Функциональное и логическое программирование

Лекция 5

2.11 Списки

Список – упорядоченный набор объектов (термов). Список может содержать объекты разных типов, в том числе и списки. Элементы списка разделяются запятыми и заключаются в квадратные скобки.

Пример 1:

[1, 2, 3]

[a, [b, c]]

2.11.1 Голова и хвост списка

Голова списка – первый элемент.
Хвост списка – часть списка без первого элемента.

Пример 2 (деление списка на хвост и голову):

Список	Голова	Хвост
[1,2,3,4]	1	[2,3,4]
[a]	a	[ ]
[ ]	не опр	не опр

Деление на голову и хвост осуществляется с помощью специальной формы представления списка: [Head|Tail].

Пример 3 (Сопоставление списков):

Список 1	**Список 2**	**Результаты сопоставления**
[X,Y,Z]	[1,2,3]	X=1, Y=2, Z=3
[5]	[X|Y]	X=5, Y=[]
[1,2,3,4]	[X,Y|Z]	X=1, Y=2, Z=[3,4]
[1,2,3]	[X,Y]	не опр
[a,X|Y]	[Z,a]	Z=a, X=a, Y=[]
[a,b,c,d,e,f]	[_,_,X|_]	X=c

2.11.2 Операции со списками
2.11.2.1 Принадлежность элемента списку

Пример 4:

```
member1(X,[X|_]).
member1(X,[_|T]):-member1(X,T).
```

При использовании можно задавать значения одного, или двух аргументов, или не задавать их вообще.

```
?- member1(1,[2,3,1,5,1,7]).
true .

?- member1(X,[2,3,1,5,1,7]).
X = 2 ;
X = 3 ;
X = 1 ;
X = 5 ;
X = 1 ;
X = 7 ;
false.
```

Есть встроенный предикат member.

```
?- member1(1,L).
L = [1|_] ;
L = [_, 1|_] ;
L = [_, _, 1|_] ;
L = [_, _, _, 1|_] .

?- member1(X,L).
L = [X|_] ;
L = [_, X|_] ;
L = [_, _, X|_] ;
L = [_, _, _, X|_]
Unknown action: / (h for help)
Action? .
```

2.11.2.2 Соединение двух списков (аналог append)

Пример 5:

```
append1([],L2,L2).
append1([H|T],L2,[H|T1]):-append1(T,L2,T1).
```

```
?- append1([1,2,3],[a,s,d,g],L).
L = [1, 2, 3, a, s, d, g].
```

Можно использовать предикат append1 для следующих целей:

- слияние двух списков;
- получение всех возможных разбиений списка;
- поиск подсписков до и после определенного элемента;
- поиск элементов списка, стоящих перед и после определенного элемента;
- удаление части списка, начиная с некоторого элемента;
- удаление части списка, предшествующей некоторому элементу.

Есть встроенный предикат append.

Вопрос в Пролог-системе	Ответ Пролог-системы
append1([1,2],[3],L).	L = [1, 2, 3]
append1(L1,L2,[1,2,3]).	L1 = [], L2 = [1, 2, 3]; L1 = [1], L2 = [2, 3]; L1 = [1, 2], L2 = [3]; L1 = [1, 2, 3], L2 = []; false
append1(Before,[3|After],[1,2,3,4,5]).	Before = [1, 2] After = [4, 5]
append1(_,[Before,3,After|_],[1,2,3,4,5]).	Before = 2 After = 4
append1(L1,[3|_],[1,2,3,4,5]).	L1 = [1, 2]
append1(_,[3|L2],[1,2,3,4,5]).	L2 = [4, 5]

2.11.2.3 Добавление и удаление элемента из списка

Пример 6 (добавление в начало списка, удаление первого вхождения заданного элемента):

```
insert(X,L,[X|L]).
select1(_,[],[]).
select1(X,[X|T],T).
select1(X,[Y|T],[Y|T1]):-select1(X,T,T1).
```

```
?- select1(1,[2,3,1,1,5,1,6,1,7],L).
L = [2, 3, 1, 5, 1, 6, 1, 7] ;
L = [2, 3, 1, 5, 1, 6, 1, 7] ;
L = [2, 3, 1, 1, 5, 6, 1, 7] ;
L = [2, 3, 1, 1, 5, 1, 6, 7] ;
L = [2, 3, 1, 1, 5, 1, 6, 1, 7] ;
false.

?- select(1,[2,3,1,1,5,1,6,1,7],L).
L = [2, 3, 1, 5, 1, 6, 1, 7] .

?- select(1,[2,3,1,1,5,1,6,1,7],L).
L = [2, 3, 1, 5, 1, 6, 1, 7] ;
L = [2, 3, 1, 5, 1, 6, 1, 7] ;
L = [2, 3, 1, 1, 5, 6, 1, 7] ;
L = [2, 3, 1, 1, 5, 1, 6, 7] ;
false.
```

Предикат select1 можно использовать также для добавления элемента в список. Есть встроенный предикат select.

