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 №1

по дисциплине «**Интеллектуальные и экспертные системы**»
на тему: Методы поиска решений интеллектуальных задач: основы логического программирования в среде SWI-Prolog

Студент _____ **Музафаров К.Р.**
группа ИДБ–21–06 подпись

Преподаватель _____ **Перепелкина Ю.В.**
подпись

Москва 2023 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Разработка SWI-Prolog проводится в университете города Амстердам (Нидерланды) с 1987 года. Его создателем и основным разработчиком является Jan Wielemaker. Название SWI – это аббревиатура от голландского Sociaal Wetenschappelijke Informatica («Social Science Informatics»), первоначального названия научной группы университета, в которой работает Jan Wielemaker. Сейчас название этой группы сменилось на новое название HCS (Human Computer Studies). SWI-Prolog, почти как и все реализации, в основном следует знаменитой "эдинбургской версии", но содержит частично реализованные особенности ISO Prolog. К числу основных особенностей последних версий SWI-Prolog следует отнести:

- Богатые библиотеки предикатов.
- Возможность написания логических модулей для веб-приложений.
- Поддержка GUI для XPCE и PreEmacs.
- Встроенная командная строка.
- Работа с файлами.

SWI-Prolog содержит развитые средства разработчика, включая интегрированную среду разработки (англ. Integrated Development Environment - IDE) с графическим отладчиком и профилировщиком, и обширную документацию. Кроме базовых функций языка, платформа реализует многопоточность, юнит-тестирование, GUI, интерфейс к языку программирования Java, ODBC и т. д. SWI-Prolog имеет встроенный собственный веб-сервер и работает на Unix, Windows, Macintosh и Linux платформах.

Задание на данную лабораторную работу:

1. Перейти на платформу SWI-Prolog по ссылке https://swish.swi-prolog.org/ и изучить ее интерфейс.

2. Запустить предоставленные листинги кода программ (см. Примеры 1-4) и проанализировать их работу.

ГЛАВА 1. ПРАКТИЧЕСКАЯ ЧАСТЬ
ПРИМЕРЫ РЕАЛИЗАЦИИ ЛОГИЧЕСКИХ ПРОГРАММ НА ЯЗЫКЕ PROLOG

Пример 1. Работа с базой знаний SWI Prolog (экспертная система на базе правил)

Имеется база знаний, которая содержит правила вида отдыхать(имя, город). Задавая системе вопросы в форме запросов, можно получить ответ, в каком городе кто отдыхал.

Листинг кода базы знаний:

```
rest('sasha', 'urmala').
rest('anna', 'antalia').
rest('dima', 'urmala').
rest('dima', 'baku').
```

Запрос системе – «Где отдыхают и Саша и Дима?»: rest('sasha', X), rest('dima', X).

Ответ системы на запрос:

