Polynomial Design Document

Program Flow

1. Handler User's input
2. Keep polynomials formatted correctly
3. Set up library so anyone can use it

Notable Data Structures

- Struct term
- Typedef struct term polynomail

Notable Functions

- struct term * term_create(int coeff, unsigned int exp);
    - Creating the polynomial
- void poly_destroy(polynomial *eqn);
    - Freeing the polynomial
- void poly_print(const polynomial *eqn);
    - Printing the polynomial
- void poly_iterate(polynomial *p, void (*transform)(struct term *));
    - Iterating through the polynomial chain
- polynomial *poly_add(const polynomial *a, const polynomial *b);
    - Adding to the polynomial chain
- polynomial *poly_sub(const polynomial *a, const polynomial *b);
    - Subtracting from the polynomial Chain
- bool poly_equal(const polynomial *a, const polynomial *b);
    - Checking if polynomial is equivalent to another one
- double poly_eval(const polynomial *p, double x);
    - Calculating a polynomial
- char *poly_to_string(const polynomial *p);
    - Making the polynomial into a string and returning it

Anticipated Challenges

1. Math
2. Chaining everything
3. Making it in order
4. Being as efficient as possible

Targeted Features

1. Man Page
2. Unicode Subscripts

Architecture

Making sure that the library can be used with any program that it is linked to and not cause the program to crash.