Lgebra: A Symbolic Language
CS****: Compiler-II Course Project

Group **

November 18, 2023

Contents

1 Introduction 3

2 Why Lgebra? 3

3 Language Specifications 3

3.1 Keywords . . . 3
3.2 Data Types . . . 3
3.3 Identifiers . . . 3
3.3.1 Rules . . . 3
3.3.2 Reserved Identifiers . . . 4
3.4 Declarations . . . 4
3.4.1 Curves . . . 4
3.4.2 Other Non-Curves . . . 4
3.5 Expression . . . 4
3.5.1 Curve . . . 4
3.5.2 Non-Curve . . . 5
3.6 Constants . . . 5
3.6.1 Built-In constants . . . 5
3.6.2 User-defined constants . . . 5
3.7 Functions . . . 5
3.7.1 Built-In Functions . . . 5
3.7.2 User-defined Functions . . . 6
3.8 Structs . . . 6
3.9 Vectors . . . 6

3.10 Error Analysis 6

4 Other Functionalities 7

4.1 Operator and Function Overloading 7

4.2 Irrational Mathematics 7

5 Compilation Steps 7

6 Performance Analysis 7

7 Drawbacks 7

8 Future Scope 7

9 Conclusion 7

1 Introduction

2 Why Lgebra?

3 Language Specifications

3.1 Keywords

Keywords	Description	Example
if		
else		
until		
repeat		
for		
break		
continue		
return		

3.2 Data Types

Data Types	Description	Example
int		
long		
float		
real		
complex		
vector<Data Type>		
curves		

3.3 Identifiers

3.3.1 Rules

1. All identifiers should start with alphabets

2.

3.3.2 Reserved Identifiers

1. Keywords and Datatype are reserved Identifiers
2. Constants like pi, e, ... are reserverd Identifiers

3.4 Declarations

3.4.1 Curves

1. Curve should be declared as follows

   ```
   curve curve_name(commma seperated variables)
   = Expression in terms of independent variable
   ```

2. Every curve should have atleast one independent variable (like x in f(x))
3. Apart from independent variables, other variable in expression should be declared and defined.
4. By default the return type of function is real. Hence it need not to be mentioned.
5. In following example, both x is different

   ```
   int x = 1;
   curve f(x) = x^2+1;
   ```

3.4.2 Other Non-Curves

1. Other declaration are C like declaration.

3.5 Expression

3.5.1 Curve

1. Curve evaluation syntax is similar to call

- Assume declaration is **curve f(x, y)**
- **f(a)**: Curve f is called with value of x. Is similar to f(x=x)
- **f(a,b)**: Curve f is called with value of x and y
- **f(a,b,c)**: Error. Excess number of arguments
- **f(x=a, y=b)**: Curve f is called with value of x as a and y as b.
- **f(x=a, y=b, z=c)**: Curve f is called with value of x as a, y as b and z as c. **No Error:** z will be substituted be with c. If there is no z then there will be no effect of z=c;

3.5.2 Non-Curve

Similar to C

3.6 Constants

3.6.1 Built-In constants

Name	**Value**	**Description**
e	2.721	Euler Constant

3.6.2 User-defined constants

Explain About Long long constant, float constant , complex constant etc

3.7 Functions

3.7.1 Built-In Functions

1. sum
2. trigonometric functions (return type: curve; arguments: (curve))
   (a) sin
   (b) cos
   (c) tan
   (d) sec

   (e) cosec
   (f) cot

3. curve input_poly(int n)

4. void print_poly(curve c)

5.

3.7.2 User-defined Functions

1. User Defined Function should be defined as follows:

3.8 Structs

1. C like functionalities

3.9 Vectors

1. Explain Operation on Vectors and how to declare it.

3.10 Error Analysis

1. Explain try and catch block

4 Other Functionalities

4.1 Operator and Function Overloading

4.2 Irrational Mathematics

5 Compilation Steps

6 Performance Analysis

7 Drawbacks

8 Future Scope

9 Conclusion