, Результат) :-  
    Список1 = [Элемент|Остаток],  
    принадлежит(Список2, Элемент),  
    пересечение_множеств(Остаток, Список2, ВременныйРезультат),  
    Результат = [Элемент|ВременныйРезультат], !.  
пересечение_множеств(Список1, Список2, Результат) :-  
    Список1 = [Элемент|Остаток],  
    не_принадлежит(Список2, Элемент),  
    пересечение_множеств(Остаток, Список2, Результат), !.
```

Результат ответа на вопросы представлен на рис. 1-8.

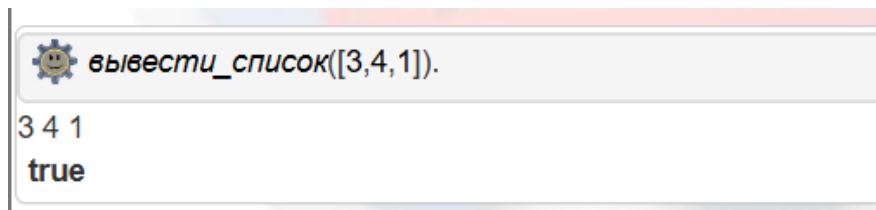


Рис. 1 Вывод списка

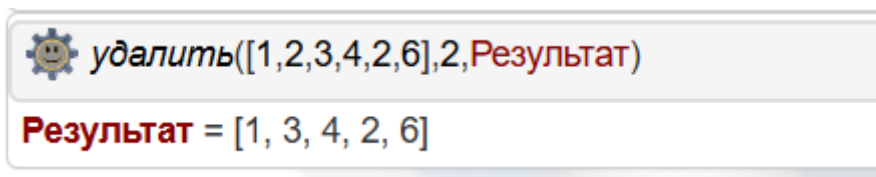


Рис. 2 Удаление первого найденного элемента

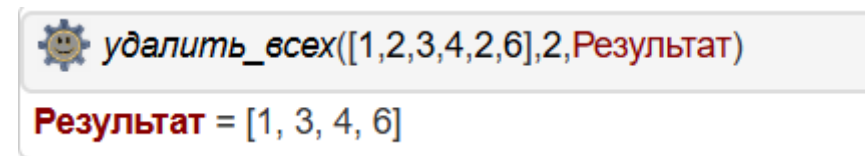


Рис. 3 Удаление всех вхождений элемента в список

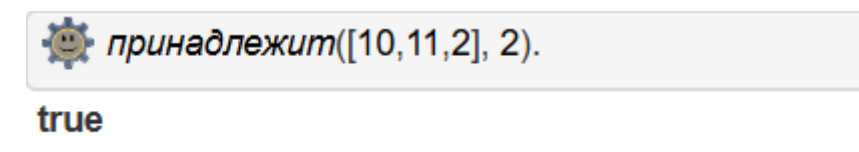


Рис. 4 Проверка принадлежности элемента к списку

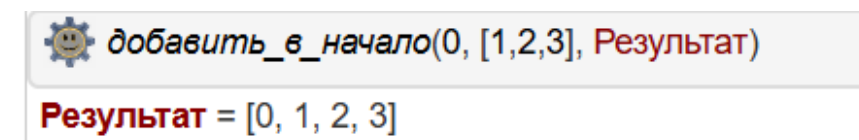



Рис. 5 Добавление элемента в начало списка

 `соединить([1,2,3], [4,5,6], Результат).`


Результат = [1, 2, 3, 4, 5, 6]

Рис. 6 Соединений двух списков

 `последний_элемент([1,2,3], Последний).`

Последний = 3

Рис. 7 Нахождение последнего элемента в списке

 `пересечение_множеств([1,2,3,4,5],[1,4,5,7,8,9],Результат)`

Результат = [1, 4, 5]

Рис. 8 Пересечение двух множеств

ЗАДАНИЕ 2

В базе знаний (см. лабораторную работу 3), представить информацию (не менее одного предиката) с использованием списков (пересечение списков).

Листинг 2 – Программный код

животное('Тигр', 'Млекопитающие', 'Занесен').

животное('Аист', 'Птицы', 'Занесен').

животное('Крыса', 'Млекопитающие', 'Не занесен').

животное('Окунь', 'Рыбы', 'Не занесен').

животное('Выхухоль', 'Млекопитающие', 'Занесен').

заповедник('Кавказский заповедник', 'Адыгея', '2803 км2').

заповедник('Остров Врангеля', 'Чукотский АО', '23017 км2').

заповедник('Астраханский заповедник', 'Астраханская область', '679 км2').

численность('Кавказский заповедник', 'Тигр', 25).

численность('Кавказский заповедник', 'Аист', 56).

численность('Кавказский заповедник', 'Выхухоль', 15).

численность('Остров Врангеля', 'Крыса', 29).

численность('Остров Врангеля', 'Окунь', 58).

численность('Остров Врангеля', 'Выхухоль', 30).

численность('Астраханский заповедник', 'Тигр', 5).

численность('Астраханский заповедник', 'Аист', 70).

численность('Астраханский заповедник', 'Окунь', 40).

% Предикат заповедник_с_животными(НазваниеЗаповедника, СписокЖивотных)

заповедник_с_животными('Кавказский заповедник', ['Тигр', 'Аист', 'Выхухоль']).

заповедник_с_животными('Остров Врангеля', ['Крыса', 'Окунь', 'Выхухоль']).

