

Reminder!

For all future communications, please use the following SUBJECT LINE:

CIS 27 Fall 2023 YourName : Needs/Questions

Without the above SUBJECT LINE, your emails will be deemed as SPAM/Phishing/Virus and will be deleted.

For all homework submission, please use the following SUBJECT LINE:

cis27Spring2024YourNameHwNumber.c

where **Number** is replaced by the homework number; i.e., 1, 2, 3, etc.

Without the above SUBJECT LINE, your submission emails will be rejected, your homework is considered as not submitted.

Reminder!

- In your program, no GLOBAL DATA are allowed, and you must write all needed functions (no library functions values are allowed).
- Again, writing code is not just the code works. It also involves care, patience, coding idioms + forms, and other reminders. Please see the posted code written in class and the explanation on coding convention/style C file.

Turn In:

Exercise #1 – Due on **Tuesday, April 2, 2024 by 11:00pm as Email Submission**

Homework Due Dates and Consideration:

All homework will be submitted by the given date and time (11:00pm). The homework submissions will be through emailing the work (programs and others) as attachments based on the specified and required formats and structures.

As submitting work through emails, there are time stamps for each email—please keep this in mind as you may want to check your work and submission.

If the submission is after 11:00pm of the given date, then it is considered as late. A late submission within 24 hours will get a 50 % penalty. Between 24 hours and 48 hours late, the penalty will be 75%. After 48 hours late, the submission will be ignored regardless of reasons and a zero (0) will be given.

There will be about 5 to 6 assignments for the semester. Some assignment will have 6-day time; some will have 10 days, and 2+ weeks as specified. Every student must start a homework as soon as it is posted on Canvas—students are responsible to check on Canvas for class information including the Canvas Homepage, Modules, Quizzes, etc.

A lot of hints for assignment are discussed in classes and code samples are posted on Canvas. Everyone will have the same length of time to submit regardless of reasons as a lot of hints are going to be provided. Start to work on homework/assignments early and all work must follow the instructions given in class.

Homework formats/conventions/styles are provided. These coding format, convention, and style must be followed to obtain full credit for all assignments.

More information will be given through code demonstration in class — I do coding live during class meetings and discussions, and I use only Visual Studio IDE!

a) For each exercise, a package must be generated to include the following items:

- Copies of your source files, and header/include files—your source file must be named as

`cis27Spring2024Hw3YourName.c`

`cis27Spring2024Hw3UtilityYourName.h`

`cis27Spring2024Hw3UtilityYourName.c`

`FractionYourName.h`

`FractionYourName.c`

`FractionUtilityYourName.h`

`FractionUtilityYourName.c`

`PolyTermYourName.h`

`PolyTermYourName.c`

`PolyTermUtilityYourName.h`

`PolyTermUtilityYourName.c`

`PolyNodeYourName.h`

`PolyNodeYourName.c`

`PolyNodeUtilityYourName.h`

`PolyNodeUtilityYourName.c`

- Copy of output (copy and paste to the end of your program in the **PROGRAM_OUTPUT** comment block as required)
- Copy of **Logic_Code_Output_Issues** (as a separate comment block) after the **PROGRAM_OUTPUT** block.

b) Emailing each package as follows,

- One email message for each exercise.
- The SUBJECT line of the message should have the following line:

`cis27Spring2024YourNameHw3.c`

- Attaching the source files that was created in Part a).

Note 0!

“YourName” means FirstnameLastname; no abbreviation!

Note 1!

- For all functions written for the homework, you must append “YourName” at the end of the function names—all functions.

For examples, assuming the name is “Nice Effort”,

```
void displayClassInfoFL(void);  
void runMenuHw2FL(void);
```

- Except for the indices of i, j, k, etc., all other variables must have the initials of your first name and last name added to the end of the variable names.

For examples,

```
int usrInputFL;  
int digitCountFL;  
int absValueFL;  
int tmpFL;  
int i, j, k;
```

- You must declare all local variables at the top of the function; one declaration per statement except for the indices that you may have.
- Please be consistent with the above format requirements.
- Please follow all of the code formats, idioms, and practices that have been presented in class as well as in code and coding convention C file (posted on Canvas).
- Penalties will be applied for bad coding and practices as discussed/quizzed/demonstrated throughout class.
- After copying the output to the driver/application file,
 - Do not manually modify/change/insert text for the output of your code!
 - You will get zero (0) for the work! I will run your code and know if you manually alter the pasted output as submitted in your C file!

