**ЛАБОРАТОРНАЯ РАБОТА № 3**

**«РАЗРАБОТКА И ИССЛЕДОВАНИЕ ЭКСПЕРТНОЙ СИСТЕМЫ»**

**Цель работы**

Разработка экспертной системы продукционного типа на Прологе, исследование базовых принципов организации экспертных систем.

**Постановка задачи**

1. Изучить модели представления знаний по лекционному материалу и учебным пособиям [1,5,8 ].

2. Детально изучить организацию продукционной системы, механизмы прямого и обратного вывода, основы построения подсистемы объяснения, примеры реализации ЭС на Прологе [1,2,4,11].

3. Выполнить анализ предметной области ЭС в соответствии с вариантом задания:

a) выбрать задачу, решаемую ЭС, и определить цели системы: конечные, промежуточные и вспомогательные;

б) выделить подзадачи, которые следует решить для достижения цели;

в) разработать продукционную базу правил для решения выделенных подзадач.

4. Ознакомиться с примерами программных кодов, приведенных в приложении В, и, по аналогии, разработать ЭС продукционного типа для решения задачи в заданной предметной области.

5. Создать в среде программирования Пролог проект ЭС и выполнить его отладку.

6. Исследовать свойства разработанной системы. Получить протоколы работы системы при доказательстве различных целевых утверждений.

7. Зафиксировать результаты работы программы в виде экранных копий.

**Вариант 3**

Предметная область – Обучение

**Ход работы**

База знаний классифицирующей ЭС (файл nb.pl):

info:-

nl,

write('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),nl,

write('\* Экспертная система \*'),nl,

write('\* для определения будущей \*'),nl,

write('\* профессии \*'),nl,

write('\*-------------------------------------\*'),nl,

write('\* Отвечайте на вопросы: \*'),nl,

write('\* да, нет, почему \*'),nl,

write('\* Для объяснения решения \*'),nl,

write('\* введите цель \*'),nl,

write('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),nl,

write('Введите любой символ'),nl,

%Ожидание ввода литеры

get0(\_).

% тестовая база продукционных правил для определения будущей профессии

правило1 :: если [нравится(работать\_за\_компьютером), склад\_ума(аналитический)] то программист.

правило2 :: если [нравится(работать\_за\_компьютером), нравится(творческая\_работа), нравится(рисовать)] то цифровой\_художник.

правило3 :: если [нравится(творческая\_работа), нравится(рисовать)] то художник.

правило4 :: если [нравится(творческая\_работа), склад\_ума(аналитический), нравится(работать\_за\_компьютером), нравится(музыка)] то цифровой\_музыкант.

правило5 :: если [нравится(творческая\_работа), нравится(музыка)] то музыкант.

правило6 :: если [склад\_ума(гуманитарный), предпочитаете(изучать\_историю)] то историк.

правило7 :: если [склад\_ума(гуманитарный), предпочитаете(изучать\_право)] то юрист.

правило8 :: если [нравится(работать\_за\_компьютером), нравится(роботы), нравится(искусственный\_интеллект)] то робототехник.

правило9 :: если [склад\_ума(аналитический), нравится(работать\_с\_железом)] то робототехник.

правило10 :: если [нравится(работать\_с\_детьми), нравится(оказывать\_моральную\_поддержку)] то педагог\_психолог.

правило11 :: если [нравится(оказывать\_медицинскую\_помощь)] то врач.

правило12 :: если [нравится(работать\_с\_детьми), нравится(оказывать\_медицинскую\_помощь)] то врач\_педиатр.

правило13 :: если [нравится(работать\_с\_детьми), нравится(заниматься\_спортом)] то детский\_тренер.

правило14 :: если [нравится(заниматься\_спортом)] то спортсмен.

% гипотезы (цели)

h1 :: гипотеза(юрист).

h2 :: гипотеза(программист).

h3 :: гипотеза(педагог\_психолог).

h4 :: гипотеза(цифровой\_художник).

h5 :: гипотеза(цифровой\_музыкант).

h6 :: гипотеза(историк).

h7 :: гипотеза(врач).

h8 :: гипотеза(спортсмен).

h9 :: гипотеза(художник).

h10 :: гипотеза(музыкант).

h11 :: гипотеза(робототехник).

h12 :: гипотеза(врач\_педиатр).

h13 :: гипотеза(детский\_тренер).

