SPL-1 Project Report

DERIVATIVE CALCULATOR

SE 305: Software Project Lab - 1

Submitted by

Md Nowsad Hossen Munna

BSSE Roll No.: 1407

BSSE Session: 2021-22

Supervised by

Dr. Muhammed Shafiul Alam khan

Designation: Professor

Institute of Information Technology

Institute of Information Technology

University of Dhaka

[17-12-2023]

Project Name: Derivative Calculator

Prepared by:

Md Nowsad Hossen Munna

BSSE 14th batch

Roll -1407

Supervised by:

Dr. Muhammed Shafiul Alam khan

Professor

Institute of Information Technology

University of Dhaka

Date submitted: 17 December, 2023

Supervisor's Approval:

____

(Signature of Dr. Muhammed Shafiul Alam Khan)

Table of Contents

1.Introduction

The derivative is a fundamental concept in mathematics that plays a crucial role in various fields, including calculus, physics, economics, engineering, and many others. Any form of expression can be differentiated to find the rate of change and many more. To ease the process of differential calculus, my project is simply a calculator to find the derivative of a given function. Also for higher secondary students, this calculator shows the process of the differentiation step by step which could be beneficial for their learning purpose.

2.Background Study

Calculus Basics: Differential calculus focuses on the concept of derivatives, which measure the rate at which a function changes. The derivative of a function at a specific point represents its instantaneous rate of change at that point. The derivative is often denoted by symbols such as f'(x), dy/dx, or df/dx.

Differentiation Rules: Differentiation rules in calculus are a set of guidelines and formulas that allow us to find the derivatives of various types of functions. These rules provide systematic methods for calculating derivatives and help simplify the differentiation process. Here are some commonly used differentiation rules:

1. Power Rule: The power rule is used to find the derivative of a function of the form f(x) = x^n, where n is a constant. According to the power rule, the derivative is given by f'(x) = nx^(n-1). For example, if f(x) = x^3, then f'(x) = 3x^2.

2. Constant Rule: The constant rule states that the derivative of a constant is always zero. For example, if f(x) = 5, then f'(x) = 0.

3. Sum and Difference Rule: The sum and difference rule states that the derivative of the sum or difference of two functions is equal to the sum or difference of their derivatives. Mathematically, ((f(x) ± g(x))' = f'(x) ± g'(x). For example, if f(x) = 2x^2 + 3x and g(x) = 4x - 1, then (f(x) + g(x))' = (2x^2 + 3x+4x-1)’ = 4x+7.

4. Product Rule: The product rule is used to find the derivative of the product of two functions. If f(x) and g(x) are two functions, then the derivative of their product f(x)g(x) is given by (f(x)g(x))' = f'(x)g(x) + f(x)g'(x).

5. Quotient Rule: The quotient rule allows us to find the derivative of the quotient of two functions. If f(x) and g(x) are two functions, then the derivative of their quotient f(x)/g(x) is given by [(f'(x)g(x) - f(x)g'(x))/g(x)^2].

6. Chain Rule: The chain rule is used to find the derivative of composite functions. If y= f(g(x)), where f and g are functions, then the derivative of y with respect to x can be

found by multiplying the derivative off with respect to g (f'(g(x))) and the derivative of g with respect to x (g'(x)). Mathematically, dy/dx = f'(g(x)) * g'(x).

7. Trigonometric Functions: There are specific rules for finding derivatives of trigonometric functions. For example, the derivative of sin(x) is cos(x), the derivative of cos(x) is -sin(x), and the derivative of tan(x) is sec^2(x).

8. Exponential and Logarithmic Functions: The derivatives of exponential and logarithmic functions also have specific rules. For example, the derivative of e^x is e^x, the derivative of ln(x) is 1/x, and the derivative of ln(x) (logarithm base e of x) is 1/(x).

9. Square Root Functions: There are specific rules for finding derivatives of square root functions. For example, the derivative of sqrt(x) is 1/(2*sqrt(x)).

String Manipulation: While a derivative calculator may involve parsing and analyzing input expressions, the primary focus is on mathematical operations rather than searching for specific patterns within strings. The algorithmic techniques used in a derivative calculator typically revolve around differentiation rules, symbol manipulation, and recursive evaluation.

Tokenization: when processing mathematical expressions provided as input. Tokenization involves breaking down a string of characters (such as a mathematical expression) into individual tokens, where each token represents a distinct element or component of the expression.