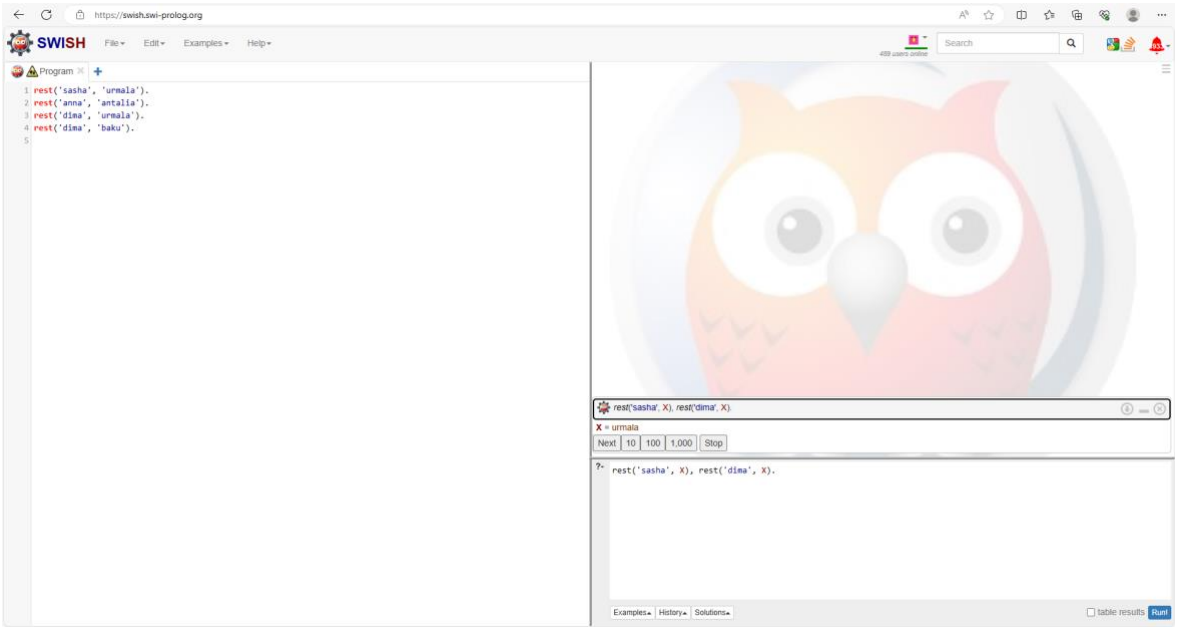

Рис. 1.1. Ответ на вопрос - «Где отдыхают и Саша и Дима?»

Запрос системе – «Кто отдыхает и в Юрмале, и в Баку?»: rest(Y, 'urmala'), rest(Y, 'baku').

Ответ системы на запрос:

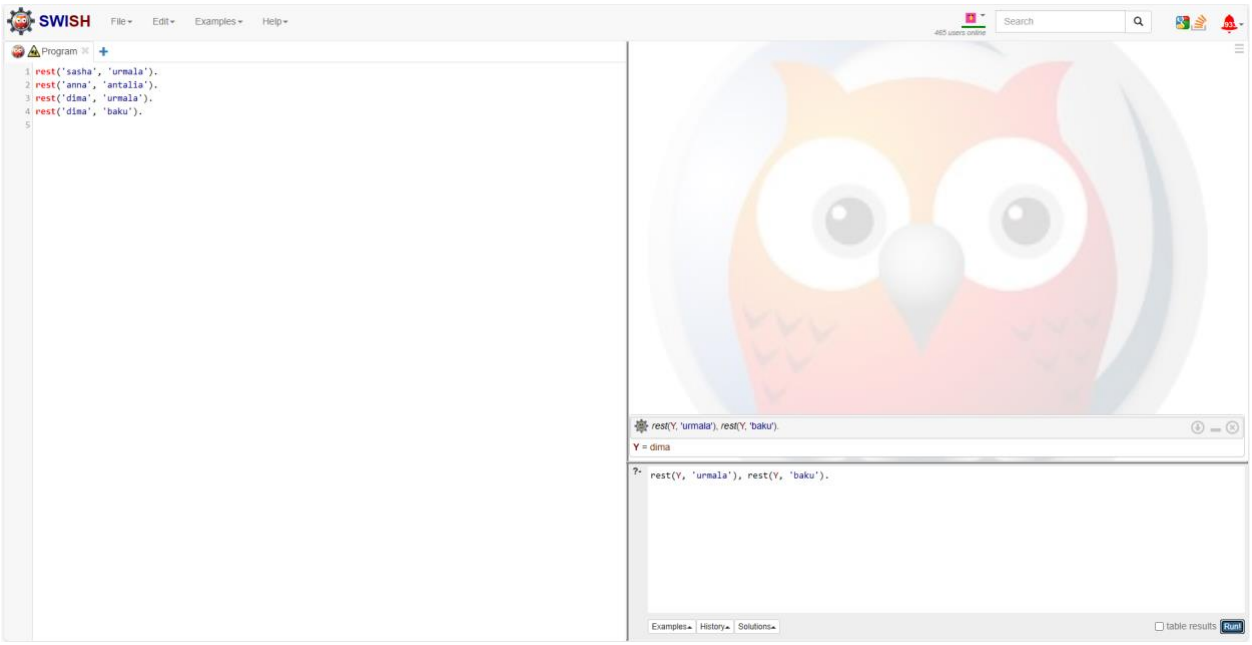

Рис. 1.2. Ответ на вопрос - «Кто отдыхает и в Юрмале, и в Баку?»

