Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

по дисциплине: «Введение в искусственный интеллект»

Студент	Косенков Александр Александрович
Группа	РК6-12М
Тип задания	Лабораторная работа №2
Тема лабораторной работы	Изучение языка Пролог

Студент _____ **Косенков А.А.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Федорук В.Г.**
подпись, дата *фамилия, и.о.*

Оценка _____

Москва, 2021 г.

Оглавление

Задание на лабораторную работу 3
Символьное дифференцирование 3
Пример работы программы. 4
Текст программы. 6

Задание на лабораторную работу

Вариант 50

Написать программу символьного дифференцирования арифметических выражений, представленных в канонической (в виде структур) для языка Пролог форме. При этом единственная переменная (по которой производится дифференцирование) обозначается буквой *x*, а константы - всеми иными строчными (маленькими) буквами.

Символьное дифференцирование

Символьным дифференцированием в математике называется операция преобразования одного арифметического выражения в другое, называемое производной.

Каноническая для языка Пролог форма записи арифметических выражений представляет собой запись выражений в виде структур, где операнды перечисляются в скобках через запятую, а операция записывается перед скобками. Так, арифметическая операция $x + 2$ в канонической форме записывается как *+(x, 2)*. Данная форма соответствует прямой польской нотации записи выражений.

Декларативное программирование символьного дифференцирования представляет собой составление базы правил и фактов, соответствующих математическим формулам правил дифференцирования. Так, обозначая арифметические выражения буквами U и V, а константы буквой c, можно составить следующую базу правил:

$$\frac{dc}{dx} = 0; \tag{1.1}$$

$$\frac{dx}{dx} = 1; \tag{1.2}$$

$$\frac{d(-U)}{dx} = -\left(\frac{dU}{dx}\right); \tag{1.3}$$

$$\frac{d(U+V)}{dx} = \frac{dU}{dx} + \frac{dV}{dx}; \tag{1.4}$$

$$\frac{d(cU)}{dx} = c\left(\frac{dU}{dx}\right); \tag{1.5}$$

$$\frac{d(UV)}{dx} = U\left(\frac{dV}{dx}\right) + V\left(\frac{dU}{dx}\right); \quad (1.6)$$

$$\frac{d\left(\frac{U}{V}\right)}{dx} = \frac{V\left(\frac{dU}{dx}\right) - U\left(\frac{dV}{dx}\right)}{V^2}; \quad (1.7)$$

$$\frac{d(U^c)}{dx} = cU^{c-1}\left(\frac{dU}{dx}\right); \quad (1.8)$$

$$\frac{d(c^U)}{dx} = c^U \ln(c)\left(\frac{dU}{dx}\right); \quad (1.9)$$

$$\frac{d(\ln U)}{dx} = \frac{\left(\frac{dU}{dx}\right)}{U}; \quad (1.10)$$

$$\frac{d(\sin U)}{dx} = \cos U\left(\frac{dU}{dx}\right); \quad (1.11)$$

$$\frac{d(\cos U)}{dx} = -\sin U\left(\frac{dU}{dx}\right); \quad (1.12)$$

$$\frac{d(\tan U)}{dx} = \frac{1}{\cos^2 U}\left(\frac{dU}{dx}\right); \quad (1.13)$$

$$\frac{d(e^U)}{dx} = e^U\left(\frac{dU}{dx}\right). \quad (1.14)$$

Помимо базы правил (1.1 – 1.14) была также разработана база правил для арифметических операций для их корректного символьного вывода в консоль.

Непосредственное символьное дифференцирование осуществляется с помощью разработанного предиката *diff(U, x, R)*, где U – выражение для дифференцирования, x – переменная дифференцирования, R – результирующее выражение.

Пример работы программы.

В качестве тестового выражения для дифференцирования было выбрано выражение:

$$f(x) = \sin 4x^2 + \ln x + x^8 + 8^x. \quad (2)$$

В прямой польской нотации данное выражение имеет следующий вид:

$$f(x)^{forward} = + + + sin * 4^{x2}\ ln\ x\ \hat{}x8\ \hat{}8x.$$

Результат символьного дифференцирования с помощью разработанного предиката представлен на рис. 1.

```
aruko@Aruko-A315-41:~/study/RC6/aiintro/RC6-aiintro/Kosenkov-12-2$ swipl lab2_task9.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- diff(+(+(+(sin(*(4, ^(x, 2))), ln(x)), ^(x, 8)), ^(8, x)), x, R).
R = cos(4*x^2)*(4*(2*x^1))+1/x+8*x^7+8^x*ln(8) .
```

Рис. 1. Пример работы программы.

