Homework # 2

The following rules should be observed throughout the assignment:

- Each polynomial should be represented as a singly-linked list (see files “list.h” and “list.c” from the book).
- Each element in the linked list should represent one of the terms in the polynomial
- The data held by each element should be type *double* representingthe constant for that term
- For example, the polynomial $6.0x^3 - 5.3x + 3.1$ would be represented by the linked list $6.0 \rightarrow 0.0 \rightarrow -5.3 \rightarrow 3.1$

a) **(1 point)** Implement a function called *appendTerm:*

void appendTerm(List *pPolynomial, double constant);

This function should append (insert at the end) the value constant to pPolynomial. For example, appending 3.1 to pPolynomial already containing $6.0 \rightarrow 0.0 \rightarrow -5.3$ should result in the value 3.1 being added at the end: $6.0 \rightarrow 0.0 \rightarrow 5.3 \rightarrow 3.1$. If the append fails the program should exit.

hw2 > hw2.cpp > main()

```cpp
#include "list.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>

void appendTerm(List *pPolynomial, double constant) {
    ListElmt *element = pPolynomial->tail;              // Get tail of list

    // Allocate memory for the constant
    double *new_constant = (double *)malloc(sizeof(double));
    if (new_constant == NULL) {                         // In case of invalidd input
        std::cout << "Invalid input found" << '\n';
        exit(EXIT_FAILURE);
    }

    *new_constant = constant;                           // Set new constant variable to given value

    // Append new_constant to end of the list
    if (list_ins_next(pPolynomial, element, new_constant) !=0) {
        std::cout << "Could not append term" << 'n';
        exit(EXIT_FAILURE);
    }
}
```

b) **(2 points)** Implement a function called *display*:

void display(List *pPolynomial);

This function should output the polynomial to stdout in proper polynomial format. For example, displaying polynomial 6.0 → 0.0 → -5.3 → 3.1 should result in $6.0x^3$ - 5.3x + 3.1 being output.

```
void display(List *pPolynomial) {
    ListElmt *element = pPolynomial->head;              // Start with the first term

    int exponent = list_size(pPolynomial) -1;           // Starting/highest exponent is number of exponents -1 because final term is a constant
    int firstTerm = true;

    while (element != NULL) {
        double coeff = *(double *)(element->data);      // coefficient is current term in the list

        // Insert + and - signs in between each term
        if (coeff != 0.0) {                             // Skip all 0.0 values
            if (coeff > 0 && !firstTerm) {              // Exclude very first term
                std::cout << " + ";
            } else if (coeff < 0) {
                std::cout << " - ";
                coeff = -coeff;                         // Remove negative sign as it was printed above
            }

            // Print whole numbers with ".0"
            std::cout << std::fixed << std::setprecision(1);

            // Print correct exponent values on x
            if (exponent == 0) {
                std::cout << coeff;                     // Constant term
            } else if (exponent == 1) {
                std::cout << coeff << "x";              // x with no exponents
            } else {
                std::cout << coeff << "x^" << exponent; // x with corresponding exponents
            }

            firstTerm = false;                              // Set to 0 after first loop (after first term)
        }
        element = element->next;
        exponent--;

    }

    std::cout << std::endl;
}
```

```
int main() {
    List polynomial;

    // Initialize polynomial
    list_init(&polynomial, free);                       // clean up data

    // Append terms
    appendTerm(&polynomial, 6.0);
    appendTerm(&polynomial, 0.0);
    appendTerm(&polynomial, -5.3);
    appendTerm(&polynomial, 3.1);

    // Display polynomial
    display(&polynomial);

    // Clean up list
    list_destroy(&polynomial);

    return 0;
}
```

```
~/Desktop/UCSD DSA C++/hw2 main* ❯ ./hw2
6.0x^3 - 5.3x + 3.1
```

c) **(2 points)** Implement a function called *evaluate*:

double evaluate(List *pPolynomial, double x);