Пример 2. Работа с экспертной системой (предикаты для списков) SWI Prolog

Листинг кода базы знаний:

```
cond(1, 'кормит детенышей молоком').
cond(2, 'имеет перья').
cond(3, 'плавает').
cond(4, 'ест мясо').
cond(5, 'имеет копыта').
cond(6, 'летает').
cond(7, 'откладывает яйца').
cond(8, 'имеет шерсть').
cond(9, 'имеет полосы').
cond(10, 'имеет пятна').
cond(11, 'имеет черно-белую окраску').

rule('гепард', [1,4,8,10]).
rule('тигр', [1,4,8,9]).
rule('зебра', [1,5,8,9,11]).
rule('пингвин', [2,3,11]).
rule('орел', [2,6]).
```

```
rule('кит', [1,3,11]).
```

Тело кода экспертной системы:

```
process(Answer, Answers, AnswersNew, Q) :-
  member(Answer, [1, 'да', 'ага', 'yes', 'yo', 'yep', 'true', 'sure']),
  append(Answers, [Q], AnswersNew), !.

process(_, AnswersNew, AnswersNew, _).
  check(Answers, _) :-
  rule(X, Answers),
  nl,
  write('это '),
  write(X),
  nl,
  halt.

check(Answers, Q) :-
  Q1 is Q + 1,
  run(Q1, Answers), !.

run(Q, Answers) :-
  cond(Q, Question),
  write(Question), nl,
  read(Answer),
  process(Answer, Answers, AnswersNew, Q),
  check(AnswersNew, Q).
```

Делаем запросы системе (вопросы вводятся в блок окна внизу справа). Запрос имеет шаблон вида cond(Y,X), где Y – номер признака, характеризующего животное и указанный в базе знаний.

Ответ системы на запрос cond(9,X).

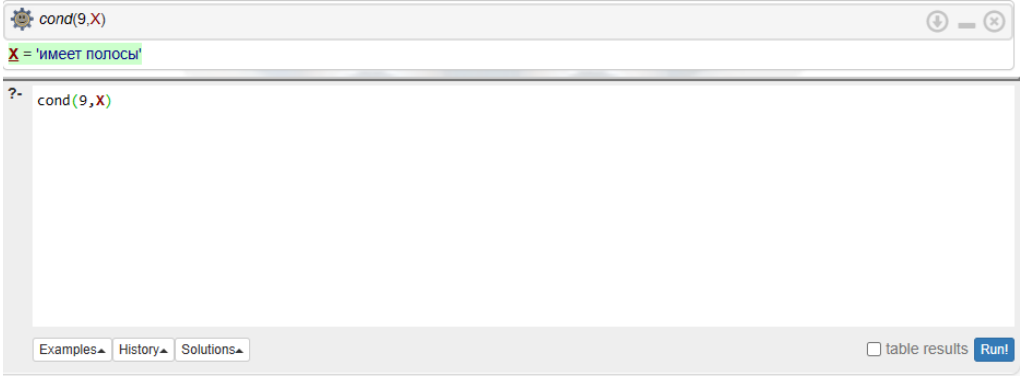

Рис. 1.3. Ответ системы на запрос cond(9,X)

Пример 3. Поиск в глубину и ширину (задача логистики на базе теории графов)

Рассматриваемые типы задач:

1) поиск всех путей из Пункта 1 в Пункт 2, проходящих через Пункт 3;

2) анализ наличие пути между пунктами, с учетом направления передвижения (ориентированный граф).