Пример 7 (удаление всех вхождений заданного элемента):

```
delete1([],_,[]):-!.
delete1([X|T],X,T1):-delete1(T,X,T1),!.
delete1([Y|T],X,[Y|T1]):-delete1(T,X,T1).
```

```
?- delete1([1,2,3,1,1,4,1,5],1,L).
L = [2, 3, 4, 5].
```

Есть встроенный предикат delete.

2.11.2.4 Деление списка на два списка по разделителю

Пример 8:

Деление списка на две части, используя разделитель M (если элемент исходного списка меньше разделителя, то он помещается в первый результирующий список, иначе – во второй результирующий список).

```
split(M,[H|T],[H|T1],L2):-H@<M,!,split(M,T,T1,L2).
split(M,[H|T],L1,[H|T1]):-split(M,T,L1,T1).
split(_,[],[],[]).
```

```
?- split(7,[g,1,f,5,8,1,0,q],L1,L2).
L1 = [1, 5, 1, 0],
L2 = [g, f, 8, q] ;
false.
```

2.11.2.5 Подсчет количества элементов в списке

Пример 9:

```
count([],0).
count([_|T],N):-count(T,N1),N is N1+1.
```

```
?- count([a,s,d,f,g],N).
N = 5.
```

Есть встроенный предикат length(L,N) – подсчет количества элементов N в списке L.

reverse(L1,L2) – обращение любого из списков-аргументов.

2.11.3 Сортировка списков (по неубыванию)
2.11.3.1 Сортировка вставкой

Пример 10:

Добавляем голову списка в нужное место отсортированного хвоста.

```
insort([],[]):-!.
insort([H|T],S_list):-insort(T,S_list1),add(H,S_list1,S_list).
add(X,[],[X]):-!.
add(X,[Y|T],[Y|T1]):-X@>Y,add(X,T,T1),!.
add(X,L,[X|L]).
```

```
?- insort([9,3,7,1,5,0,10],L).
L = [0, 1, 3, 5, 7, 9, 10].
```

2.11.3.2 Пузырьковая сортировка

Пример 11:

Меняем местами соседние элементы до тех пор, пока есть неверно упорядоченные пары.

```
pusort(L,S_list):-swap(L,L1),!,pusort(L1,S_list).
pusort(L,L).
swap([X,Y|T],[Y,X|T]):-X@>Y.
swap([X|T],[X|T1]):-swap(T,T1).
```

```
?- pusort([9,3,7,1,5,0,10],L).
L = [0, 1, 3, 5, 7, 9, 10].
```

2.11.3.3 Быстрая сортировка

Пример 12:

Разбиваем список на два списка по разделителю – голове списка, сортируем эти списки и соединяем их.

```
qsort([],[]):-!.
qsort([H|T],S_list):-split(H,T,Less,More),
                qsort(Less,S_less),qsort(More,S_more),
                append(S_less,[H|S_more],S_list).
```

```
?- qsort([9,3,7,1,5,0,10],L).
L = [0, 1, 3, 5, 7, 9, 10] .
```

Временная сложность алгоритма быстрой сортировки примерно $n \cdot \log n$.

Есть встроенные предикаты сортировки по неубыванию: sort (с удалением дубликатов), msort.

2.11.4 Компоновка данных в список

bagof(X,P,L)

- порождает список L всех объектов X, удовлетворяющих цели P.

(X и P содержат общие переменные). Если таких объектов нет, то bagof неуспешен.

Если один и тот же X найден многократно, то все его экземпляры будут занесены в L, что приведет к появлению в L повторяющихся элементов.

Пример 13:

Опишем предикат буква для разбиения букв из некоторого множества на гласные и согласные.

```
bukva(a,glas).
bukva(b,soglas).
bukva(i,glas).
bukva(o,glas).
bukva(g,soglas).
bukva(f,soglas).
```

Список согласных:

```
?- bagof(X,bukva(X,soglas),L).
L = [b, g, f].
```

Списки гласных и согласных:

```
?- bagof(X,bukva(X,Y),L).
Y = glas,
L = [a, i, o] ;
Y = soglas,
L = [b, g, f].
```

setof(X,P,L)

- аналогичен bagof.

Отличие от bagof: список L упорядочен по отношению ‘@<', и не содержит повторяющихся элементов.

Пример 14:

Список пар вида <буква>/<вид>.

```
?- setof(X/Y,bukva(X,Y),L).
L = [a/glas, b/soglas, f/soglas, g/soglas, i/glas, o/glas].
```

findall(X,P,L)

- аналогичен предикату bagof.

Отличия от bagof:

- Собирает в список все объекты X, не обращая внимание на возможно отличающиеся для них конкретизации тех переменных из P, которых нет в X.
- Если объекты не найдены, то предикат успешен, а список L – пустой.

Пример 15:

Имеется предикат ребенок, связывающий имя ребенка и его возраст. Сформировать список детей старше 5 лет.

```
rebenok(a,6).
rebenok(s,10).
rebenok(g,15).
rebenok(f,4).
rebenok(y,3).
rebenok(aa,12).
```

```
?- findall(X,(rebenok(X,Age),Age>5),L).
L = [a, s, g, aa].
```