```

заповедник_с_животными('Астраханский заповедник', ['Тигр','Аист','Окунь']).

% Проверка: все элементы из первого списка есть во втором списке
все_принадлежат(ПервыйСписок, _ВторойСписок) :-
    ПервыйСписок = [], !.
все_принадлежат(ПервыйСписок, ВторойСписок) :-
    ПервыйСписок = [Голова|Хвост],
    принадлежит(ВторойСписок, Голова),
    все_принадлежат(Хвост, ВторойСписок).

% Принадлежность элемента списку
принадлежит(Список, Элемент) :-
    Список \== [],
    Список = [Голова|Хвост],
    (Голова = Элемент ; принадлежит(Хвост, Элемент)).

% Основной предикат поиска заповедников по списку животных
поиск_заповедника(ИскомыеЖивотные, НазваниеЗаповедника) :-
    заповедник_с_животными(НазваниеЗаповедника, ЖивотныеЗаповедника),
    все_принадлежат(ИскомыеЖивотные, ЖивотныеЗаповедника).

```

Результаты ответов на вопросы представлены на рис. 9.

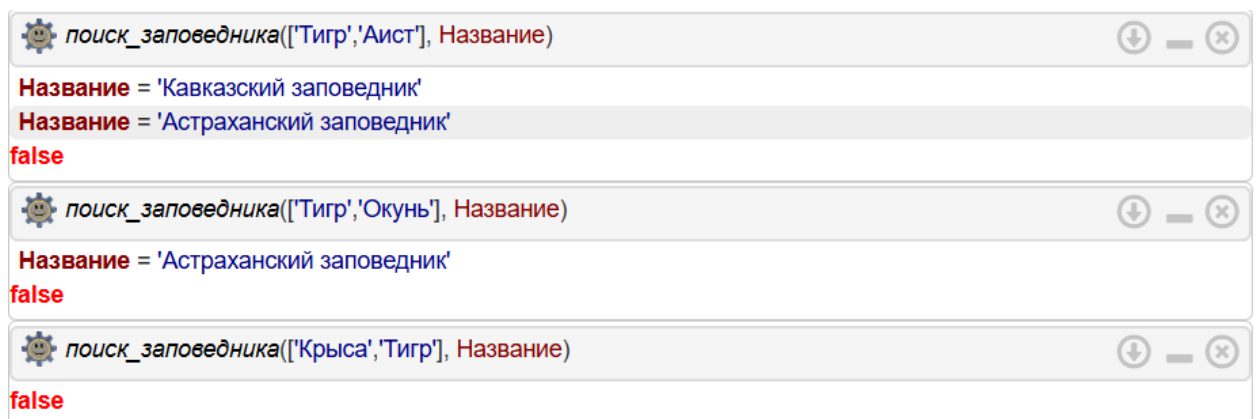


Рис. 9 Поиск пересекающихся животных в заповедниках

ВЫВОД

В ходе лабораторной работы были изучены и реализованы основные операции над списками в языке Prolog, включая проверку принадлежности элемента к списку, добавление и удаление элементов, а также объединение списков. Была разработана программа для нахождения последнего элемента списка с использованием рекурсии. Также выполнена задача обработки множеств, представленных в виде списков, с устранением повторяющихся элементов. В рамках работы успешно реализовано вычисление пересечения двух множеств и применено для индивидуальной базы знаний.