**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**Лабораторная работа №1**

**по курсу «Логическое программирование»**

Студент: Архипов М. Д.

Группа: М8О-407Б-18

Преподаватель: Левинская М. А.

Сошников Д.В.

Оценка:

Москва 2021

**Тема:** Обработка списков.

**Цель работы:**первоначальное ознакомление с выбранной системой программирования на языке Пролог, реализация предикатов обработки списков в различных представлениях.

**Задание:** Вариант № 1.

Получение последнего эл-та списка.

Вычисление суммы эл-тов числового списка.

**Использованная версия языка Prolog:** SWI-Prolog 6.6.4.

**Текст программы:**

leng([],0). % вычисление длины списка

leng([X|T],N):-

leng(T,N1),

N is N1+1.

mem(X,[X|T]). % членство в списке

mem(X,[\_|T]):-

mem(X,T).

app([],X,X). % слияние списков

app([X|Y],Z,[X|T]):-

app(Y,Z,T).

rem(X,[X|T],T). % удаление X из списка

rem(X,[Y|T],[Y|T1]):-

rem(X,T,T1).

permute([],[]). % перестановки списка

permute(L,[X|T]):-

rem(X,L,R),

permute(R,T).

subl(X,L):- % подсписок

app(Y1,Y2,L),

app(X,X1,Y2).

deln([\_|Y],1,Y):-!. % удаление элемента, стоящего

deln([X|Y],N,[X|L]):- % на n-ой позиции

N1 is N-1,!,

deln(Y,N1,L).

nummem(A,[e(A,\_)|\_]). % принадлежность к нумерованному списку

nummem(A,[\_|Y]):-

nummem(A,Y).

numapp(X,[],X). % слияние нумерованных списков

numapp(X,[e(C,D)|Y],M):-

leng(X,N),

N1 is N + 1,

app(X,[e(C,N1)],L),

numapp(L,Y,M).

dec([e(A,B)],[e(A,C)]):- % декремент номеров

C is B-1.

dec([e(A,B)|C],[e(A,B1)|C1]):-

B1 is B-1,

dec(C,C1).

numrem(A,[e(A,N)],[]). % удаление вхождения эл-та из списка

numrem(A,[e(A,B)|C],C1):-

dec(C,C1).

numrem(X,[A|C],[A|C1]):-

numrem(X,C,C1).

numpermute([],[]). % перестановка нумерованного списка

numpermute(L,X):-

permute(L,M),

numapp([],M,X).

lastitem([X],X). % получение последнего эл-та (без использования стандартных предикатов).

lastitem([Y|L],X):-

lastitem(L,X).

laststand(L,X):- % получение последнего эл-та (с использованием стандартных предикатов).

app(T,[X],L).

numlast([e(A,N)],A). % получение последнего эл-та нумерованного списка( без исользования станд. предикатов)

numlast([e(A,N)|T],X):-

numlast(T,X).

numlaststand(L,[]):- % получение последнего эл-та нумерованного списка( c исользованием станд. предикатов)

length(L,0).

numlaststand(L,X):-

length(L,N),

deln(L,N,T),

numapp(T,[e(X,N)],L).

suml([X],X). % сумма эл-тов числового списка

suml([X|T],N):-

suml(T,N1),

N is N1+X.

numsuml([e(X,N)],X).% сумма эл-тов числового нумерованного списка

numsuml([e(X,\_)|T],N):-

numsuml(T,N1),

N is N1+X.

**Примеры работы стандартных предикатов:**

?- length([a,n,j,m],N).

N = 4.

?- length([a,n,[j,m]],N).

N = 3.

?- member([a,n,m,h],n).

false.

?- member(n,[a,n,m,h]).

true ;

false.

?- member(n,[a,n,m,n,h]).

true ;

true ;

false.

?- delete([a,b,y,u,y],y,X).

X = [a, b, u].

?- delete([a,b,y,[u,y]],y,X).

X = [a, b, [u, y]].

?- delete([a,b,y,[u,y]],[u,y],X).

X = [a, b, y].

**Примеры работы реализованных предикатов:**

?- leng([n,m,k,l],N).

N = 4.

?- leng([n,m,[k,l]],N).

N = 3.

?- mem(n,[n,m,k,l]).

true .

?- mem(X,[n,m,k,l]).

X = n ;

X = m ;

X = k ;

X = l ;

false.

?- app([],[],X).

X = [].

?- app([t,u],[b,n,m],X).

X = [t, u, b, n, m].

?- app(X,Y,[t,u,b,n,m]).

X = [],

Y = [t, u, b, n, m] ;

X = [t],

Y = [u, b, n, m] ;

X = [t, u],

Y = [b, n, m] ;

X = [t, u, b],

Y = [n, m] ;

X = [t, u, b, n],

Y = [m] ;

X = [t, u, b, n, m],

Y = [] ;

false.

?- rem(b,[t,u,b,n,m],X).