Листинг кода базы знаний:

```
edge(moskva, nnovgorod).
edge(moskva, samara).
edge(nnovgorod, perm).
edge(nnovgorod, ekaterinburg).
edge(perm, ekaterinburg).
edge(ekaterinburg, samara).
edge(samara,ekaterinburg).
edge(ekaterinburg, novosib).
edge(samara, novosib).
```

Тело кода экспертной системы:

```
:- if(current_predicate(use_rendering/1)).
    :- use_rendering(table, [header(go('откуда', 'куда'))]).
:- endif.

/** <examples>
?- find_way(moskva, novosib, perm, Way).
*/

edge(moskva, nnovgorod).
edge(moskva, samara).
edge(nnovgorod, perm).
edge(nnovgorod, ekaterinburg).
edge(perm, ekaterinburg).
edge(ekaterinburg, samara).
edge(samara,ekaterinburg).
edge(ekaterinburg, novosib).
edge(samara, novosib).

find_way(From, To, Across, Way) :-
    find_way(From, To, Across, no, [], Way0),
    reverse(Way0, Way).
```

```
find_way(Point, To, _, yes, Track, [go(Point, To) | Track]) :-
   edge(Point, To).
find_way(From, To, Across, IsAcross, Track, Way) :-
   edge(From, Point),
   not( member(go(From, Point), Track) ),
   ( Point = Across, IsAcross1 = yes ; Point \= Across, IsAcross1 = IsAcross ),
   find_way(Point, To, Across, IsAcross1, [go(From, Point) | Track], Way).
```

Ответ системы на запрос (поиск пути из Москвы в Новосибирск через Пермь): find_way(moskva, novosib, perm, Way)

Рис. 1.4. Ответ системы на запрос (поиск пути из Москвы в Новосибирск через Пермь)

Ответ системы на запрос (проверка наличия пути между Нижним Новгородом и Самарой): edge(nnovgorod, samara).

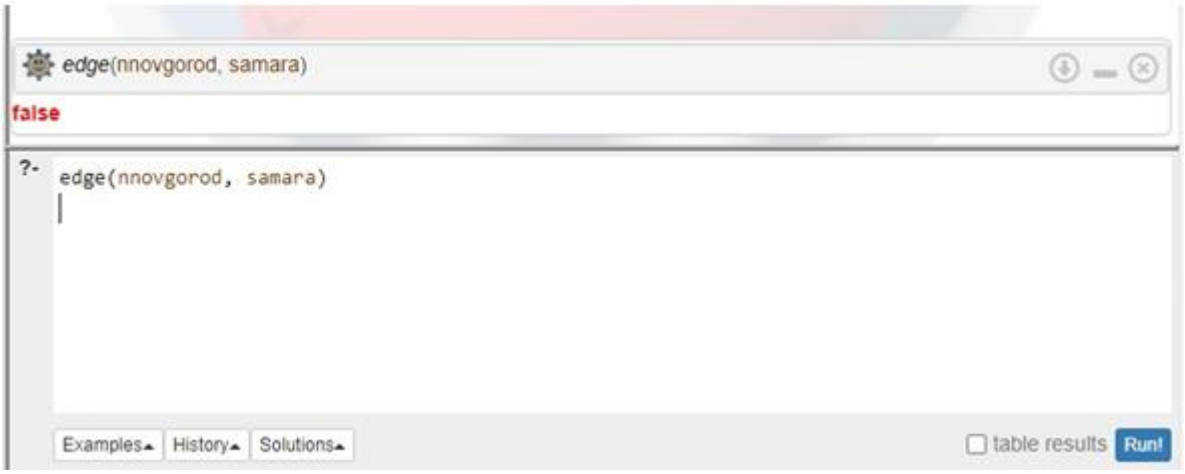

Рис. 1.5. Ответ системы на запрос (проверка наличия пути между Нижним Новгородом и Самарой)

Пример 4. Работа с базой знаний SWI Prolog (поиск в предметной области, определение значений переменных в динамическом режиме)

Постановка задачи:

Имеется набор объектов разных категорий - люди, собаки, кошки, мыши. Для каждой категории указываются определенные представители (по именам, кличкам и признакам).

В начале листинга программы описываются категории и указываются имена их определённых представителей (связки идентификации).