Для сравнения, было также проведено дифференцирование выражения (2) с помощью сервиса *WolframAlpha* (рис. 2), решение на котором было принято считать эталонным.

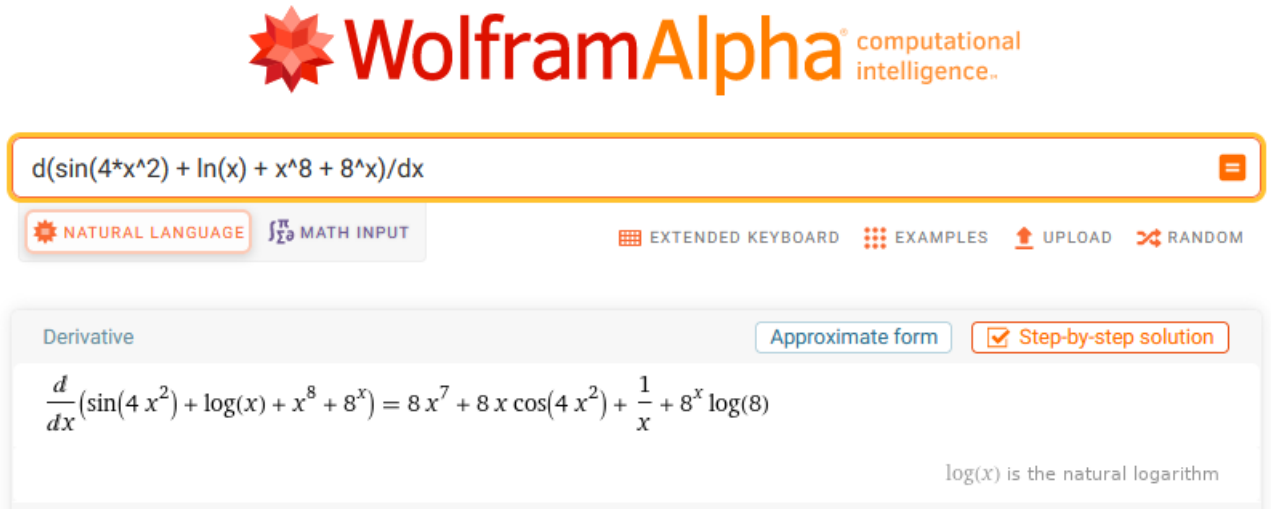

Рисунок 2. Результат дифференцирование тестового выражения с помощью сервиса *WolframAlpha*.

Сравнение результатов на рис. 1 и рис. 2 дает основание полагать о корректности работы разработанного предиката символьного дифференцирования.

