МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3
по дисциплине «Логическое программирование»
Тема: Возможности внутренних баз данных.
Построение экспертных систем
Вариант 3.

Студент гр. 0303 _____ Бодунов П.А.
Студент гр. 0303 _____ Болкунов В.О.
Студент гр. 0303 _____ Калмак Д.А.
Преподаватель _____ Родионов С.В.

Санкт-Петербург
2024

Цель работы.

Целью работы является изучение возможностей по модификации внутренних баз данных языка Пролог, освоение принципов использования «статических переменных» в языке Пролог, а также изучение возможности создания экспертных систем на языке Пролог, освоение принципов формирования полноценных приложений, которые могут взаимодействовать с пользователем для сбора дополнительной информации.

Задачи.

Для достижения поставленной цели требуется решить следующие задачи:

1) Изучить теоретический материал.

2) Выполнить задание с номером варианта, равным номеру бригады.

3) Проверить выполнение программы.

4) Составить отчет о выполнении работы.

5) Представить на проверку файл отчета, файл текста программы на языке GNU Prolog, решающей поставленную задачу, и файл базы знаний экспертной системы.

Номер варианта и текст варианта задания должны быть представлены в форме комментариев в тексте программы. Номер группы и номер варианта должны присутствовать в имени файла с текстом программы.

Основные теоретические положения.

assert(X) – добавляет факт X в программу.

retract(X) – удаляет факт X из программы.

Добавление имеет две модификации assertz – добавить в конец программы, asserta – добавить в начало программы.

Пример использования:

?- assertz(data(1)).

yes

?- data(X).

X = 1

yes

?- listing.

data(1).

yes

?- retract(data(_)).

yes

?- listing.

yes

Для добавления правил в программу их необходимо поместить в дополнительные скобки: assertz((Правило)).

Пример:

?- assertz(man(socrat)), assertz((mortal(X):-man(X))).

yes

?- mortal(socrat).

yes

?- mortal(X).

X = socrat

yes

?- listing.

% file: user_input

mortal(A) :- man(A).

man(socrat).

yes

Предикат abolish(Имя_предиката/Арность_предиката) удаляет все вхождения предиката с данным именем и данной арностью.

Напишем аналогичный предикат удаления retractAll(X) через retract:

retractAll(X) :- retract(X), retractAll(X).

retractAll(_).

Он удаляет все вхождения X в нашу программу.

Статические переменные в Прологе можно определить с использованием assert и retrtact.

Способ работы:

init(ИмяПеременной, Значение) – инициализация статической переменной заданным значением.

set(ИмяПеременной, Значение) – установка значения в переменную.

get(ИмяПеременной, Значение) – получение значения из переменной.

Решение:

init(Var, Val) :- assertz(variables(Var,Val)).

set(Var, Val) :- retract(variables(Var, _)), assertz(variables(Var,Val)).

get(Var, Val) :- variables(Var, Val).

Пример использования:

?- init(t, 3), init(v, 4).

yes

?- set(t, data), get(t, T), get(v, V).

T = data

V = 4

yes

Естественно, повторная инициализация приведет к неправильной работе программы.

Задание.

Реализуйте экспертную систему, на языке Пролог, согласно одному из предложенных вариантов в соответствии со своим номером варианта.

Вопросы, задаваемые пользователю, не должны повторяться (дублироваться).

Базу знаний (начальные факты) требуется придумать самостоятельно (не менее 20 фактов).

Желательно, чтобы база знаний сохранялась в файле (и читалась при запуске программы).

В ответах/фактах из нескольких слов вместо пробелов следует писать символ "_".

Желательно, чтобы факты, вопросы и интерфейс были написаны на русском языке или на транслите.

Консультант по сотовой связи.

Имеется база знаний о моделях телефонов и их характеристиках/функциях. Также имеются факты о том, какие функции телефона нужны для конкретных целей (общение, выход в Интернет, мобильный офис...). Программа должна задать пользователю (покупателю) ряд вопросов о том, для чего ему нужен телефон, и предложить те модели телефонов, которые удовлетворяют его запросам.

Выполнение работы.

