Chris Perlette, chris.perlette@uleth.ca

Justin Petluk, justin.petluk@uleth.ca

Ryan Ozar, ryan.ozar@uleth.ca

The program has the main interpreter that allows you to do everything from defining variables to evaluating expressions, however for the most part this interpreter don’t really do anything other than send data to other functions. It calls itself at the end of every function so that it can continue until you enter #exit.

**(UofL)**

This simply starts the program and is case sensitive.

**#definevari & #definefunc**

Using either of the #define options takes in the name of something, and if it’s a variable it takes in the data type and sets the variable to 0, and if it’s a function it just takes in a full function as a string.

**Input**

Input takes in a variable that has already been defined and a value and assigns that value to a variable in the format “input [varName] [data]”.

**Output**

Output takes in a variable and outputs the data associated with that variable in the format “output [varName]”. If there is a and assignment statement, it evaluates the statement and then assigns the variable “[varName] = [expression]”.

**Expressions**

If it is a basic expression, it evaluates it and then outputs the result. If there is a variable in these expressions, it replaces that variable name with the data that variable holds.

**Assignments**

If you make an expression the has an “=” as the first operator the program will evaluate the expression after the “=” and then assign that value to the variable in front of the “=”

**#clear**

This just clears the lists that contain the variables and the functions.

**#exit**

This exits the program

The testing process we used was to start the program, define a set of variables, and then use them in as many ways as possible to completely test the functionality of our program. Ways we did this are better outline in the project document