Consider the input expression: "2x^3 + 5"
In this example, the tokenization process involves:

1. Splitting the expression into individual characters or groups: "2", "x", "^", "3", "+","5".

2. Identifying and grouping the tokens: ["2", "x", "^", "3", "+", "5"]

The tokens in this case include the coefficient "2", the variable "x", the exponent symbol "^", the exponent value "3", the plus sign "+", and the constant term "5".

3.Description of the project

The derivative calculator is a software application designed to calculate the derivatives of mathematical expressions. The calculator takes input in the form of mathematical expressions, parses and analyzes them, applies differentiation rules, and provides the corresponding derivative as output. The project aims to provide an efficient and user-friendly tool for performing derivative calculations.

Key Feature:

1. Input Parsing: The calculator accepts mathematical expressions as input, which can include variables, constants, operators, parentheses, and mathematical functions. Ex. (3*x^2 + 2*x + 1)
2. Tokenization: The input expression is tokenized, breaking it down into smaller units or tokens. Tokens represent distinct elements such as numbers, variables and operators.

(3*x^4) → **Diff (int, char, int)**

coefficient =3 **variable = x** **power= 4**

3. Differentiation Rules: The calculator implements a set of differentiation rules, such as the power rule, product rule, quotient rule, and chain rule, to calculate derivatives. Here are examples illustrating how differentiation rules work in this project:

3.1. Power rule: f(x) = x^3. The power rule states that the derivative of x^n is n * x^(n-1). Applying the power rule, the derivative of f(x) = x^3 becomes f'(x) = 3x^2.

3.2. Product Rule: f(x) = x^2 * sin(x). Compute the derivative of the product rule states that the derivative of the product of two functions u(x) and v(x) is given by u'(x) * v(x) + u(x) * v'(x). Applying the product rule, the derivative of f(x) = x^2 * sin(x) becomes f'(x) = 2x * sin(x) + x^2 * cos(x).

3.3. Quotient Rule: f(x) = (x^2 + 1) / x. The quotient rule states that the derivative of the quotient of two functions u(x) and v(x) is given by (u'(x) * v(x) - u(x) * v'(x)) / v(x)^2. Applying the quotient rule, the derivative of f(x) = (x^2 + 1) / x becomes f'(x) = (2x * x - (x^2 + 1) * 1) / x^2 = (x^2 - 1) / x^2. 9

3.4. Chain Rule: f(x) = sin(x^2). The chain rule states that the derivative of a composition of functions f(g(x)) is given by f'(g(x)) * g'(x). Applying the chain rule, the derivative of f(x) = sin(x^2) becomes f'(x) = cos(x^2) * 2x = 2x * cos(x^2).

4.Symbolic Manipulation: The calculator performs symbolic manipulation to simplify expressions and apply differentiation rules algebraically.

5.Output Generation: After parsing, symbolic manipulation, and applying differentiation rules, the calculator generates the derivative of the input expression. It formats and presents the derivative to the user as the final output.

6.User Interface: The calculator features a user-friendly interface that allows users to input expressions, view the calculated derivative, and receive clear and concise results

