МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНОЙ РАБОТЕ № 2 ПО КУРСУ «ПРЕДСТАВЛЕНИЕ ЗНАНИЙ В ИНФОРМАЦИОННЫХ СИСТЕМАХ»

Рекурсивные программы работы со списками на языке логического программирования ПРОЛОГ

СПИСКИ В ЯЗЫКЕ ЛОГИЧЕСКОГО ПРОГРАММИРОВАНИЯ ПРОЛОГ

Сведения о представлении списков в языке логического программирования ПРОЛОГ и принципах составления рекурсивных логических программ работы со списками изложены в выложенных лекциях № 8, 10 по курсу «Представление знаний в информационных системах». Дополнительная информация о принципах программирования задач работы со списками находится в учебнике по языку логического программирования Visual Prolog в подразделе DOC раздела VIP.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ № 2

Каждый студент получает индивидуальное задание на программирование задачи работы со списками, но исходные факты для всех студентов одинаковы.

Этими фактами являются следующие две группы фактов, имеющие соответственно предикатные символы woman и man. Первый терм имеет тип string. Его значением является имя женщины (woman) или мужчины (man). Второй терм имеет тип integer и его значением является возраст соответствующих мужчин и женщин.

```
woman("Galina", 23).
woman("Katerina", 31).
```

```
woman("Dasha", 21).
woman("Nelly", 27).
woman("Aleksandra", 18).

man("Vladimir", 23).
man("Sergey", 21).
man("Petr", 31).
man("Trofim", 27).
man("Aleksey", 18).
```

Согласно индивидуальному заданию необходимо составить и отладить логическую программу, которая позволяет извлекать из фактов списки и обрабатывать их некоторым образом.

Пример логической программы такого извлечения и обработки приведен в следующей программе.

```
DOMAINS
   name = string
   age = integer
   list = age*
   list2 = name*

PREDICATES
   nondeterm person(name, age)
   sumlist(list, age, integer)
   nondeterm execute
   nondeterm less(integer, integer, name)
   nondeterm findname(integer, list2)


CLAUSES
     sumlist([], 0, 0).
     sumlist([H|T], Sum, N):-
        sumlist(T, S1, N1),
        Sum=H+S1, N=1+N1.

     person("A", 23). /*Age is less than average*/
     person("B", 31).
     person("C", 21). /*Age is less than average*/
     person("D", 27).
     person("E", 18). /*Age is less than average*/

     less(Average,Age,_):-
        Average<=Age.
```

```
less(Average,Age,X):-
   Average>Age,
   write(X),nl.

findname(_, []).
findname(Average, [X|Rest]):-
   person(X, Y),
   less(Average, Y, X),
   findname(Average, Rest).

execute:-
   findall(Age, person(_, Age), L),
   sumlist(L, Sum, N),
   Average = Sum/N,
   write("Average = ", Average), nl,
   write("Age is less than average:"),nl,
   findname(Average,["A", "B", "C", "D", "E"]).

GOAL
  execute.
```

Данная программа содержит четыре раздела domains, predicates, clauses, goal. В раздел CLOUSES включены пять фактов, описывающие персон (person) с именами A, B, C, D, E:

```
person("A", 23).
person("B", 31).
person("C", 21).
person("D", 27).
person("E", 18).
```

Целью execute программы является нахождение и распечатка имен всех персон, возраст которых меньше или равен среднему.

При достижении этой цели:

1. библиотечный предикат findall(Age, person(_, Age), L) формирует список возрастов всех персон;
2. предикат sumlist(L, Sum, N) находит сумму возрастов и число персон;
3. предикат findname(Average,["A", "B", "C", "D", "E"]) находит и распечатывает имена только тех персон, средний возраст которых меньше или равен среднему;

4. результатом работы программы является следующий:

```
Average = 24
Age is less than average:
A
C
E
yes
```

ВЫПОЛНЕНИЕ, ОФОРМЛЕНИЕ И ЗАЩИТА ЛАБОРАТОРНОЙ РАБОТЫ

Выполнение работы состоит в составлении индивидуальной для каждого студента программы работы со списками, отладке ее на интерпретаторе ПРОЛОГа, получении решения и составлении отчета.

В тексте программы должны приводиться комментарии на русском языке, поясняющие содержание программы.

Каждый студент выкладывает свой отчет на https://moodle.iu3.bmstu.ru

Содержание отчета:

1) Титульный лист
2) Исходный текст задания согласно индивидуальному варианту
3) Текст составленной программы
4) Решения, полученные в результате выполнения программы

Защита работы состоит в проверке преподавателем предоставленного отчета и его оценивании.

В случае ошибок или несоответствия поставленной задаче отчет возвращается на доработку.