1. Порядок выполнения

Была создана база знаний для программы в файле *database.pl*. Были созданы следующие факты (переменные представлены лишь для описания фактов и правил, в базе знаний представлены реальные значения):

- модели телефонов с помощью правила *telephone(X)*, где X - модель телефона

- описание функций телефонов с помощью правила *opisanie(X, Y)*, где X - модель телефона, а Y - его функция
- виды деятельности пользователя с помощью правила *deatelnost(X)*, где X - вид деятельности пользователя.

Были созданы правила, которые соотносят специализацию телефона, или его цели, с функциями, которые для этого необходимы:
specializacia(P, X) :- opisanie(P, Y), где P - модель телефона, X - цель использования телефона, или деятельность, а Y - функция телефона, необходимая для этого.

А также созданы факты для вопросов, которые будут задаваться пользователю. Факт имеет вид *vopros(X, Y)*, где X - это цель телефона, а Y - текст для вопроса пользователю.

Программа запускается с правила start.

С помощью retractall проводится очистка базы знаний от telephone, opisanie, deyatelnost, specializacia, vopros, чтобы следующим шагом прочитать базу знаний из файла с помощью see.

Далее в соответствии с правилом *zagruzka* происходит считывание данных из файла и добавление фактов и правил с помощью assertz до тех пор с помощью repeat, пока не будет конец файла - end_of_file.

Далее считывание оканчивается и входной поток на чтение файла закрывается с помощью seen и очищаются ответы пользователя с помощью retractall.

Далее в соответствии с правилом *voprosi* с помощью *findall(X, deatelnost(X), D)* все виды деятельности будут помещены в список D. И используется правило *zadat_vopros*, в которое передается список D, чтобы задать вопросы пользователю для чего ему нужен телефон. Правило *zadat_vopros([D|Ds])* извлекает голову списка, а именно одну деятельность для вопроса. Факт *vopros(D, V)* находит соответствие между деятельностью и вопросом, который в понятном виде, должен быть

выведен пользователю на экран с помощью write. Далее ожидается ввод ответа в соответствии с правилом *vvod_otveta* в виде *da.*, *net.*, *konec.* с проверкой корректности ответа, используя repeat. Если введён *konec.*, то вопросы перестают задаваться. Иначе с помощью assertz добавляется факт *otvet(D, X)*, который сохраняет ответы пользователя для соответствующей деятельности D. Хвост Ds передается рекурсивно в *zadat_vopros(Ds)*.

Последнее правило в start - *recomendacia*, которое выводит рекомендации телефонов в соответствии с полученными желаниями пользователя. Изначально проводится проверка с помощью предиката clause, что у пользователя были положительные ответы на вопросы, тем самым создавая факты otvet с ответом da. Иначе выводится пользователю, что рекомендаций нет, как и если под желания пользователя не будет моделей. С помощью предиката *bagof(X, otvet(X, da), O)* каждая деятельность X, которая получила ответ da от пользователя, помещается в список O. Далее с помощью предиката *setof(P, (telephone(P), forall(member(X, O), specializacia(P, X))), M)* ищутся модели телефонов P, которые удовлетворяют всем деятельностям X, которые находятся в списке O. Соответствие между деятельностью и моделью телефона устанавливается через *specializacia*. Каждая модель, которая удовлетворяет всем деятельностям, отмеченным пользователем, помещаются в список M. Далее происходит вывод списка M рекомендованных моделей с помощью правила *vivod_spiska*. В соответствии с правилом *vivod_spiska* рекурсивно выводятся элементы списка - модели телефонов.

Тестирование программы представлено в разделе 3 Примеры вызова соответствующих правил и результаты выполнения.