This function should evaluate the polynomial for the given value of x and return the result. For example, displaying polynomial 6.0 → 0.0 → -5.3 → 3.1 and x having value 7.0 the function should return 2024.0 (the result of evaluating $6.0*7.0^3 - 5.3*7.0 + 3.1$).

```
double evaluate(List *pPolynomial, double x) {
    ListElmt *element = pPolynomial->head;
    int exponent = list_size(pPolynomial) -1;
    double result = 0.0;
    while (element != NULL) {
        double coeff = *(double *)(element->data);       // Coefficient is the current term in the list

        // Calculate each term and add to result
        result += coeff * pow(x, exponent);

        // Move to the next term in list and repeat
        element = element->next;
        exponent--;
    }

    return result;
}
```

```
int main() {
    List polynomial;

    // Initialize polynomial
    list_init(&polynomial, free);                        // clean up data

    // Append terms
    appendTerm(&polynomial, 6.0);
    appendTerm(&polynomial, 0.0);
    appendTerm(&polynomial, -5.3);
    appendTerm(&polynomial, 3.1);

    // Display polynomial
    display(&polynomial);

    // Evaluate polynomial for x= 7.0
    double result = evaluate(&polynomial, 7.0);
    std::cout << "Result: " << result << '\n';

    // Clean up list
    list_destroy(&polynomial);

    return 0;
}
```

```
~/Desktop/UCSD DSA C++/hw2 main* ❯ ./hw2
6.0x^3 - 5.3x + 3.1
Result: 2024.0
```

d) **(4 points)** Write a program to test the function from parts a-c. Your test program should demonstrate creating, displaying, and evaluating the following polynomials with the given values for x:

- $x + 1.0$ with x = 1.0
- $x^2 - 1.0$ with x = 2.03
- $-3.0x^3 + 0.5x^2 - 2.0x$ with x = 05.0
- $-0.3125x^4 - 9.915x^2 - 7.75x - 40.0$ with x = 123.45

```
int main() {
    List test1, test2, test3, test4;

    // Initialize polynomials
    list_init(&test1, free);
    list_init(&test2, free);
    list_init(&test3, free);
    list_init(&test4, free);

    // x + 1.0                                  x = 1.0
    appendTerm(&test1, 1);
    appendTerm(&test1, 1.0);
    std::cout << "Test 1: ";
    display(&test1);
    std::cout << "= " << evaluate(&test1, 1.0) << " when x = 1.0" << '\n' << std::endl;

    // x^2 - 1.0                                x = 2.03
    appendTerm(&test2, 1);
    appendTerm(&test2, 0.0);
    appendTerm(&test2, -1.0);
    std::cout << "Test 2: ";
    display(&test2);
    std::cout << "= " << evaluate(&test2, 2.03) << " when x = 2.03" << '\n' << std::endl;

    // -3.0x^3 + 0.5x^2 -2.0x                   x = 05.0
    appendTerm(&test3, -3.0);
    appendTerm(&test3, 0.5);
    appendTerm(&test3, -2.0);
    appendTerm(&test3, 0);
     std::cout << "Test 3: ";
    display(&test3);
    std::cout << "= " << evaluate(&test3, 5.0) << " when x = 05.0" << '\n' << std::endl;

    // -0.3125x^4 - 9.915x^2 - 7.75x - 40.0     x = 123.45
    appendTerm(&test4, -0.3125);
    appendTerm(&test4, 0.0);
    appendTerm(&test4, -9.915);
    appendTerm(&test4, -7.75);
    appendTerm(&test4, -40.0);
     std::cout << "Test 4: ";
    display(&test4);
    std::cout << "= " << std::fixed << evaluate(&test4, 123.45) << " when x = 123.45" << '\n' << std::endl;

    // Clean up lists
    list_destroy(&test1);
    list_destroy(&test2);
    list_destroy(&test3);
    list_destroy(&test4);

    return 0;
}
```

```
~/Desktop/UCSD DSA C++/hw2 main* ❯ ./hw2
Test 1: 1x + 1
= 2 when x = 1.0

Test 2: 1x^2 - 1
= 3.1209 when x = 2.03

Test 3:  - 3x^3 + 0.5x^2 - 2x
= -372.5 when x = 05.0

Test 4:  - 0.3125x^4 - 9.915x^2 - 7.75x - 40
= -72731671.686258 when x = 123.45
```

Final Imports:

```
#include "list.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <iomanip>
#include <cmath>
```