Note 2!

1. We write code to manipulate data (which are provided by the user) to produce the required outcome in the most efficient way!
2. Getting the program to work is not enough to earn full credit. Your program must run correctly and follow all proper convention and consistent styles as explained in class in order to receive credit accordingly.
3. Writing code is not just the code works. It also involves patience, care and code idioms + forms along with others. Please see the posted code that have been written in class as well as the coding convention C file — posted on Canvas.

Note 3!

1. You will get zero (0) points if your code does not compile! Please make sure that you compile your code frequently and properly throughout the working session. Please check and run your submission again exactly as you just submitted.

2. You are only allowed to use four (4) functions of `stdio.h` and `stdlib.h` for `printf()`, `scanf()`, `malloc()`, and `free()`. All other functions must be written by you!
3. A `runMenuHwNumberFL()` function has a menu that MUST BE a combination of `do-while` and `switch`. Any other form of the menu will receive zero (0) for the whole homework.
4. You will be penalized heavily if there are violations on code conventions as explained in class, homework submissions, quizzes, and document posted on Canvas!
5. Your code must work with all reasonable data sets/patterns. At least, your code should be tested with the given data samples indicated/given from the exam/document, etc.
6. Pay attention to naming; that means the specified/required names, your own generated names, and filenames. Penalty points will apply if you do not follow the instructions.
7. Again — You are only allowed to use four (4) functions of `stdio.h` and `stdlib.h` for `printf()`, `scanf()`, `malloc()`, and `free()`. All other functions must be written by you!
8. All function names must have the initials of your firstname and lastname appended at the end.
9. The following functions are absolutely bad in my compilation — “cannot compile” error! Do not ever use them in my classes.

```
void doNotUseArraySyntax01() {
    int arySize1;

    printf("\nEnter an int for array size: ");
    scanf("%d", &arySize1);

    int ary2[arySize1];

    // TODO

    return;
}

void doNotUseArraySyntax02() {
    int arySize2 = 5;
    int ary2[arySize2];

    // TODO

    return;
}
```

10. Q.E.D.

Reminder!

- In your program, no GLOBAL DATA are allowed, and you must write all needed functions (no library functions values are allowed).

- Again, writing code is not just the code works. It also involves care, patience, coding idioms + forms, and other reminders. Please see the posted code written in class and the explanation on coding convention/style C file.

+++++

Again, consider the following is yours!

We write code to manipulate data (which are provided by the user) to produce the required outcome in the most efficient way!

For Code Convention and Style -

Again, writing code is not just the code works. It also involves care, patience, coding idioms + forms, and other reminders. Please see the posted code written in class and the explanation of coding convention C/CPP file.

AND

If writing in C, you are only allowed to use printf() and scanf() from the stdio.h header. You will write everything else by yourself.

If writing in C++, you are only allowed to use cout and cin from the iostream header. You will write everything else by yourself.

ALSO,

- Including PROGRAM COMMENT block!
- Inserting required Comment lines!
- Removing unnecessary blank lines!
- If needed, 1 blank line is sufficient!
- Keeping lines to be reasonable short; for examples, around 65 characters!
- For functions without arguments, insert void in the function prototypes!
- For functions without arguments, remove void in the argument list of the function definitions!
- Keeping indentation levels with the same consistently!
- Declaring all variables at the top of function!
- Declaring one variable per line except for the indices!
- Using better name!
- Following the naming rules!
- Removing all comments that are not required!
- Inserting proper spaces around '(', ')', '{',

and operators!

- Using proper code idioms, and efficient operators and styles!
- Paying attention to the exact layout of the required output to earn full credits!
- Replacing FL with the initial of your first-name and last-name!