2. Текст программы с комментариями

2.1. Текст программы (main.pl)

```
/*
Группа 0303. Вариант 3.
```

```
Консультант по сотовой связи.
Имеется база знаний о моделях телефонов и их характеристиках функциях. Также
имеются факты о том, какие функции телефона нужны для конкретных целей
(общение, выход в Интернет, мобильный офис...). Программа должна задать
пользователю (покупателю) ряд вопросов о том, для чего ему нужен телефон,
и предложить те модели телефонов, которые удовлетворяют его запросам.
*/

% Чтение базы знаний из файла
zagruzka :-
   repeat,
   read(T),
   assertz(T),
   T = end_of_file, !.

% Ввод ответа пользователя с проверкой на корректность
vvod_otveta(O) :-
   repeat,
   read(X),
   (X = da; X = net; X = konec),
   O = X, !.

% Вопрос пользователю (рекурсивно проходится по списку деятельностей и задаёт
соответствующие вопросы)
zadat_vopros([]).
zadat_vopros([D|Ds]) :-
   vopros(D, V),
   write(V), write(' (da. / net. / konec.): '),
   vvod_otveta(X),
   (
       (X = konec, !);
       (
           assertz(otvet(D, X)),
           zadat_vopros(Ds)
       )
   ).

% Поиск деятельностей и вызов вопросов
voprosi :-
   findall(X, deatelnost(X), D),
   zadat_vopros(D).

% Вывод списка через запятую
vivod_spiska([]).
vivod_spiska([X|Xs]) :- write(X), write(', '), vivod_spiska(Xs).

% Поиск и вывод рекомендованных моделей
recomendacia :-
   (
       clause(otvet(_, da), _),
       bagof(X, otvet(X, da), O),
       setof(
           P,
           (
               telephone(P),
               forall(member(X, O), specializacia(P, X))
           ),
           M
       ),
       write('Rekomendovannie modeli: '), vivod_spiska(M), nl
   );
```

```
    write('Rekomendacii otsutstvuyut').

% Запуск программы - очистка и чтение базы, опрос пользователя и вывод
результатов
start :-
    retractall(telephone(_, _)),
    retractall(opisanie(_, _)),
    retractall(deatelnost(_)),
    retractall(specializacia(_, _)),
    retractall(vopros(_, _)),
    see('database.pl'),
    zagruzka,
    seen,
    retractall(otvet(_, _)),
    voprosi,
    recomendacia.
```

2.2. Содержимое базы знаний (database.pl)

```
% Модели
telephone(nookia).
telephone(samsung).
telephone(honor).
telephone(huawai).
telephone(xiaomi).
telephone(iphone).

% Описание телефонов (функции)
opisanie(nookia, zvonok).
opisanie(samsung, zvonok).
opisanie(honor, zvonok).
opisanie(huawai, zvonok).
opisanie(xiaomi, zvonok).
opisanie(iphone, zvonok).

opisanie(samsung, internet).
opisanie(honor, internet).
opisanie(huawai, internet).
opisanie(xiaomi, internet).
opisanie(iphone, internet).

opisanie(honor, office).
opisanie(huawai, office).
opisanie(xiaomi, office).
opisanie(iphone, office).

opisanie(huawai, videokarta).
opisanie(xiaomi, videokarta).
opisanie(iphone, videokarta).

opisanie(xiaomi, kitaiskaya_telemetriya).
opisanie(iphone, amerikanskaya_telemetriya).

% Виды деятельности пользователя
deatelnost(razgovor).
deatelnost(internet).
deatelnost(rabota).
deatelnost(igri).
deatelnost(agent_sha).
deatelnost(agent_kitaya).
```

```
% Специализация телефона (соответствие деятельности и функции)
specializacia(P, razgovor) :- opisanie(P, zvonok).
specializacia(P, internet) :- opisanie(P, internet).
specializacia(P, rabota) :- opisanie(P, office).
specializacia(P, igri) :- opisanie(P, videokarta).
specializacia(P, agent_sha) :- opisanie(P, amerikanskaya_telemetriya).
specializacia(P, agent_kitaya) :- opisanie(P, kitaiskaya_telemetriya).

% Вопросы для специализации
vopros(razgovor, 'Vam nuzhen telephon dl`a razgovora?').
vopros(internet, 'Vam nuzhen dostup v internet?').
vopros(rabota, 'Vam nuzhen telephon dl`a raboty?').
vopros(igri, 'Vi igraete v mobilnie igri?').
vopros(agent_sha, 'Vi rabotaete na CRU?').
vopros(agent_kitaya, 'Vi rabotate na communisticheskuyu partiyu Kitaya?').
```

