Министерство науки и высшего образования РФ

Севастопольский государственный университет

Кафедра информатики и управления в технических системах

ОТЧЕТ

по лабораторной работе №5

ОБРАБОТКА СПИСКОВ

по дисциплине «Логическое программирование»

Выполнил:

Студент группы ИВТ/б 22-о

Черняев Н.Г.

Проверил:

Ткаченко К.С.

г. Севастополь 2019

**Цель работы**

Ознакомиться с реализацией структуры данных типа список в языке Пролог и методами их обработки.

**Вариант № 23(4, 15, 6)**

Задание:

Реализовать набор предикатов для обработки списков:

· Добавление элемента X к списку List.

· Удаление элемента X из списка List.

· Конкатенация списков.

· Определение длины списка.

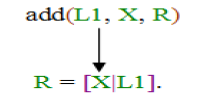
· Определение принадлежности элемента Х списку List.

Выполнить задание согласно вариантам.

* Вариант 15 - Вычесть 1-цу от каждого элемента списка.
* Вариант 4 - Определить отношение перевод(Список1, Список2) для перевода списка чисел от 0 до 9 в список соответствующих слов нуль, один и т.д.
* Вариант 6 - Определите два предиката четнаядлина**(**Список**)** и нечетная длина**(**Список**)** таким образом, чтобы они были истинными, если их аргументом является список четной или нечетной длины соответственно. Например, список [a,b,c,d ] имеет четную длину.

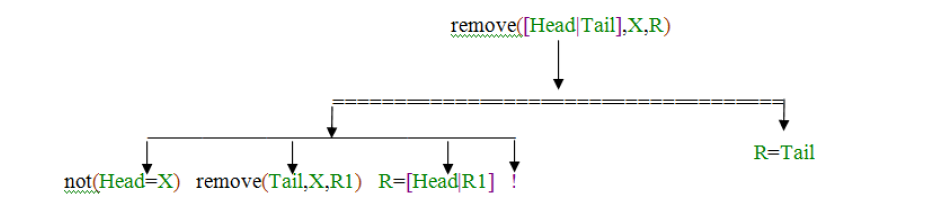
**Ход работы**

Реализуем предикат - Добавление элемента X к списку List.



add(L1, X, R):- R = [X|L1].

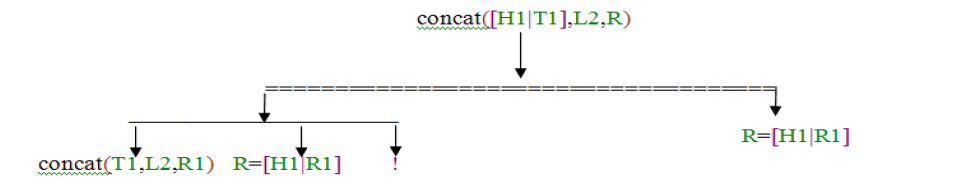
Реализуем предикат - Удаление элемента X из списка List.



remove([],\_,[]):-!

remove([Head|Tail],X,R):-not(Head=X),remove(Tail,X,R1),R=[Head|R1],!. remove([\_|Tail],\_,R):-R=Tail.

Реализуем предикат - Конкатенация списков.

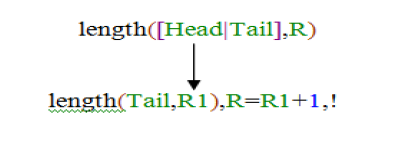


concat(\_,\_,[]):-!

concat([H1|[]],L2,R):-R = [H1|L2],!.

concat([H1|T1],L2,R):- concat(T1,L2,R1), R=[H1|R1],!.

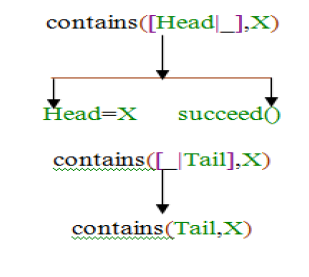
Реализуем предикат - Определение длины списка.



length([],0):-!

length([Head|Tail],R):-length(Tail,R1),R=R1+1,!.

Реализуем предикат - Определение принадлежности элемента Х списку List.



contains([],\_):-fail().

contains([Head|\_],X):-Head=X, succeed(),!.

contains([\_|Tail],X):-contains(Tail,X).

Текст программы:

implement main   
    open core   
constants   
    className = "main".   
    classVersion = "".   
domains   
    list = integer\*.   
class predicates   
    add : (list, integer, list) procedure(i,i,o).   
    remove : (list, integer, list) procedure(i,i,o).   
    concat : (list,list,list) procedure(i,i,o).   
    length :(list,integer) procedure(i,o).   
    contains :(list, integer) determ.  
    minus :(list,list) procedure(i,o).  
clauses   
    classInfo(className, classVersion).   
    add(L1, X, R):- R = [X|L1].   
    remove([],\_,[]):-!.  
    remove([Head|Tail],X,R):-not(Head=X),remove(Tail,X,R1),R=[Head|R1],!.   
    remove([\_|Tail],\_,R):-R=Tail.   
    concat([],L2,L2):-!.  
    concat([H|L1],L2,[H|L3]):- concat(L1,L2,L3),!.  
    length([],0):-!.  
    length([Head|Tail],R):-length(Tail,R1),R=R1+1,!.   
    contains([],\_):-fail().  
    contains([Head|\_],X):-Head=X, succeed(),!.   
    contains([\_|Tail],X):-contains(Tail,X).  
      