Текст программы.

```
% База фактов и правил арифметических операций

% Операция сложения
simp_sum(A, 0, A).
simp_sum(0, A, A).

simp_sum(A, B, Sum) :-
        number(A),
        number(B),
        Sum is A + B.

simp_sum(A, B, A + B).

% Операция вычитания
simp_sub(A, 0, A).
simp_sub(0, A, -A).

simp_sub(A, B, Sub) :-
        number(A),
        number(B),
        Sub is A - B.

simp_sub(A, B, 0) :- A == B.

simp_sub(A, B, A - B).

% Операция умножения
simp_mul(0, _, 0).
simp_mul(_, 0, 0).

simp_mul(A, 1, A).
simp_mul(1, A, A).

simp_mul(A, B, Mult) :-
        number(A),
        number(B),
        Mult is A * B.

simp_mul(A, B, A * B).

% Операции деления
simp_div(_, 0, _) :- throw("Divide by zero error").

simp_div(_, 0, 0).

simp_div(A, 1, A).

simp_div(A, A, 1).

simp_div(A, B, Div) :-
        number(A),
        number(B),
        Div is A / B.

simp_div(A, B, A / B).


% База фактов и правил операций дифференцирования

% d(const)/dx = 0
diff(Eq, _, 0) :- number(Eq).
```

```
% d(y)/dx = 0
diff(Eq, Var, 0) :-
        atom(Eq),
        Eq \== Var.

% d(x)/dx = 1
diff(Eq, Var, 1) :-
        atom(Eq),
        Eq == Var.

% d(u + v)/dx = d(u)/dx + d(v)/dx
diff(+(Eq1, Eq2), Var, Res) :-
        diff(Eq1, Var, Res1),
        diff(Eq2, Var, Res2),
        simp_sum(Res1, Res2, Res).

% d(u - v)/dx = d(u)/dx - d(v)/dx
diff(-(Eq1, Eq2), Var, Res) :-
        diff(Eq1, Var, Res1),
        diff(Eq2, Var, Res2),
        simp_sub(Res1, Res2, Res).

% d(u * v)/dx = v * (d(u)/dx) + u * (d(v)/dx)
diff(*(Eq1, Eq2), Var, Res) :-
        diff(Eq1, Var, Res1),
        diff(Eq2, Var, Res2),
        simp_mul(Eq2, Res1, L),
        simp_mul(Eq1, Res2, R),
        simp_sum(L, R, Res).

% d(u / v)/dx = (v * (d(u)/dx) - u * (d(v)/dx)) / (v * v)
diff(/(Eq1, Eq2), Var, Res) :-
        diff(Eq1, Var, Res1),
        diff(Eq2, Var, Res2),
        simp_mul(Eq2, Res1, L),
        simp_mul(Eq1, Res2, R),
        simp_mul(Eq2, Eq2, Denom),
        simp_sub(L, R, Nom),
        simp_div(Nom, Denom, Res).

% d(sin(u))/dx = cos(u) * (d(u)/dx)
diff(sin(Eq), Var, Res) :-
    diff(Eq, Var, Res1),
        simp_mul(cos(Eq), Res1, Res).

% d(cos(u))/dx = -sin(u) * (d(u)/dx)
diff(cos(Eq), Var, Res) :-
    diff(Eq, Var, Res1),
        simp_mul(sin(Eq), Res1, Mult),
        simp_sub(0, Mult, Res).

% d(tg(u))/dx = (1 / cos(u)^2) * (d(u)/dx)
diff(tg(Eq), Var, Res) :-
        diff(Eq, Var, Res1),
        simp_div(Res1, cos(Eq)^2, Res).

% d(exp(u))/dx = exp(u) * (d(u)/dx)
diff(exp(Eq), Var, Res) :-
    diff(Eq, Var, Res1),
        simp_mul(exp(Eq), Res1, Res).

% d(ln(u))/dx = (1 / u) * (d(u)/dx)
```

```
diff(ln(Eq), Var, Res) :-
    diff(Eq, Var, Res1),
        simp_div(Res1, Eq, Res).

% d(f(u)^a)/dx = a * f(u)^(a - 1) * (d(f(u))/dx)
diff(^(Eq, Num), Var, Res) :-
    number(Num),
    diff(Eq, Var, Res1),
    simp_sub(Num, 1, Pow),
    simp_mul(Num, (Eq)^Pow, R),
    simp_mul(R, Res1, Res).

% d(a^(f(u)))/dx = a^(f(u)) * ln(a) * (d(f(u))/dx)
diff(^(Num, Eq), Var, Res) :-
    number(Num),
    diff(Eq, Var, Res1),
    simp_mul(Num^(Eq), ln(Num), R),
    simp_mul(R, Res1, Res).

% d(f(u)^g(u))/dx = f(x)^(g(x) - 1) * (g(x) * d(f(x))/dx + d(g(x))/dx * f(x) *
ln(f(x))
diff(^(Eq1, Eq2), Var, Res) :-
    diff(Eq1, Var, Res1),
    diff(Eq2, Var, Res2),
        simp_sub(Eq2, 1, Pow),
        simp_mul(Eq2, Res1, L),
        simp_mul(Eq1, ln(Eq1), Tmp),
        simp_mul(Tmp, Res2, R),
        simp_sum(L, R, Sum),
        simp_mul(Eq1^(Pow), Sum, Res).
```