X = [t, u, n, m] ;

false.

?- rem(b,[b,t,u,b,n,m],X).

X = [t, u, b, n, m] ;

X = [b, t, u, n, m] ;

false.

?- permute([v,b,n],X).

X = [v, b, n] ;

X = [v, n, b] ;

X = [b, v, n] ;

X = [b, n, v] ;

X = [n, v, b] ;

X = [n, b, v] ;

false.

?- permute([v,b,n],[b,n,v]).

true ;

false.

?- permute([v,b,n],[b,n,u]).

false.

?- subl([h,k,i],[y,k,o,h,y,h,k,i,g,t,r]).

true ;

false.

?- subl(X,[y,k,o]).

X = [] ;

X = [y] ;

X = [y, k] ;

X = [y, k, o] ;

X = [] ;

X = [k] ;

X = [k, o] ;

X = [] ;

X = [o] ;

X = [] ;

false.

?- nummem(t,[e(n,1),e(o,2),e(r,3),e(g,4),e(y,5)]).

false.

?- nummem(n,[e(n,1),e(o,2),e(r,3),e(g,4),e(y,5)]).

true .

?- nummem(n,[e(n,1),e(o,2),e(r,3),e(n,4),e(y,5)]).

true ;

true ;

false.

?- nummem(X,[e(n,1),e(o,2)]).

X = n ;

X = o ;

false.

?- numapp([e(n,1),e(o,2)],[e(n,1),e(o,2),e(r,3),e(g,4),e(y,5)],X).

X = [e(n, 1), e(o, 2), e(n, 3), e(o, 4), e(r, 5), e(g, 6), e(y, 7)] .

?- numrem(r,[e(n, 1), e(o, 2), e(n, 3), e(o, 4), e(r, 5), e(g, 6), e(y, 7)],X).

X = [e(n, 1), e(o, 2), e(n, 3), e(o, 4), e(g, 5), e(y, 6)] ;

false.

?- numrem(r,[e(n, 1), e(o, 2), e(r, 3), e(o, 4), e(r, 5), e(g, 6), e(y, 7)],X).

X = [e(n, 1), e(o, 2), e(o, 3), e(r, 4), e(g, 5), e(y, 6)] ;

X = [e(n, 1), e(o, 2), e(r, 3), e(o, 4), e(g, 5), e(y, 6)] ;

false.

?- numpermute([e(n, 1), e(o, 2), e(r, 3)],X).

X = [e(n, 1), e(o, 2), e(r, 3)] ;

X = [e(n, 1), e(r, 2), e(o, 3)] ;

X = [e(o, 1), e(n, 2), e(r, 3)] ;

X = [e(o, 1), e(r, 2), e(n, 3)] ;

X = [e(r, 1), e(n, 2), e(o, 3)] ;

X = [e(r, 1), e(o, 2), e(n, 3)] ;

false.

?- lastitem([a,nm,k,y,u,i],X).

X = i ;

false.

?- lastitem([a,nm,k,y,u,i],i).

true ;

false.

?- lastitem([a,nm,k,y,u,i],y).

false.

?- laststand([a,n,m,k,y,u,i],X).

X = i ;

false.

?- laststand([a,n,m,k,y,u,i],i).

true .

?- numlast([e(n, 1), e(o, 2), e(o, 3), e(r, 4), e(g, 5), e(y, 6)],X).

X = y .

?- numlast([e(n, 1)],X).

X = n .

?- numlaststand([e(n, 1), e(o, 2), e(o, 3), e(r, 4), e(g, 5), e(y, 6)],X).

X = y .

?- numlaststand([e(n, 1)],X).

X = n .

?- suml([1,6,7,8,9,0],X).

X = 31 .

?- suml([1,6,7,8,9,0],45).

false.

?- numsuml([e(1,1),e(6,2),e(7,3),e(8,4),e(9,5),e(0,6)],45).

false.

?- numsuml([e(1,1),e(6,2),e(7,3),e(8,4),e(9,5),e(0,6)],X).

X = 31 .

**Пример совместного использования предикатов получение последнего**

**эл-та и слияния списков:**

Данные предикаты можно использовать для составления предиката реверса списка:

reversl([X],[X]).

reversl([X,Y],[Y,X]).

reversl(L,[X|R]):-

lastitem(L,X),

app(T,[X],L),

reversl(T,R).

Пример работы:

?- reversl([a],X).

X = [a] .

?- reversl([a,b],X).

X = [b, a] .

?- reversl([a,b,c,n,j],X).

X = [j, n, c, b, a] .

**Выводы:** работа со списками в Prolog`е в некотором смысле легче чем в императивных языках, т.к. не нужно расписывать алгоритмы обработки ,достаточно указать соответствующий предикат. Но Prolog программа  может оказаться медленнее чем ее аналог на императивном языке, из –за того ,что Prolog «ищет» результат за пользователя.