    minus([], []).  
    minus([H1|T1], [H2|T2]) :-  
        H2 = H1 - 1,  
        minus(T1, T2).  
          
clauses   
    run():-   
        console::init(),   
        stdio::write("% add check\n"),   
        add([5,10,13,11],12,R),stdio::write(R), stdio::nl(),   
        add([],12,S),stdio::write(S), stdio::nl(),   
        stdio::write("% remove check\n"),  
        remove([5,10,13,11],14,A),stdio::write(A), stdio::nl(),   
        remove([5,10,13,11],13,B),stdio::write(B), stdio::nl(),   
        remove([5,10,13,11],5,C),stdio::write(C), stdio::nl(),   
        stdio::write("% concat check\n"),   
        concat([10,14],[1,3,11],E),stdio::write(E), stdio::nl(),   
        concat([10,14],[],F),stdio::write(F), stdio::nl(),   
        stdio::write("% length check\n"),   
        length([10,11,12,13,14,15,16],L),stdio::write(L), stdio::nl(),   
        length([],L2),stdio::write(L2), stdio::nl(),   
        length([11,12,13,14,15,16],L1),stdio::write(L1), stdio::nl(),   
        stdio::write("% contains check\n"),  
        contains([10,11,12,13,14,15],16),stdio::write("Contains\n"),!;stdio::write("Not contains\n"),   
        stdio::write("% minus check\n"),succeed().  
end implement main   
goal   
    mainExe::run(main::run).

**Задание по варианту 15.**

Реализуем предикат - Вычесть 1-цу от каждого элемента списка.

Дерево И/ИЛИ:

minus([H1|T1], [H2|T2])

H2 = H1 - 1

minus(T1, T2)

Текст программы:

minus([], []).  
minus([H1|T1], [H2|T2]) :-  
        H2 = H1 - 1,  
        minus(T1, T2).

clauses   
    run():-

minus([5,10,13,11],A1),stdio::write(A1), stdio::nl(),succeed().  
end implement main   
goal   
    mainExe::run(main::run).

Результат выполнения программы:



**Задание по варианту 4.**

Реализуем предикат - Определить отношение перевод(Список1, Список2) для перевода списка чисел от 0 до 9 в список соответствующих слов нуль, один и т.д.

Дерево И/ИЛИ:

perev([H1|T1], [H2|T2])

a2r(H1,H2)

perev(T1, T2)

Текст программы:

class predicates  
    a2r :(integer, string) determ(i,o).  
    perev :(list,list1) determ(i,o).  
clauses  
    classInfo(className, classVersion).   
    a2r(0, "Нуль  ").  
    a2r(1, "Один  ").  
    a2r(2, "Два  ").  
    a2r(3, "Три  ").  
    a2r(4, "Четыре  ").  
    a2r(5, "Пять  ").  
    a2r(6, "Шесть  ").  
    a2r(7, "Семь  ").  
    a2r(8, "Восемь  ").  
    a2r(9, "Девять  ").  
    perev([],[]).  
    perev([H1|T1], [H2|T2]) :-  
         a2r(H1,H2),  
        perev(T1,T2).  
  
clauses   
    run():-   
        console::init(),   
        stdio::write("% perev check\n"),  
        perev([1,2,3,4],A1),stdio::write(A1),stdio::nl(),fail().  
    run():-  
        console::write("End").  
          
end implement main   
goal   
    mainExe::run(main::run).Результат выполнения программы:



**Задание по варианту 6.**

Реализуем предикат - Определите два предиката четнаядлина**(**Список**)** и нечетная длина**(**Список**)** таким образом, чтобы они были истинными, если их аргументом является список четной или нечетной длины соответственно. Например, список [a,b,c,d ] имеет четную длину.

Дерево И/ИЛИ:

list\_even\_count(List,R)

length(List, Length)

R = "Even"

0 = Length mod 2

list\_odd\_count(List,R)

not(list\_even\_count(List,\_))

R = "Odd"

Текст программы:

class predicates  
  
    length :(list,integer) procedure(i,o).   
     list\_even\_count :(list,string) determ(i,o).  
     list\_odd\_count :(list,string) determ(i,o).  
clauses  
    classInfo(className, classVersion).   
    length([],0):-!.  
    length([Head|Tail],R):-length(Tail,R1),R=R1+1,!.   
    list\_even\_count(List,R) :-  
        length(List, Length),  
        0 = Length mod 2,  
        R = "Even".  
    list\_odd\_count(List,R) :-  
        not(list\_even\_count(List,\_)),  
        R = "Odd".  
          
clauses   
    run():-   
        console::init(),   
        stdio::write("% even, odd, check\n"),

        list\_even\_count([1,2,3,4],A1),stdio::write(A1),stdio::nl(),

    run():-  
        console::write("End").  
          
end implement main   
goal   
    mainExe::run(main::run).

Результат выполнения программы:



**Выводы**

Вданной лабораторной работе были изучена реализация структуры данных типа список в языке Пролог и методами их обработки.