Полный листинг программы:

```
% Условие задачи: Сократ — человек. Все люди и животные смертны.
% Найти: Кто смертен? Кто смертен из людей? Какие смертные
животные не старше 5 лет?

% люди
human('Сократ').
human('Платон').

% собаки
dog('Шарик').
dog('Тоша').

% кошки
cat('Сима').

% мыши
mouse('Джерри').

% животное - это или собака, или кошка, или мышь.
animal(X) :-
  dog(X);
  cat(X);
  mouse(X).

% возраст наших живых существ
age('Платон', 80).
age('Сократ', 70).
age('Шарик', 10).
age('Тоша', 7).
age('Сима', 5).
age('Джерри', 3).
```

```
% смертный -> либо человек, либо животное.
mortal(Someone) :-
  human(Someone);
  animal(Someone).
```

Ответ системы на запрос «Кто является смертным из людей?» (поиск по категории «люди»): mortal(Who), human(Who).

Рис. 1.6. Ответ системы на запрос «Кто является смертным из людей?» (поиск по категории «люди»)

Ответ системы на более сложный запрос «Смертные животные не старше 5 лет»): mortal(Who), animal(Who), age(Who, Age), Age=<5.

Рис. 1.7. Ответ системы на более сложный запрос «Смертные животные не старше 5 лет»)

САМОСТОЯТЕЛЬНАЯ РАБОТА

Задание: необходимо создать экспертную систему по заданной предметной области (см. перечень областей в п.3.2).

Требования к выполнению задания:

1.Экспертная система должна быть выполнена в среде SWI-Prolog.

2.Задания индивидуальные (каждый студент выбирает свою экспертную область и сообщает выбранную тематику преподавателю).

3.Экспертная система должна иметь не менее 20-ти объектов (правил/элементов) для логических связок.

Тема экспертной системы – **Автомобильное оборудование**.

Листинг кода:

```
cond(1,  'Однорядный L Одноцилиндровый 2').
cond(2,  'Однорядный L Одноцилиндровый 3').
cond(3,  'Однорядный L Одноцилиндровый 4').
cond(4,  'Однорядный L Одноцилиндровый 5').
cond(5,  'Однорядный L Одноцилиндровый 6').
cond(6,  'Однорядный L Одноцилиндровый 8').
cond(7,  'Однорядный L Одноцилиндровый 9').
cond(8,  'Однорядный L Одноцилиндровый 10').


cond(11, 'Оппозитный — B 2').
cond(12, 'Оппозитный — B 4').
cond(13, 'Оппозитный — B 6').
cond(14, 'Оппозитный — B 8').
cond(15, 'Оппозитный — B 10').
cond(16, 'Оппозитный — B 12').
cond(17, 'Оппозитный — B 16 H').


cond(18, 'V-образный 2 ').
cond(19, 'V-образный 4 ').
cond(20, 'V-образный 5 ').
cond(21, 'V-образный 6 ').
cond(22, 'V-образный 8 ').
cond(23, 'V-образный 10 ').


cond(24, 'W-образный 8 ').
```

```
cond(25, 'W-образный 12 ').
cond(26, 'W-образный 16 ').
cond(27, 'W-образный 18').

rule("16C20",[1, 11, 18, 24]).
rule("17C21",[2, 12, 19, 25]).
rule("18C22",[3,13, 20, 26]).
rule("19C23",[4, 14, 21, 26]).
rule("11C20",[5, 15, 22, 24]).
rule("12C29",[6, 16, 23, 25]).
rule("13C27",[7, 17, 18, 26]).
rule("14C23",[8, 11, 19, 27]).
rule("16C24",[1, 11, 18, 25]).

member(X, [X|_]).
member(X, [_|T]) :- member(X, T).

find_rules(Conditions, Rules) :-
   findall(Rule, (rule(Rule, RuleConditions),
         check_conditions(Conditions, RuleConditions)),
         Rules).

map_conditions([], []).
map_conditions([C|Conditions], [N|Numbers]) :-
   cond(N, C),
   map_conditions(Conditions, Numbers).

check_conditions([], _).
check_conditions([C|Conditions], RuleConditions) :-
   member(C, RuleConditions),
   check_conditions(Conditions, RuleConditions).

find_combustion(Conditions, Rules):-
   map_conditions(Conditions, Numbers),
   find_rules(Numbers, Rules).
```

Ответ системы на вопрос “Автомобильный двигатель который является «Однорядный L Одноцилиндровый 2» и «W-образный 8»”: Conds = ['Однорядный L Одноцилиндровый 2', 'W-образный 8 '],

find_combustion(Conds, Rules)