% признаки, истинность которых можно выяснить у пользователя

q1 :: признак(склад\_ума(гуманитарный)).

q2 :: признак(склад\_ума(аналитический)).

q3 :: признак(нравится(решать\_логические\_задачи)).

q4 :: признак(нравится(работать\_за\_компьютером)).

q5 :: признак(нравится(творческая\_работа)).

q6 :: признак(нравится(рисовать)).

q7 :: признак(нравится(музыка)).

q8 :: признак(нравится(заниматься\_спортом)).

q9 :: признак(предпочитаете(изучать\_историю)).

q10 :: признак(предпочитаете(изучать\_право)).

q11 :: признак(нравится(работать\_с\_детьми)).

q12 :: признак(нравится(оказывать\_моральную\_поддержку)).

q13 :: признак(нравится(оказывать\_медицинскую\_помощь)).

q14 :: признак(нравится(роботы)).

q15 :: признак(нравится(искусственный\_интеллект)).

q16 :: признак(нравится(работать\_с\_железом)).

Интерпретатор классифицирующей ЭС (файл interp.pl):

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Интерпретатор (машина вывода) для ЭС продукционного типа

% Метод вывода: обратный вывод

% Вариант 2: интерпретатор обрабатывает правила, в которых

% предпосылки задаются в виде списка условий.

% Это позволяет в условной части правила, задавать произвольное

% количество условий.

% -------------------------------------------------------------------------------

% Примеры правил см. в загружаемой тестовой базе знаний - new\_anim.pl

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

:-dynamic

сообщено/2.

определить\_операторы:-

op(950, xfx, то),

op(960, fx, если),

op(970, xfx, '::').

:-определить\_операторы.

%============обратный вывод======================================================

% реализуется предикатом найти(S,Стек,Д),где S - список проверяемых гипотез,

% Стек - стек из имен доказываемых гипотез и правил (используется при ответе на

% вопросы "почему), Д - дерево вывода целевого утверждения (используется при отве-

% те на вопросы "как"). Предикат получает на вход список [Н] и Стек=[H] и в про-

% цессе обратного вывода строит дерево вывода Д.

% Предикат "найти" для доказательства отдельных гипотез из списка S

% использует предикат найти1(Н,Стек,Дерево).

%--------------------------------------------------------------------------------

% случай1:если цель Н была подтверждена пользователем,

% то дерево вывода Д=сообщено(Н).

найти1(H,Стек,сообщено(H)):-сообщено(H,да).

найти1(H,Стек,сообщено(H)):-запрашиваемая(H),

not(сообщено(H,\_)),спроси(H,Стек).

% случай2:если цель Н подтверждается фактом, уже известным системе,

% то дерево вывода Д=Факт :: H

найти1(H,Стек,Факт :: H):-Факт :: H.

% случай3: если цель Н соответствует следствию одного из

% правил -> Правило :: если H1 то H

% и если Д1 дерево вывода для подцели Н1,

% то Д= Правило :: если Д1 то H и добавить № правила в Стек

найти1(H,Стек,Правило :: если Д1 то H):-

Правило :: если H1 то H,

найти(H1,[Правило | Стек],Д1).

% случай4: если доказывается конъюнкция гипотез, заданная списком гипотез,

% то найти доказательство первой гипотезы Н1 из списка

% с помощью найти1(H1,Стек,Дерево1), а затем найти доказательство оставшихся

% гипотез Т с помощью найти(T,Стек,Дерево) и

% объединить деревья вывода в общий список [Дерево1 | Дерево].

найти([],Стек,Дерево):-Дерево=[].

найти([H1|T],Стек,[Дерево1 | Дерево]):-

найти1(H1,Стек,Дерево1),найти(T,Стек,Дерево).

% проверка: является ли гипотеза признаком, значение которого можно спросить

запрашиваемая(H):-Факт :: признак(H).

%=========вывод вопросов и обработка ответов "да, нет, почему" ==================

%вывод вопроса и ввод ответа

спроси(H,Стек):-write(H),write('?'),nl,

read(O),ответ(H,O,Стек).

%обработка ответов: да, нет

ответ(H,да,Стек):-assert(сообщено(H,да)),!.

ответ(H,нет,Стек):-assert(сообщено(H,нет)),!,fail.

%обработка ответов - "почему"

% случай1: стек целей пустой

ответ(H,почему,[]):-!,write(' Вы задаете слишком много вопросов'),nl,

спроси(H,[]).

%случай2: в стеке осталась только первая введенная цель, т.е доказываемая гипотеза

ответ(P,почему,[H]):-!,write('моя гипотеза: '),

write(H),nl,спроси(P,[]).

%случай3: вывод заключения и номера правила для доказываемой текущей подцели Н

ответ(H,почему,[Правило | Стек]):-!,

Правило :: если H1 то H2,

write('пытаюсь доказать '),

write(H2),nl,

write('с помощью правила: '),

write(Правило),nl,

спроси(H,Стек).

%неправильный ответ: повторяем вопрос

ответ(H,\_,Стек):-write(' правильный ответ: да, нет, почему'),nl,

спроси(H,Стек).

