Carlos I. López Sosa
Alejandro Proskauer Valerio
Aryam L. Yulfo Soto
COMP4036-070
Prof. Alcibiades Bustillo
February 26, 2021

Parsertounge

Project-Phase 1

Introduction

With the development of this programming language we propose a language that will help us solve multivariable equations such as those studied in Linear Algebra. We are currently developing this program as if it were a calculator for the users who will use our language. Where the user may input its equations and the language will not only recognize that it is a multivariable expression but it will also have the capability of solving such equations when paired with other expressions. Our language, **Parsertounge**, shares the same library as the programming language Python. The motivation behind this project is simply to facilitate the user's experience when dealing with multivariable expressions, especially, if said user is enrolled in a Linear Algebra course where they would have to solve said equations.

The problem we want to solve for our users with our language is the user's solution accuracy. They may use our language to verify and compare results with precision.

Language Features

As mentioned in the introduction our language **Parsertounge** will use Python in the background. We choose python since it has good mathematical modules plus it is easy to write

and read. Our language will have certain restrictions when it comes to the syntax. Also it will recognize the +, -, = characters. Since our language will solve problems regarding multivariable equations we will restrict the input to only have the letters: X, Y, Z, however, it won't have a restriction in numbers. There are many ways to solve these types of problems but our language will take the equations and transform them into a matrix. The dimensions of the matrix will change depending on the input given. For example if the system of equations given is: 4x + 5y - 4z = 4 , 5x - 5y + 5z = 25 and 9x + 9y + 9z = 81, the matrix will be a 3x3. In some cases the equation will not have a variable,for example 6y + 7z = 65. In this case, our language will read the blank, which will be a whitespace, and turn it into a 0 for the matrix. For example: [0 6 7 | 65].

Example of a Program

User input = x-2y+3z=9; -x+3y=-4;2x-5y+5z=17;

Parsertounge = [[1 -2 3 | 9] , [-1 3 0 | -4], [2 -5 5 | 17]]

Parsertounge = A = [[1 -2 3] , [-1 3 0], [2 -5 5]]

Parsertounge = x = [x y z]

Parsertounge = b = [[9] , [-4], [17]]

This is an example on how the language will behave. The user will input the equations, then our language will transform them into a matrix as shown above. Then it will assign the corresponding values to their respective variables. The language will do the necessary calculations using the mathematical modules in Python to output the correct values.

Implementation Requirements and Tools

As previously mentioned, our language will be implemented using Python. Specifically, we will be using the Anaconda distribution of Python 3.8. To do this, we will be using the *SLY (Sly Lex Yacc)* lexer parser library. We will have to use the *Lexer* class of SLY to allocate tokens to variables, coefficients, constants, and equations. Then we can use *Parser* class to interpret the equations and turn them into matrices.

In addition to SLY, we will also be using some modules from the NumPy library of Anaconda. As mentioned, once the language interprets the systems of equations, it needs to turn them into matrices, and it will do this as NumPy *ndarray* objects. Then, we can use the *linalg* module of NumPy to solve the equations. Then, the output must be displayed.

We also need to use Python's built in exceptions to account for various errors that could occur when interpreting and executing the **Parsertounge** language.

Anaconda Distribution: https://docs.anaconda.com/anaconda/

SLY (Sly Lex Yacc): https://sly.readthedocs.io/en/latest/sly.html

NumPy: https://numpy.org/doc/1.20/contents.html

Project Timeline:

PHASE 1

Project Plan:

PHASE 1-3