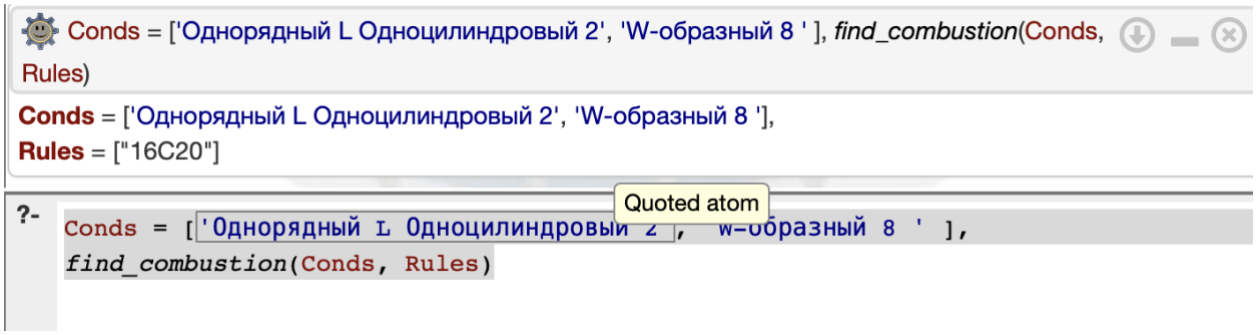

Рис. 1.8. Ответ системы на вопрос “Автомобильный двигатель который является «Однорядный L Одноцилиндровый 2» и «W-образный 8»”

Ответ системы на вопрос “Автомобильный двигатель который является «Однорядный L Одноцилиндровый 2»”: Conds = ['Однорядный L Одноцилиндровый 2'],

find_combustion(Conds, Rules)

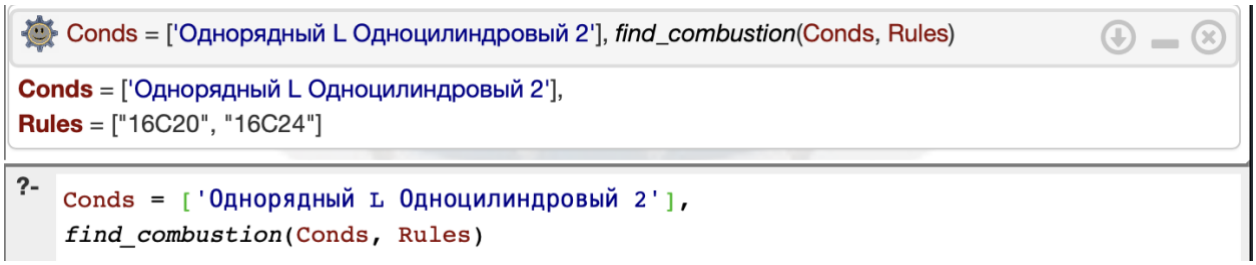

Рис. 1.9. Ответ системы на вопрос “Автомобильный двигатель который является «Однорядный L Одноцилиндровый 2»

ЗАКЛЮЧЕНИЕ

Были рассмотрены основные конструкции рекурсивно-логического программирования: термы и утверждения, унификацию термов на языке Prolog и изучины приемы работы с интерпретатором SWI Prolog; изучены элементы интерфейса интерпретатора SWI Prolog: окно запросов; встроенный редактор текстов; были приобретены практические навыки создания, отладки и выполнения простых программ в среде SWI Prolog.

СПИСОК ДОПОЛНИТЕЛЬНЫХ ИСТОЧНИКОВ

1. Лысачев М. Н. Искусственный интеллект. Анализ, тренды, мировой опыт / М. Н. Лысачев, А. Н. Прохоров; научный редактор Д. А. Ларионов. – Корпоративное издание. – Москва; Белгород: КОНСТАНТА-принт, 2023. – 460 с. : ил., табл. ISBN 978-5-6048180-7-7, Электронное издание (ссылка на Яндекс-диск https://disk.yandex.ru/i/d-ky8jRcWqHR6g).

2. Ссылка сайта SWI Prolog: https://swish.swi-prolog.org/.