**Практическая работа 1**

**Разработка экспертной системы на языке логического программирования – Пролог**

**Цель работы:**

1. Ознакомление с исчислением предикатов;

2. Изучить основы и синтаксис языка Пролог;

3. Выработать навыки работы с реализацией языка Prolog - SWI- Prolog.

**Задание:**

Разработать экспертную систему на языке Пролог для выбора подходящей системы безопасности. Экспертная система должна по запросам пользователя выбрать подходящее решение для обеспечения безопасности телекоммуникационной сети предприятия.

Должны быть реализованы следующие составные запросы:

- Выбор решений, имеющие рейтинг не ниже 4 и сертификат ФСТЭК;

- Выбор open-sourсe решений;

- Выбор решений, не имеющее антивирусную защиту;

- Выбор решений, которые можно купить меньше, чем за 800 $.

**Теоретические сведения. Язык логического программирования – Пролог**

Пролог (англ. Prolog) — язык логического программирования, основанный на исчислении предикатов. Prolog является декларативным языком программирования: логика программы выражается в терминах отношений, представленных в виде фактов и правил. SWI-Prolog - стабильная свободная реализация языка Prolog. Программа на языке SWI-Prolog имеет расширение .pl.

Программирование на языке Пролог состоит из следующих этапов:

1) объявления некоторых фактов об объектах и отношениях между ними;

2) определения некоторых правил об объектах и отношениях между ними;

3) формулировки запросов об объектах и отношениях между ними.

Факт — это фраза без условий, утверждение о том, что соблюдается некоторое отношение. Факт записывается как имя предиката, за которым следует заключенный в скобки список аргументов. Имя предиката и объекта должно начинаться с маленькой буквы и может содержать латинские буквы, кириллицу, цифры и символ подчеркивания. Если значение аргумента предиката или его имя начинается с заглавной буквы, то оно пишется в апострофах. Кириллица используется наравне с латинскими буквами. Обычно предикатам дают такие имена, чтобы они отражали смысл отношения. Два предиката могут иметь одинаковые имена, если они имеют различное число аргументов (арность). Каждый факт должен заканчиваться точкой:

предикат (аргумент\_1, аргумент\_2, …, аргумент\_n).

Правило — это факт, истинность которого зависит от истинности других фактов. Правила описывают связи между предикатами. Состоит из головы и тела, разделенных знаком :- , который читается как «если» и соответствует операции импликации. Голова правила — это предикат, а тело правила — последовательность предикатов, разделенных запятыми. B:-A. соответствует импликации A→B («ЕСЛИ A , ТО B»). Правило должно заканчиваться точкой, запятая в теле правила означает конъюнкцию (логическое И), точка с запятой ; обозначает дизъюнкцию:

предикат\_В (аргумент\_B, X) :- предикат\_A (аргумент\_A, X).

Интуитивный смысл правила состоит в том, что цель, являющаяся головой, будет истинной, если Пролог сможет показать, что все выражения (подцели) в теле правила являются истинными.

Голова :- Подцель1, Подцель2, ... , ПодцельN

Факты и правила называются также утверждениями и образуют собой базу знаний.

Переменная - именованная область памяти, где может храниться значение. имена объектов (значения аргументов) имя отношения (имя предиката). Переменная в SWI/PROLOG обозначается как последовательность латинских букв, кириллицы и цифр, начинающаяся с заглавной буквы или символа подчеркивания \_. Переменная не имеет предопределенного типа данных и может быть связана со значением любого типа данных. Переменная, однажды получив свое значение при унификации в процессе работы программы, не может его изменить. Если переменная не связана со значением – она называется свободной переменной.

Унификация - процесс получения свободной переменной значения в результате сопоставления при логическом выводе в SWI-Prolog.

Запрос — это последовательность предикатов, разделенных запятыми или точкой с запятой и завершающаяся точкой. На естественном языке запятая соответствует союзу «и», а на языке математической логики обозначает конъюнкцию. Точка с запятой соответствует союзу «или» и обозначает дизъюнкцию. Предикат запроса называется целью. Простые запросы, не содержащие никаких переменных, допускают лишь два возможных ответа: "true" или "false". В случае ответа "true" говорят, что запрос завершился успехом, цель достигнута. Использование переменных в запросах позволяет задавать более сложные вопросы.

Для того чтобы инициировать вычисления, выполняется специальный запрос к базе знаний, на которые система логического программирования генерирует ответы «истина» и «ложь». Для обобщённых запросов с переменными в качестве аргументов созданная система Пролог выводит конкретные данные в подтверждение истинности обобщённых сведений и правил вывода. Иначе говоря, предикат можно определить, как функцию, отображающую множество произвольной природы в множество булевых значений {ложно, истинно}. Задача пролог–программы заключается в том, чтобы доказать, является ли заданное целевое утверждение следствием из имеющихся фактов и правил.