For Operational Code -

- No global variables/values!
- No extern setups!
- Menu must be a combination of do - while and switch structures, and it has no arguments!
- Use initialization with proper initial value!
- If C++, you must use proper uniform initialization.
- Variables may not need to be initialized!
- If it is C++ coding, then
"string" class is not allowed in this class. For the classes that are not written by you, only ostream (for cout) and istream (for cin) are allowed. You are only allowed to use cout and cin from the iostream header.
- Again, uniform initialization!
- All pointers must have values; that means initial address or setting address.
- After releasing the dynamic block, its pointer must be reset to NULL (in C) or nullptr (in C++).
- Dynamic objects and data->Memory management!
- In C, you are allowed to use only malloc() and free().
- In C++, you must use "new" and "delete" operators
- In C/C++, do not write
 someVariable = -1 * somevariable;
 someIncrement = someIncrement + 1;
 if (someTestingValue != 0)
 while (someTestingValue != 0)
- In C/C++, do write
 someVariable = -somevariable;

```
someIncrement++;  
if (someTestingValue)  
while (someTestingValue)
```

- Use proper code idioms, and efficient operators and styles!
- Write your own code!
- Except for the member data, member functions, function arguments and local variables within member functions, and indices of i, j, k, etc., all other variables must have the initials of your first-name and last name added to the end of the variable names.
- For examples,

```
int usrInputFL;  
int digitCountFL;  
int absValueFL;  
int tmpFL;  
  
// Function Prototypes  
  
void doGood(void);  
void displayDigit(int);  
  
// Function Definitions  
  
void doGood() {  
  
    // Function Body  
}  
  
void displayDigit(int arg) {  
  
    // Function Body  
}
```

- All stand - alone function names must have the initials of your first-name and last-name appended at the end.
- All filenames must have your complete first-name and last-name appended as required.

For OUTPUT and Copy of OUTPUT -

After copying the output to the driver, do not manually modify / change / insert text for the output of your code! You will get zero (0) for the work! I will run your code and know if you manually alter the pasted output as submitted in your C / C++ (and header) files!

For LOGIC_CODE_OUTPUT_Issues block -

Do not forget this block!

1. Coding Assignment

Exercise 1 – Due Tuesday, April 2, 2024 by 11:00pm as Email

- (1) Write a C program with call to functions to produce the output given below.
- (2) The program should first display the output to screen (from function calls) as

We write code to manipulate data (which are provided by the user) to produce the required outcome in the most efficient way!

CIS 27 - Data Structures and Algorithms
Laney College
Your Name

Information --

Assignment:	HW #3 Exercise #1
Implemented by:	Your Name
Required Submission Date:	2024/03/31
Actual Submission Date:	____/____/____

Where “Your Name” means **Firstname Lastname**; no abbreviation!

For examples, if your name is **First Last** then Your Name should be First Last throughout all of your work/code as mentioned.

- (3) The program will have the setup as follows,

A. struct FractionYourName is defined as below.

```
/**
 * Program Name: fractionYourName.h
 * Discussion:   Specification File
 *              struct Fraction & Relevance
 * Written By:   Your Name
 * Date:         2024/____/____
 */

#define _CRT_SECURE_NO_WARNINGS

#ifndef FRACTIONYOURNAME_H
#define FRACTIONYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>

struct FractionFirstLast {
    int num;
    int denom;
};
```

```
// typedef
typedef struct FractionFirstLast TdFractionFL;
typedef TdFractionFL* TdFractionPtrFL;
typedef TdFractionFL* TdFractionAddrFL;

// Function Prototypes

TdFractionAddr createFractionFL(void);
int removeFractionFL(TdFractionAddr);
void printFraction(const TdFractionAddrFL);

// Updated and Additional Function Prototypes

#endif
```

- B. The set of functions to support your **struct** `FractionYOURNAME` is given in the header file below.

```
/**
 * Program Name: fractionUtilityYourName.h
 * Discussion:   Specification File
 *               Support Functions for Fraction
 * Written By:   Your Name
 * Date:         2024/__/__
 */

#ifndef FRACTIONUTILITYYOURNAME_H
#define FRACTIONUTILITYYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>
#include "fractionYourName.h"

// Function Prototypes

int gcdFL(int, int);

// Updated and Additional Function Prototypes

#endif
```

- C. **struct** `PolyTermYOURNAME` is defined as below.

```
/**
 * Program Name:   polyTermYourName.h
 * Discussion:     Specification File
 *               struct Fraction & PolyTerm Relevance
 * Written By:     YourName - First + Space + Last
 * Submitted Date: 2024/__/__
 */

#ifndef POLYTERMYOURNAME_H
#define POLYTERMYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>
```

```
#include "fractionYourName.h"

struct PolyTermYOURNAME {
    int order;
    struct FractionYOURNAME coeff;
};

// Function Prototypes

struct PolyTermYOURNAME* createPolyTermYOURNAME(void);

// will be updated

#endif
```

