Язык SWI Prolog

Типовые процедуры обработки списков в программах на языке Пролог

Типовые процедуры обработки списков

Типовые процедуры обработки списков не являются стандартными предикатами системы программирования языка Пролог.

Предикат add

Предикат add(X,Y,Z) истинен, если список Z получается добавлением терма X в начало списка Y. Схема отношения этого предиката имеет вид:

add(<терм>,<список>,<список>).

Декларативное описание предиката add

Декларативное описание предиката add формулируется следующим образом:

Терм X является головой списка Z, а список Y —хвостом списка Z.

Процедура add(X,Y,Z) состоит из факта:

add(X,Y,[X|Y]).

Предикаты revers1 и revers2.

Предикаты revers1 и revers2 являются предикатами обращения списков и определяют одно и то же отношение различными способами. Схема отношения этого предиката имеет вид:

revers(<список>,<список>).

Предикаты revers1 и revers2.

Процедура revers определяется двумя способами:

- простым обращением;
- обращением с накоплением.

Предикаты revers

Предикат revers(X,Y) истинен, если список Y содержит все элементы списка X, которые записаны в списке Y в обратном порядке. Перестановку элементов списка в обратном порядке можно произвести путем многократного выполнения процедуры append.

Простое обращение. Декларативное определение предика revers1

1) обращенный пустой список есть пустой список;
2) если список X можно разделить на голову H и хвост Xs, то Zs есть обращенный список, если Ys —обращенный хвост списка X, Zs получен путем присоединения к Ys головы H списка X.

Процедура простого обращения

Простое обращение выполняется процедурой revers1, которая использует процедуру append и состоит из двух предложений:

```
revers1([ ],[ ]).
revers1([H|Xs],Zs): —revers1(Xs,Ys),
                     append(Ys,[H],Zs).
```

Обращение с накоплением

В процедуре revers2 введен дополнительный **предикат rev** с тремя аргументами —списками, где первый аргумент —исходный список, второй аргумент —накапливающийся список, а третий аргумент —результирующий, обращенный список.

Декларативное определение предиката rev

Декларативное определение предиката rev формулируется следующим образом:

1) если первый аргумент есть пустой список, то второй и третий аргументы представляют собой один и тот же список;

Декларативное определение предиката rev

2) первый аргумент — непустой список [H|Xs], и его можно разделить на голову H и хвост Xs; в этом случае применение предиката rev к списку [H|Xs] и накапливающемуся списку L равносильно применению предиката rev к хвосту списка Xs и списку [H|L]; при этом получается обращенный список Y.

Процедура revers2

Обращение списка с накоплением выполняется процедурой revers2, состоящей из трех предложений и содержит дополнительный предикат rev:

```
revers2(X,Y): –rev(X,[ ],Y).
rev([ ],Y,Y).
rev([H|Xs],L,Y): –rev(Xs,[H|L],Y).
```

Предикат delete

Предикат delete(X,L,M) принимает значение “истина”, если список M получается в результате удаления первого вхождения терма X из списка L.

Схема отношения этого предиката имеет вид:

delete(<терм>,<список>,<список>).

Декларативное описание предикат delete

Декларативное описание предиката next формулируется следующим образом:

1) Если X — голова списка L, то предикат delete(X,L, M) истинен и M есть хвост списка L.

2) Если X принадлежит хвосту списка, то предикат delete необходимо применить к хвосту списка L.

Декларативное описание предикат delete

3) *Если X не принадлежит списку L, то предикат delete(X,L, M) ложен.*

Процедура delete

Процедура delete(X,Y,L) состоит из двух правил:

delete(X,[X|B],B): —!.

delete(X,[Y|L],[Y|M]): —delete(X,L,M).

Предикат number_list

Предикат number_list(L) определяет, является ли список X списком числовых термов. Схема отношения этого предиката имеет вид:
number_list(<список>).

Декларативное описание предиката number_list(L)

1) Список L включает один элемент X. Тогда предикат number_list([X]) истинен, если X числовой терм.
2) Список L можно разделить на голову H и хвост Xs. Тогда L есть список числовых термов, если H —числовой терм и хвост списка есть список числовых термов.

Процедура number_list

Процедура number_list(X,Y) состоит из двух правил:

```
number_list([]): —number(X).
number_list(X,[_|T]): —number(X),
                       number_list(X,T).
```

Предикат number(X) —стандартный предикат системы Arity Prolog, этот предикат истинен, если X —числовой терм.

Предикат sumlist

Предикат sumlist(L,Sum) определяет сумму элементов числового списка.

Схема отношения этого предиката имеет вид:

sumlist(<список>,<целочисленный терм>).

Декларативное описание предиката sumlist(L)

- Сумма элементов пустого списка равна нулю.
- Если исходный список состоит L из головы H и хвоста T, то сумма элементов списка L равна сумме элементов хвоста списка T плюс H.

Процедура sumlist

Процедура sumlist(L,Sum) состоит из двух правил:

```
sumlist([ ],0).
sumlist([H|T],Sum): —sumlist(T,SumT),
                    Sum is SumT+H.
```

Предикат delrepeat

Предикат delrepeat (L,LS) истинен, если список получается из списка S путем удаления всех повторений элементов. Схема отношения этого предиката имеет вид:

delrepeat(<список>,<список>).

Предикат delrepeat

Удаление повторений элементов выполняется процедурой delrepeat с накоплением списка, состоящей из четырех предложений и содержит дополнительный предикат delrep.

Декларативное описание предиката delrepeat

1) если первый аргумент есть пустой список, то второй и третий аргументы представляют собой один и тот же список;
2) если первый аргумент — непустой список [H|Xs], и голова H принадлежит хвосту списка T, то процедура delrep рекурсивно вызывается с аргументами T и S1; при этом элемент H не включается в накапливающийся список S1;

Декларативное описание предиката delrepeat

3) *если первый аргумент —непустой список [H|Xs], и голова H не принадлежит хвосту списка T, то элемент H включается в накапливающийся список S1 и получается список S2. Затем процедура delrep рекурсивно вызывается с аргументами T и S2.*

Процедура delrepeat

```
delrepeat(S,SF):-delrep(S,[ ],SF).
delrep([ ],S,S).
delrep([H|T],S1,SF):-member(H,T),!,delrep(T
    ,S1,SF).
delrep([H|T],S1,SF):-append(S1,[H],S2),delr
    ep(T,S2,SF).
```

Полный текст процедуры delrepeat

```
delrepeat(S,SF):-delrep(S,[ ],SF).
delrep([ ],S,S).
delrep([H|T],S1,SF):-member(H,T),!,
                     delrep(T,S1,SF).
delrep([H|T],S1,SF):-append(S1,[H],S2),
                     delrep(T,S2,SF).
member(X,[X|_]).
member(X,[Y|T]):-member(X,T).
append([],X,X).
append([H|T1],X,[H|T2]):-append(T1,X,T2).
```

Выполнение процедуры delrepeat

```
% d:/ИИС/Для МИСИС/ПРАКТИКА/delrepeat.txt
   compiled 0.02 sec, 2,208 bytes
1 ?- delrepeat([1,1,2,2,4,7,4,8],SF).

SF = [1, 2, 7, 4, 8]

Yes
2 ?-
```