Составные объекты Составные объекты или структуры состоят из имени структуры-функтора и нескольких компонент структуры, которые могут быть любыми объектами, в том числе структурами.

Составной запрос — это несколько предикатов, соединенных знаками запятой (оператор «и»), точкой с запятой (оператор «или»), которые реализуют соответствующие логические связки. В конце целевого утверждения указывается точка.

Рекурсия. Важным методом программирования в Прологе является рекурсия. Рекурсивным называется определение функции (предиката) через эту же функцию (предикат). Пример рекурсивного вычисления факториала в SWI-Prolog показан в листинге 1.

Листинг 1 — Пример способа вычисления факториала

% Факт

factorial(1,1).

% Правило

factorial(N,F) :-N>1, N1 is N-1, factorial(N1, F1),F is N\*F1.

% Запрос

?- factorial(5,X). X = 120

Комментарий в строке программы начинается с символа % и заканчивается концом строки. Блок комментариев выделяется специальными скобками: /\* (начало) и \*/ (конец).

**Экспертная система на языке Пролог для выбора подходящей системы безопасности.**

В листинге 2 приведен пример программы на SWI-Prolog экспертной системы по выбору подходящей системы безопасности. Решения (системы) представлены с вымышленными названиями. Каждое решение имеет наличие/отсутствие сертификата ФСТЭК, свою цену, рейтинг, наличие/отсутствие opensource решений и наличие/отсутствие антивируса. В программе представлены некоторые правила (чтобы вывести все подходящие решения после вывода каждого решения, нужно нажать символ «;»):

- fstec\_filter\_and\_rate – выбор решений, имеющий рейтинг не ниже 4 и сертификат ФСТЭК ;

- with\_opensource – выбор opensource решения;

- use\_without\_antivirus – выбор решений, не имеющие антивирусную защиту;

- cost\_filter – выбор решений, которые дешевле 800$.

Листинг 2 — Код программы и примеры запуска

/\*

Экспертная система выбора решений для обеспечения

безопасности на языке SWI-Prolog

\*/

% Факты.

**system**(name('5G'),

fstec(false),

cost(1200),

rate(4.5),

opensource(true),

antivirus(true)).

**system**(name('CarGreen'),

fstec(true),

cost(500),

rate(4.2),

opensource(false),

antivirus(false)).

**system**(name('FiveGold'),

fstec(false),

cost(430),

rate(3.9),

opensource(true),

antivirus(false)).

**system**(name('Qwerty'),

fstec(true),

cost(999),

rate(3.1),

opensource(false),

antivirus(true)).

**system**(name('Rare'),

fstec(true),

cost(1000),

rate(5),

opensource(true),

antivirus(false)).

% Правила.

fstec\_filter\_and\_rate(Name):-

system(name(Name),fstec(X),\_,rate(Y),\_,\_),

X=true,

Y>4.

with\_opensource(Name):-

system(name(Name),\_,\_,\_,opensource(true),\_).

use\_without\_antivirus(Name):-

system(name(Name),\_,\_,\_,\_,antivirus(false)).

cost\_filter(Name):-

system(name(Name),\_,cost(X),\_,\_,\_),

X>800.

% Примеры запросов (пишутся в консоль)

%

% Запрос №1

% ?- fstec\_filter\_and\_rate(Name).

%

% Запрос №2

% ?- with\_opensource(Name).

%

% Запрос №3

% ?- use\_without\_antivirus(Name).

%

% Запрос №4

% ?- cost\_filter(Name).

Запустим полученную программу и посмотрим на результат, который показан на рисунке 1.

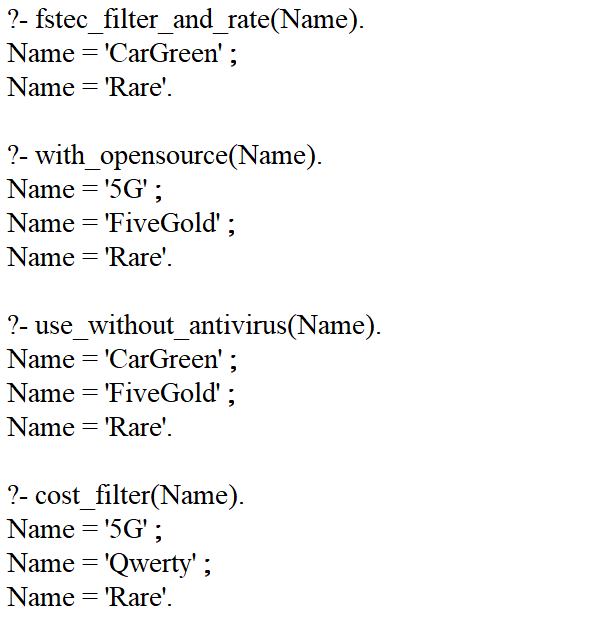


Рисунок 1 - Результат работы программы