User Manual for Math Function Compiler

1. Introduction

Building a specialized compiler is a challenging yet fascinating task, particularly for domains requiring mathematical computation. The goal of this project is to create a Math Function Compiler in Java that processes mathematical expressions with predefined functions (like add, subtract, modulus, etc.) and executes them efficiently.

Traditional tools like calculators are limited in handling complex multistep operations. A custom compiler solves this by allowing users to input mathematical expressions in a high-level human-readable format. The primary challenge lies in creating a system that interprets, validates, and executes user-defined formulas while adhering to a structure, syntax, and handling errors gracefully.

Purpose

- **Tokenize** and analyze user input.
- **Check syntax** correctness using custom parsing rules.
- **Evaluate mathematical operations** using predefined functions.
- **Provide clear error messages** for incorrect syntax or unsupported functions.
- **Enable a user-friendly experience** for executing mathematical expressions efficiently.

---

2. Features

Tokenization

- Identifies keywords, function names, numbers, operators, delimiters, and variables.
- Converts user input into structured tokens for analysis.

Parsing

- Ensures the input follows the correct syntax rules.
- Verifies the structure of function calls and mathematical expressions.

Evaluation

- Executes 50+ predefined mathematical functions including arithmetic, trigonometric, logarithmic, factorial, and combinatorial functions.
- Returns accurate numerical or boolean results.

Error Handling

- Detects and reports errors such as:
    - Incorrect syntax (e.g., missing parentheses, semicolons, or invalid argument types)

- Provides descriptive error messages for easy debugging.

---

3. Supported Data Types

- **int**: Integer values (e.g., 1, 2, 100)
- **double**: Floating point numbers (e.g., 3.14, 2.718)
- **boolean**: True or False values
- **long**: Large integer values
- **double[]**: Array of floating-point numbers

---

4. Functions

Here are the predefined mathematical functions available in the compiler:

1. Arithmetic Functions

- **Add(int a, int b):** Returns the sum of a and b.
- **Subtract(int a, int b):** Returns the difference (a - b).
- **Multiply(int a, int b):** Returns the product (a * b).
- **Divide(double a, double b):** Returns the division result (a / b).
- **Modulus(int a, int b):** Returns the remainder of a / b.
- **Power(double base, double exponent):** Returns base raised to the power of exponent.

2. Square & Cube Operations

- **Squareroot(double a):** Returns the square root of a.
- **Cuberoot(double a):** Returns the cube root of a.
- **NthRoot(double a, int n):** Returns the n-th root of a.

3. Even/Odd Operations

- **IsEven(int a):** Returns true if a is even.
- **IsOdd(int a):** Returns true if a is odd.
- **HalfValue(int a):** Returns half of the value of a.
- **DoubleValue(int a):** Returns double the value of a.
- **Increment(int a):** Increments the value of a by 1.
- **Decrement(int a):** Decrements the value of a by 1.

4. Max/Min Operations

- **FindMax(int a, int b):** Returns the larger of a and b.

- **FindMin(int a, int b):** Returns the smaller of a and b.
- **MaxOfThree(int a, int b, int c):** Returns the largest value among a, b, and c.
- **MinOfThree(int a, int b, int c):** Returns the smallest value among a, b, and `c.

5. Prime and Special Number Checks

- **IsPrime(int a):** Returns true if a is prime.
- **IsPerfectSquare(int a):** Returns true if a is a perfect square.
- **IsPalindrome(int a):** Returns true if a is a palindrome.

6. Trigonometric Functions

- **Sin(double angle):** Returns sin(angle).
- **Cos(double angle):** Returns cos(angle).
- **Tan(double angle):** Returns tan(angle).
- **Cot(double angle):** Returns cot(angle).
- **Sec(double angle):** Returns sec(angle).
- **Cosec(double angle):** Returns cosec(angle).

7. Greatest Common Divisor (GCD) & Least Common Multiple (LCM)

- **Gcd(int a, int b):** Returns the greatest common divisor of a and b.
- **Lcm(int a, int b):** Returns the least common multiple of a and b.

8. Rounding and Absolute Functions

- **AbsoluteValue(double a):** Returns the absolute value of a.
- **Ceil(double a):** Rounds value up to the nearest integer.
- **Floor(double a):** Rounds value down to the nearest integer.
- **Round(double a):** Rounds value to the nearest integer.
- **AbsoluteDifference(int a, int b):** Returns the absolute difference between a and b.

9. Statistical Functions

- **Mean(double[] values):** Returns the mean (average) of the values.
- **SumOfDigits(int a):** Returns the sum of the digits of a.
- **SumOfSquares(int a):** Returns the sum of squares of digits of a.

10. Conversion Functions

- **DegreesToRadians(double degrees):** Converts degrees to radians.
- **RadiansToDegrees(double radians):** Converts radians to degrees.
- **Percentage(double a, double total):** Calculates the percentage of a with respect to total.

11. Area Calculations

- **AreaOfSquare(double side):** Calculates the area of a square given its side length.
- **AreaOfRectangle(double length, double width):** Calculates the area of a rectangle.
- **AreaOfCircle(double radius):** Calculates the area of a circle given its radius.

12. Miscellaneous Functions

- **CubeOfDiff(int a, int b):** Returns the cube of the difference between a and b.
- **AverageOfThree(int a, int b, int c):** Returns the average of a, b, and c.
- **IsMultiple(int a, int b):** Returns true if a is a multiple of b.
- **Reciprocal(double a):** Returns the reciprocal (1/a).
- **ReverseNumber(int a):** Reverses the digits of the number a.

---

5. Error Handling

- **Incorrect syntax:** If parentheses or semicolons are missing, the compiler will flag the error and suggest the correct format.
- **Mathematical errors:** If the function encounters issues like division by zero or invalid operations, it will report an error and offer solutions.

---

6. User Interface Features

- **Function Input Area:** A simple text box where the user can input their function.
- **Run Button:** Executes the input function and shows results or errors.
- **Error Display:** Displays any errors encountered during execution for debugging.
- **User Manual Button:** Provides quick access to the user manual.