3. Примеры вызова соответствующих правил и результаты выполнения

Вызов правила *start* для поиска моделей телефонов, с помощью которых можно разговаривать и выходить в интернет, представлен на рис. 1:

```
| ?- start.

| ?- consult('C:/Work/Prolog/main.pl').
compiling C:/Work/Prolog/main.pl for byte code...
C:/Work/Prolog/main.pl compiled, 66 lines read - 5723 bytes written, 15 ms

yes
| ?- start.
Vam nuzhen telephon dl`a razgovora? (da. / net. / konec.): da.
Vam nuzhen dostup v internet? (da. / net. / konec.): da.
Vam nuzhen telephon dl`a raboty? (da. / net. / konec.): net.
Vi igraete v mobilnie igri? (da. / net. / konec.): net.
Vi rabotaete na CRU? (da. / net. / konec.): net.
Vi rabotate na communisticheskuyu partiyu Kitaya? (da. / net. / konec.): net.
Rekomendovannie modeli: honor, huawai, iphone, samsung, xiaomi,

true ?

yes
```

Рисунок 1 - Вызов правила start для поиска моделей телефонов для разговора и интернета

Вызов правила *start* для поиска моделей телефонов, с помощью которых можно разговаривать, представлен на рис. 2, причем пользователь не захотел получать следующие вопросы, а сразу перейти к рекомендациям:

```
| ?- start.

| ?- start.
Vam nuzhen telephon dl`a razgovora? (da. / net. / konec.): da.
Vam nuzhen dostup v internet? (da. / net. / konec.): konec.
Rekomendovannie modeli: honor, huawai, iphone, nookia, samsung, xiaomi,

true ?

yes
```

Рисунок 2 - Вызов правила start для поиска моделей телефонов для разговора без остальных вопросов

Вызов правила *start* для поиска моделей телефонов представлен на рис. 3, однако пользователь передумал и сразу захотел отменить поиск:

```
| ?- start.

| ?- start.
Vam nuzhen telephon dl`a razgovora? (da. / net. / konec.): konec.
Rekomendacii otsutstvuyut

yes
```

Рисунок 3 - Вызов правила start для поиска моделей телефонов с завершением на первом вопросе

Вызов правила *start* с положительным ответом на все вопросы представлен на рис. 4:

```
(1 ms) yes
| ?- start.
Vam nuzhen telephon dl`a razgovora? (da. / net. / konec.): da.
Vam nuzhen dostup v internet? (da. / net. / konec.): da.
Vam nuzhen telephon dl`a raboty? (da. / net. / konec.): da.
Vi igraete v mobilnie igri? (da. / net. / konec.): da.
Vi rabotaete na CRU? (da. / net. / konec.): da.
Vi rabotate na communisticheskuyu partiyu Kitaya? (da. / net. / konec.): da.
Rekomendacii otsutstvuyut

(1 ms) yes
| ?-
```

Рисунок 4 - Вызов правила start для поиска моделей телефонов с положительным ответом на все вопросы, где рекомендации отсутствуют

Выводы

В результате выполнения лабораторной работы была создана база знаний о телефонах, описании их функций, целей применения, связи с помощью специализации деятельности, или целей, и функций, а также вопросы для пользователя. Были описаны правила на языке GNU Prolog для поиска моделей телефонов, которые удовлетворяют запросам пользователя. Были приведены примеры вызова правила start для случаев, когда пользователю нужен телефон для двух целей, для одной цели с прерыванием следующих вопросов, с прекращением работы на первом вопросе, а так же когда запросам пользователя ни одна из моделей не удовлетворяет.

Роли членов бригады:

Бодунов Пётр 0303 написание кода, написание комментариев в код, оформление отчета.

Болкунов Владислав 0303 написание кода, написание комментариев в код, оформление отчета.

Калмак Даниил 0303 написание кода, написание комментариев в код, оформление отчета.

Трудности:

1. В случае отсутствия ответов пользователя (отсутствия правила otvet), правила использующие данный предикат завершались с ошибкой. Для решения данной проблемы потребовалось проверять существование предиката otvet с помощью предиката ***clause***.
2. Для поиска моделей удовлетворяющих всем требованиям потребовалось использовать предикат forall совместно с предикатом setof, однако без указания дополнительного ограничения на переменную P (как на модель телефона), forall срабатывал некорректно. Итоговый вариант вызова выглядит следующим образом:

```
setof(
    P,
    (
        telephone(P),
        forall(member(X, O), specializacia(P, X))
    ),
    M
)
```