%==============обработка ответов на вопросы "как?"===============================

% предикат как(H,Д)- выполняет поиск подцели Н в построенном

% с помощью предиката "найти" дереве вывода Д и отображает соответствующий

% фрагмент дерева вывода, объясняя, как было получено доказательство Н.

% Дерево вывода Д представляет собой последовательность вложенных правил

% в виде списка, например:

% [правило5::если[правило1::если[сообщено(имеет(шерсть))]то млекопитающее,

% сообщено(ест\_мясо)]то хищник,...]

%--------------------------------------------------------------------------------

% поиск целевого утверждения Н в дереве

как(H,Дерево):-как1(H,Дерево),!.

% вывод сообщения, если Н не найдено

как(H,\_):-write(H),tab(2),write('не доказано'),nl.

% случай1: если Н сообщено пользователем,

% то вывести "Н было введено"

как1(H,\_):-сообщено(H,\_),!,

write(H),write('было введено'),nl.

% случай2: если дерево вывода Д представлено фактом, подтверждающим Н

как1(H,Факт :: H):-!,

write(H), write( 'является фактом'), write(Факт),nl.

% случай3: если дерево вывода Д - правило в заключение, которого есть Н,

% то отобразить это правило

как1(H,[Правило :: если \_ то H]):-!,

write(H),write(' было доказано с помощью'),nl,

Правило :: если H1 то H,

отобрази\_правило(Правило :: если H1 то H).

% случай4: если в дереве Д нет правила с заключением Н,

%то поиск Н надо выполнять в дереве вывода предпосылок, т.е. в Дерево

как1(H,[Правило :: если Дерево то \_]):-как(H,Дерево).

% случай5: если дерево вывода - список поддеревьев вывода

% каждой конъюнктивной подцели правила из БЗ,

% то поиск Н следует выполнять в каждом из поддеревьев;

% поиск Н следует выполнять сначала в поддереве [Д1], а

% если Н не найдено, то продолжить поиск в оставшихся поддеревьях

как1(H,[]):-!.

как1(H,[Д1|Д2]):-как(H,[Д1]),!;

как1(H,Д2).

%вывод правила на экран

отобрази\_правило(Правило :: если H1 то H):-

write(Правило), write( ':'),nl,

write('если '), write(H1), nl,

write('то '), write(H),nl.

/\* Вызов интерпретатора\*/

инициализация:-retractall(сообщено(\_,\_)).

start:-

/\* Загрузка базы знаний из файла\*/

reconsult('./nb.pl'),

info,

%отображение информации о базе знаний\*

go\_exp\_sys.

go\_exp\_sys:-

инициализация,

Факт :: гипотеза(H),

найти([H],[H],Дерево),

write('решение:'),write(H),nl,

объясни(Дерево),

возврат.

%объяснение вывода утверждения

объясни(Дерево):-write( 'объяснить ? [цель/нет]:'), nl,read(H),

(H\=нет,!,как(H,Дерево),объясни(Дерево));!.

%поиск следующих решений

возврат:-write('Искать ещё решение [да/нет] ?: '),nl, read(нет).

Пример выполнения программы:

?- start.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Экспертная система \*

\* для определения будущей \*

\* профессии \*

\*-------------------------------------\*

\* Отвечайте на вопросы: \*

\* да, нет, почему \*

\* Для объяснения решения \*

\* введите цель \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Введите любой символ

|:

склад\_ума(гуманитарный)?

|: нет.

нравится(работать\_за\_компьютером)?

|: да.

склад\_ума(аналитический)?

|: почему.

пытаюсь доказать программист

с помощью правила: правило1

склад\_ума(аналитический)?

|: да.

решение:программист

объяснить ? [цель/нет]:

|: нет.

Искать ещё решение [да/нет] ?:

|: да.

нравится(работать\_с\_детьми)?

|: почему.

пытаюсь доказать педагог\_психолог

с помощью правила: правило10

нравится(работать\_с\_детьми)?

|: нет.

нравится(творческая\_работа)?

|: почему.

пытаюсь доказать цифровой\_художник

с помощью правила: правило2

нравится(творческая\_работа)?

|: да.

нравится(рисовать)?

|: нет.

нравится(музыка)?

|: почему.

пытаюсь доказать цифровой\_музыкант

с помощью правила: правило4

нравится(музыка)?

|: да.

решение:цифровой\_музыкант

объяснить ? [цель/нет]:

|: цифровой\_музыкант.

цифровой\_музыкант было доказано с помощью

правило4:

если [нравится(творческая\_работа),склад\_ума(аналитический),нравится(работать\_за\_компьютером),нравится(музыка)]

то цифровой\_музыкант

**Выводы**

В ходе выполнения лабораторной работы была разработана экспертная система продукционного типа на Прологе, исследованы базовые принципы организации экспертных систем.