4. Implementation and Testing:

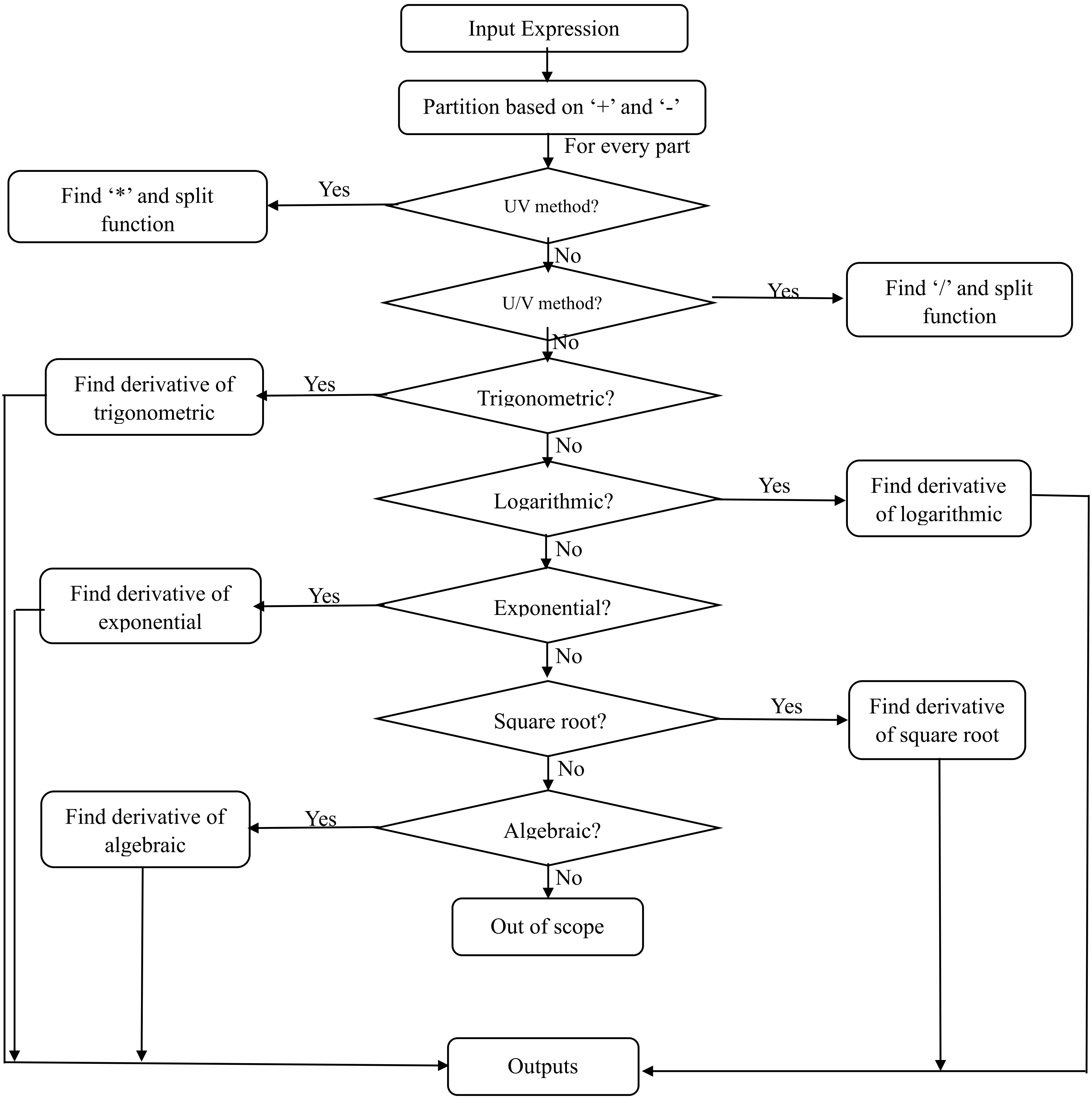


Figure: Control Flow Graph

5. User Manual

1. The lowercase letter 'x' is allowed as the variable. Ex. tan(x), (x^2+5*x-6) etc.

2. Following format has to be maintained while giving an input for these operations-

Operation	Character to be used
Addition	**+**
Subtraction	**-**
Multiplication	*****
Division	**/**
Power	**^**

3. If bracket is needed, properly input the starting bracket and closing bracket.
Ex. ln(tan(x)), (3*x^2-4*x), (tan(x))*(ln(x)) etc.

4. use extra brackets while handling negative signs. Appropriate input format is-
(-3)*x^(-5) where coefficient = -3, variable = x, power = -5.

5. For composite functions, use brackets appropriately to identify the composite parts. Ex. tan(3*x^2) here "("is necessary to separate the composite functions.

6. If the function is a product or division of two functions, the user needs to put a character representing multiplication or division at the correct place. Each function must be included inside of first parentheses. Ex. (3*x^2)*(tan(x)), (2*x)/(ln(x)) etc.

7. To use the square root function, the user needs to put sqrt() which will represent the square root function. Ex. sqrt(3*x^5) etc.

8. No fractions allowed either as coefficients or as powers. Both will be integers.

9. The user needs to use appropriate words to mean appropriate function such as-
Ex. tan(x), cosec(x), ln(x), e^(x), sqrt(x^2) etc.

10. Don't use extra unnecessary bracket.
Ex. (tan(x)+cos(x)) here the first '('and last ')' is unnecessary. So the expression must be given as "tan(x)+cos(x)".

11. The whole project is used '+' and '-' as delimiters.

6. Challenges Faced

Developing a derivative calculator project can come with several challenges.

