

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

РЕШЕНИЕ ЛОГИЧЕСКИХ ЗАДАЧ НА ЯЗЫКЕ ПРОЛОГ

Отчет по лабораторной работе №5

По дисциплине

«Функциональное и логическое программирование»

Студент гр. 431-3

_____ Д.П. Андреев

« ____ » _____ 2024 г.

Проверил: доцент кафедры АСУ

_____ С.М. Алфёров

« ____ » _____ 2024 г.

Томск 2024

1 Цель работы

Получить навыки логического программирования.

2 Задание на лабораторную работу

Написать программу в соответствии с вариантом. Вариант 3: Упрощение логического выражения по заданным тождествам.

3 Листинг программы

% Результат 0

```
rule(X, 0):-  
    (X=0*_);  
    (X=_*0);  
    (X=(-A)*A);  
    (X=A*(-A)).
```

% Результат 1

```
rule(X, 1):-  
    (X=1+_*);  
    (X=_+1);  
    (X=(-A)+A);  
    (X=A+(-A)).
```

% Результат только A

```
rule(X, A):-  
    (X=1*A);  
    (X=A*1);  
    (X=0+A);  
    (X=A+0);  
    (X=(-(-A)));  
    (X=A*A);  
  
    (X=A+A);  
    (X=A+(A*_*));  
    (X=(A*_*)+A);  
    (X=A+(_*A));  
    (X=(_*A)+A);  
  
    (X=A*(A+_*));  
    (X=(A+_*)*A);  
    (X=A*(_*+A));  
    (X=(_*+A)*A);  
    (X=A*B+A*(-B)).
```

% Двойное отрицание

```
rule(-(-A), A).
```

```

%Де Моргана
rule(-(A*B), (-A)+(-B)).
rule(-(A+B), (-A)*(-B)).

% Дистрибутивность
rule(A*(B+C), A*B+A*C).
rule((A+B)*(C+D), A*C+A*D+B*C+B*D).
rule((A+B)*C, A*C+B*C).

% Свертка
rule(A+(-A)*B, A+B).
rule((-A)+A*B, (-A)+B).

% Расширение
rule(A*B+(-A)*C+B*C, A*B+(-A)*C).

rule(X, X).

expression(X, A):- rule(X, A).

expression(X+Y, A+B):-
    simplify(X, A),
    simplify(Y, B).
expression(X*Y, A*B):-
    simplify(X, A),
    simplify(Y, B).

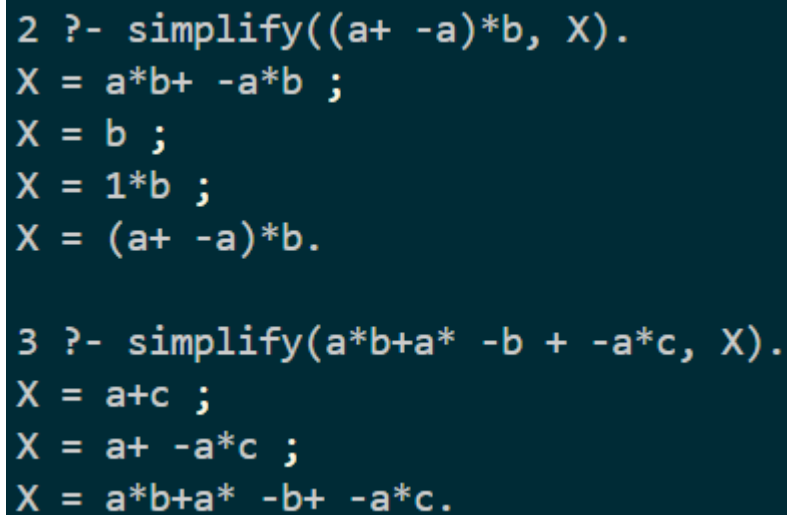
simplify(X, Y):-
    expression(X, Z),
    (X\=Z ->
        simplify(Z, Y)).
simplify(X, X).

```

4 Ход работы

При выполнении задачи программа, условно, проходит три уровня. Первый уровень — точка входа, правило `simplify(X, Y)`, где X — входное выражение, а Y точка перебора подходящих вариантов, которые далее отсеиваются. Если X отлично от Y — происходит косвенная рекурсия посредством вызова правила `expression(X+*Y, A+*B)`, которая разделяет выражение на составные части до и после знака операции. К каждой части применяется снова правило `simplify(X, Y)`. Когда выражение невозможно поделить по операциям — выполняется правило `expression(X, A)`, откуда вызываются базовые тождества, по которым и происходит сокращение выражения. Определены такие тождества как дистрибутивность, закон Де Моргана, правило свертки, а также правила приведения к нулю, единице и одному операнду. Сокращение заканчивается, когда `expression()` возвращает в правило `simplify` значение эквивалентное тому, что было до вызова правила `simplify`.

При запуске программы в консоли мы получим такой результат (рисунок 4.1).



```
2 ?- simplify((a+ -a)*b, X).
X = a*b+ -a*b ;
X = b ;
X = 1*b ;
X = (a+ -a)*b.

3 ?- simplify(a*b+a* -b + -a*c, X).
X = a+c ;
X = a+ -a*c ;
X = a*b+a* -b+ -a*c.
```

Рисунок 4.1 – Результат работы программы

5 Вывод

В ходе выполнения лабораторной работы я получил навыки логического программирования.