Отчет по лабораторной работе №4

По курсу логического программирования.

Работу выполнила

студентка группы 8О-208Б

Понагайбо Анастасия Олеговна

Сдано: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Тема:**

Обработка естественного языка.

**Цель работы:**

Познакомиться на практике с методами анализа естественно-языковых текстов в системах логического программирования, реализовать в соответствии с вариантом задания несложный фрагмент естественного-языкового интерфейса к модельной задаче и протестировать его на ряде примеров.

**Задание:**

Вариант № 4. Реализовать синтаксический анализатор арифметического выражения для перевода его в префиксную форму. В выражении допустимы операции +,-,\*,/.

**Использованные версии языка Prolog:**

Strawberry Prolog, 3.0 Beta 4.

**Текст программы:**

operation("+",0).

operation("-",0).

operation("\*",1).

operation(":",1).

% разбиение выражения на подвыражения

expr([Num|T],T,number(Num),2).

expr(Input,Tail,Expr,N):-

N=<1,

N1 is N+1,

expr(Input,Tail1,Expr1,N1),

subexpr(Tail1,Tail,Expr1,Expr,N).

% рекурсивная обработка подвыражений

subexpr([O|T],Tail,Expr1,Expr,N):-

operation(O,N),

N1 is N+1,

expr(T,List1,Expr2,N1),!,

interpret(O,Expr1,Expr2,Term),

subexpr(List1,Tail,Term,Expr,N).

subexpr(List,List,Exp,Exp,\_).

% связывание аргументов определенной операцией

interpret("+",Left,Right,add(Left,Right)).

interpret("-",Left,Right,subtract(Left,Right)).

interpret("\*",Left,Right,multiply(Left,Right)).

interpret(":",Left,Right,divide(Left,Right)).

% перевод результата обработки в список

modify(add(Left,Right),["+"|List]):-

modify(Left,LL),

modify(Right,LR),

append(LL,LR,List).

modify(subtract(Left,Right),["-"|List]):-

modify(Left,LL),

modify(Right,LR),

append(LL,LR,List).

modify(multiply(Left,Right),["\*"|List]):-

modify(Left,LL),

modify(Right,LR),

append(LL,LR,List).

modify(divide(Left,Right),[":"|List]):-

modify(Left,LL),

modify(Right,LR),

append(LL,LR,List).

modify(number(N),[N]).

%вычисление префиксной формы

calculate(Input,Res):-

expr(Input,\_,Term,0),

modify(Term,Res).

?- L1 = [10],

write("\*\* convert "), write(L1), write(" into prefix \*\*\n"),

calculate([10],Res1),

write(Res1),

L2 = [1,"+",3,"\*",2, "-", 7, ":", 6],

write("\n\n\*\* convert "), write(L2), write(" into prefix \*\*\n"),

calculate(L2,Res2),

write(Res2),write("\n").

**Вывод программы:**

\*\* convert [10] into prefix \*\*

[10]

\*\* convert [1, +, 3, \*, 2, -, 7, :, 6] into prefix \*\*

[-, +, 1, \*, 3, 2, :, 7, 6]

Yes.

No.

**Грамматика:**  
*Терминальный алфавит:*

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, \*, :.

*Нетерминальный алфавит:*

add, subtract, multiply, divide, number, expr, subexpr, interpret, modify.

*Правила перехода:*

expr→subexpr, number

expr→number

subexpr→expr

subexpr→subexpr, interpret

*Начальный символ:*

expr.

**Выводы:**

Благодаря языку Prolog можно компактно реализовать синтаксические анализаторы выражений и естественно-языковых текстов для их дальнейшей обработки. Реализация подобного анализатора на императивном языке была бы более громоздкой.