Московский Авиационный Институт

(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа**

**по курсу «ЛП»**

**Тема:**

**Работа со списками и реляционным представлением данных.**

|  |  |
| --- | --- |
| Студент: | Суворова С. А. |
| Группа: | М80-206Б-18 |
| Преподаватель: | Сошников Д.В. |
| Преподаватель: | Левинская М. А. |
| Оценка: |  |
| Дата: |  |

Москва

2019

**Введение.**

Списки в языке Пролог отличаются от принятых в императивных языках тем, что не специализируются на типе данных и являются основной структурой для обработки последовательности данных.Данные списки похожи на линейные списки в традиционных языках.

**Предикат обработки списка.**

Предикат описанный ниже двумя способами удаляет из указанного списка первые 3 элемента формирует список без них.

Реализация:

remove1\_3\_first([X|Y],Y2):-

Y=[X1,Y1],Y1=[X2,Y2].

remove2\_3\_first([],[]).

remove2\_3\_first([X|Y],[X3|Y3]):-remove(X,[X,Y],[X1,Y1]),remove(X1,[X1,Y1],[X2,Y2]),remove(X2,[X2,Y2],[X3,Y3]).

Пример работы:

?- remove1\_3\_first([1,2,3,4],Y2).

Y2=[4].

Принимая в качестве первой переменной список, программа используя или не используя встроенных предикаты удаляет из указанного списка [X,Y] первые 3 элемента формирует список Y2 без них.

**Предикат обработки числового списка.**

Предикат описанный ниже ищет максимальный элемент в числовом списке.

Реализация:

for\_digit\_list([],\_).

for\_digit\_list([X,Y],N):-(N<X)->N=X,for\_digit\_list(Y,N).

Пример использования:

?- for\_digit\_list([1,3,2],N).

N=3.

Принимая в качестве первой переменной список [X,Y], программа проходится по нему последовательно, проверяя каждый элемент для того, чтобы найти максимальный. Переменная N идентифицируется с максимальным элементом.

**Реляционное представление данных.**

findsredingroup\_2(A,B)-для группы, ищет студента с максимальным средним балом.

findstudent\_2(L2)-для каждого студента ищет его максимальный средний балл и, сдал ли он экзамены.

findnopass\_2(A,N)- для предмета, ищет количество несдавщих студентов.

Реализация:

findmaxsred(L,N):-

[H|T]=L,

H=student(\_,\_,[grade(\_,N1),grade(\_,N2),grade(\_,N3),grade(\_,N4),grade(\_,N5),grade(\_,N6)]),

Help is N1+N2+N3+N4+N5+N6,

Help2 is Help/6,

(Help2>N)->N=Help2,

findmaxsred(T,N).

findsredingroup\_1(L,N,B):-

[H|T]=L,

H=student(\_,\_,[grade(\_,N1),grade(\_,N2),grade(\_,N3),grade(\_,N4),grade(\_,N5),grade(\_,N6)]),

Help is (N1+N2+N3+N4+N5+N6)/6,

(Help=:=N)->

B=Help

else

findsredingroup\_1(T,N,B).

findsredingroup\_2(A,B):-

findall(student(A,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),student(A,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),L),

findmaxsred(L,N),

findsredingroup\_1(L,N,B).

findstudent\_1(L,L2):-

[H|T]=L,

H=student(\_,A,[grade(\_,N1),grade(\_,N2),grade(\_,N3),grade(\_,N4),grade(\_,N5),grade(\_,N6)]),

Help is (N1+N2+N3+N4+N5+N6)/6,

(N1>2)->

((N2>2)->

((N3>2)->

((N4>2)->

((N5>2)->

((N6>2)->

(B=[A,"Yes",Help]);

B=[A,"No",Help]);

B=[A,"No",Help]);

B=[A,"No",Help]);

B=[A,"No",Help]);

B=[A,"No",Help]);

B=[A,"No",Help],

append(L2,B,L3),

findstudent\_1(T,L3).

findstudent\_2(L2):-

findall(student(A,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),student(A,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),L),

findstudent\_1(L,L2).

findnopass\_1(L,A,N):-

[H|T]=L,

H=student(\_,\_,[grade(A1,N1),grade(A2,N2),grade(A3,N3),grade(A4,N4),grade(A5,N5),grade(A6,N6)]),

(A1=:=A)->((N1=<2)->(N is N1+1));

((A2=:=A)->((N2=<2)->(N is N1+1)));

((A3=:=A)->((N3=<2)->(N is N1+1)));

((A4=:=A)->((N4=<2)->(N is N1+1)));

((A5=:=A)->((N5=<2)->(N is N1+1)));

((A6=:=A)->((N6=<2)->N is N1+1)),

findnopass\_1(T,N1).

findnopass\_2(A,N):-

findall(student(\_,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),student(\_,\_,[grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_),grade(\_,\_)]),L),

findnopass\_1(L,A,N).

Достоинства реляционной модели:

простота и доступность для понимания пользователем. Единственной используемой информационной конструкцией является "таблица";  
полная независимость данных. Изменения в прикладной программе при изменении реляционной базы данных минимальны;  
для организации запросов и написания прикладного программного обеспечения нет необходимости знать конкретную организацию базы данных во внешней памяти.

Недостатки реляционной модели:

далеко не всегда предметная область может быть представлена в виде "таблиц";  
в результате логического проектирования появляется множество "таблиц". Это приводит к трудности понимания структуры данных;  
база данных занимает относительно много внешней памяти;  
относительно низкая скорость доступа к данным.

findsredingroup\_2(A,B)-для группы A, ищет студента B с максимальным средним балом, .

findstudent\_2(L2)-для каждого студента ищет его максимальный средний балл и, сдал ли он экзамены, L2-список, состоящий из элементов типа[имя студента, сдал ли он все экзамены, его средний бал].

findnopass\_2(A,N)- для предмета А, ищет количество несдавщих его студентов N.

**Вывод.**

При выполнении данной лабораторной работы я научилась работать со списками, поняла, что в прологе обработка последовательности данных происходит с помощью рекурсии и в принципе осознала каким образом Пролог отличается от других языков программирования и каковы его особенности и как с ним работать .