1. Expression Parsing: Parsing mathematical expressions can be complex, especially when dealing with various operators, parentheses, and precedence rules. Implementing a robust and efficient expression parser can be challenging, requiring careful handling of different cases and error checking.

2. Symbol Manipulation: Manipulating symbols, such as variables, operators, and mathematical functions, requires careful design and implementation. Managing symbol tables, handling algebraic simplifications, and correctly applying differentiation rules can be intricate tasks.

3. Parsing: The input type is String type. But to do the calculation, we need to tokenize the string and parse the coefficients and powers into integers to process it consequently. Again for the output array, the parsing was an important factor.

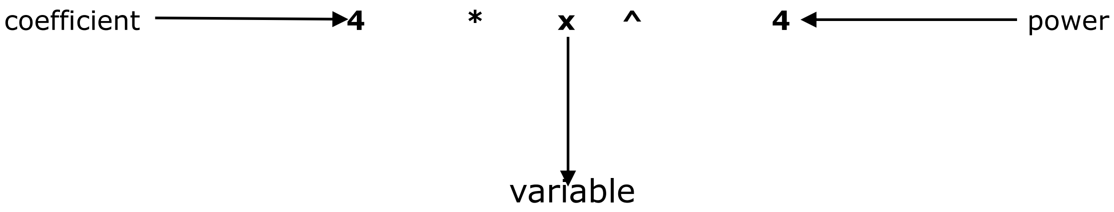
4. Caching: Every turn requires a lot of calculations and those results are often needed for further calculations. Again for new calculations, the previous results need to be cleaned before getting the input.

5. Handling Negative Signs: Given function often has a '-' (minus). Well it can be there before coefficients or as power or just as a minus among two things. To find properly what that specific '-' sign meant, it needed exploring different logical aspects.

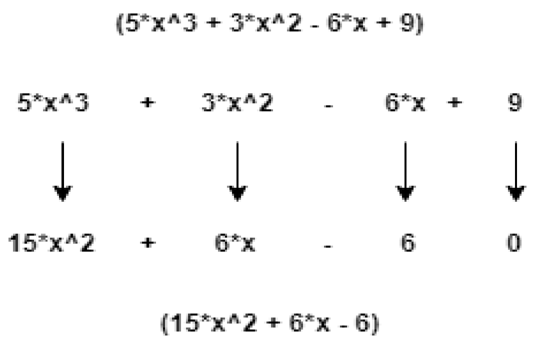
6. Chain-rule Function Type Identification: For composite functions, a specific string such as 'tan', finding if it occurred before or after another string 'ln' was necessary to split the functions. According to the type of function, the string has been sent to further and provide the derivative and thus do it recursively until the chain finish.

As we know what type of input we're dealing with, the input will be assessed according to the type. In case of an algebraic function, extraction of coefficients, powers, signs etc. is necessary to find the derivative. For example-

4*x^4



Here, '+' and '-' are used as delimiters to split parts of a function and every part will be passed as a parameter through that function recursively. The outcome will be stored in a string and will be added as a suffix every time it traverses through the derivative function.



7. Conclusion

In conclusion, the derivative calculator project aims to develop a robust and user-friendly software tool for calculating derivatives of mathematical expressions. By implementing features such as input handling, expression parsing, symbolic manipulation, differentiation rules, error handling, and a user interface, the calculator empowers users to quickly and accurately compute derivatives for various functions and expressions. The project enables users to input mathematical expressions, parse them into tokens, and apply differentiation rules to calculate the derivatives. It supports common mathematical functions and operations, including variables, constants, arithmetic operators, exponentiation, and trigonometric functions. The calculator provides clear and concise outputs, presenting the calculated derivatives in a user-friendly format. Additional features such as higher-order derivatives, partial derivatives, implicit differentiation, graphical visualization, optimization problem solving, input validation, error recovery, history and recall, and user-defined functions enhance the versatility and utility of the derivative calculator. Overall, the derivative calculator project aims to provide a powerful and reliable tool for students, educators, researchers, and anyone working with calculus or mathematical analysis. It simplifies the process of computing derivatives, aids in understanding the behavior of functions, and supports problem-solving in various mathematical contexts.

References

Derivative Formula - What is Derivative Formula? Examples (cuemath.com) , cuemath, (Last accessed on 14/11/23)

What is Tokenization | Tokenization In NLP (analyticsvidhya.com) , analyticsvidhya, (Last accessed on 14/11/23)