|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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](#_Toc91776469)

[Символьное дифференцирование 3](#_Toc91776470)

[Пример работы программы. 4](#_Toc91776471)

[Текст программы. 6](#_Toc91776472)

# 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 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 и , а константы буквой , можно составить следующую базу правил:

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:

В прямой польской нотации данное выражение имеет следующий вид:

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

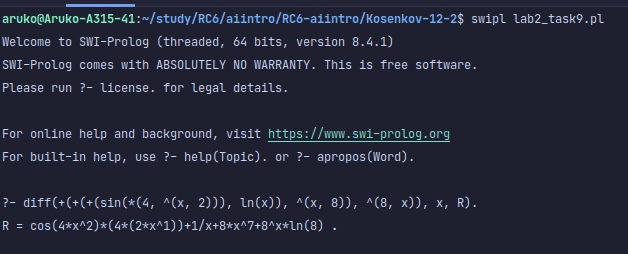


Рис. . Пример работы программы.

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

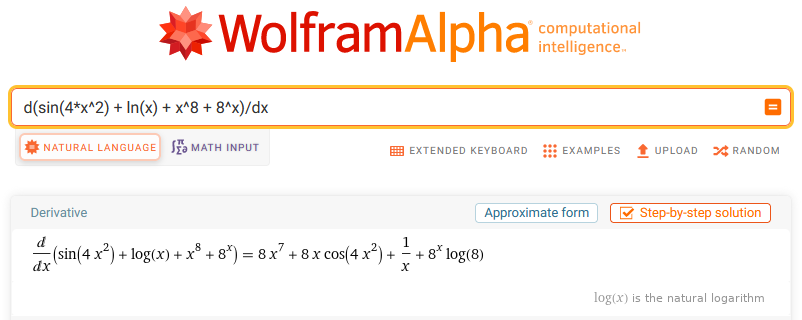


Рисунок . Результат дифференцирование тестового выражения с помощью сервиса *WolframAlpha.*

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).