D. **struct PolyTermUtilityYOURNAME** is defined as below.

```
/**
 * Program Name:    polyTermUtilityYourName.h
 * Discussion:      Specification File
 *                  struct Fraction & PolyTerm Relevance
 * Written By:      YourName - First + Space + Last
 * Submitted Date:  2024/__/__
 */

#ifndef POLYTERMUTILITYYOURNAME_H
#define POLYTERMUTILITYYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>
#include "fractionYourName.h"
#include "polyTermYourName.h"

// Function Prototypes

// will be updated

#endif
```

E. **struct PolyNodeYOURNAME** is defined as below.

```
/**
 * Program Name:    polyNodeYourName.h
 * Discussion:      Specification File
 *                  struct Fraction & PolyTerm Relevance
 * Written By:      YourName - First + Space + Last
 * Submitted Date:  2024/__/__
 */

#ifndef POLYNODEYOURNAME_H
#define POLYNODEYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>
```

```
#include "fractionYourName.h"
#include "polyTermYourName.h"

struct PolyNodeYOURNAME {
    struct PolyTermYOURNAME* termPtr;
    struct PolyNodeYOURNAME* next;
};

// Function Prototypes

struct PolyNodeYOURNAME* createPolyNodeYOURNAME(void);

// Will be updated

#endif
```

F. **struct PolyNodeUtilityYOURNAME** is defined as below.

```
/**
 * Program Name:    polyTermUtilityYourName.h
 * Discussion:      Specification File
 *                  struct Fraction & PolyTerm Relevance
 * Written By:      YourName - First + Space + Last
 * Submitted Date: 2024/__/__
 */

#ifndef POLYTERMUTILITYYOURNAME_H
#define POLYTERMUTILITYYOURNAME_H

// Header/include File
#include <stdio.h>
#include <stdlib.h>
#include "fractionYourName.h"
#include "polyTermYourName.h"
#include "polyNodeYourName.h"

// Function Prototypes

// will be updated

#endif
```

G. The set of functions to support HW #3 is given in the header file below.

```
/**
 * Program Name:    cis27Spring2024YourNameHw3Utility.h
 * Discussion:      Specification File
 *                  Support Functions for HW #3
 * Written By:      Your Name
 * Date:           2024/__/__
 */

#ifndef CIS27SPRING2024YOURNAMEHW3UTILITY_H
#define CIS27SPRING2024YOURNAMEHW3UTILITY_H

// Header/include File
```

```
#include <stdio.h>
#include <stdlib.h>
#include "fractionYourName.h"
#include "fractionUtilityYourName.h"

// Function Prototypes

void displayCodingStatementFL(void);
void displayClassInfoFL(void);
void runMenuHwFL(void);
void displayFractionInfoFL(TdFractionPtrFL);
void initSubmenu(/*To Be Updated*/);

// Updated and Additional Function Prototypes

#endif
```

H. The program will then continue to call other functions and display the results as follows,

```
// OUTPUT - Sample Run
We write code to manipulate data (which are
provided by the user) to produce the required
outcome in the most efficient way!

CIS 27 - Data Structures and Algorithms
Laney College
Your Name

Information --
Assignment: HW #3 Exercise #1
Implemented by: Your Name
Required Submission Date: 2024/04/02
Actual Submission Date: 2024/__/__

*****
* MENU - HW #3 *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials *
* (3) Adding 2 Polynomials *
* (4) Multiplying 2 Polynomials *
* (5) Displaying 2 Polynomials *
* (6) Display Resulting Polynomial *
* (7) Quit *
*****
Enter an integer for option + ENTER: 9

Wrong Option!

*****
* MENU - HW #3 *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials *
* (3) Adding 2 Polynomials *
* (4) Multiplying 2 Polynomials *
* (5) Displaying 2 Polynomials *
* (6) Display Resulting Polynomial *
* (7) Quit *
```

Enter an integer for option + ENTER: 6

Not appropriate as there are no Polynomials!

```
*
*          MENU - HW #3          *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials      *
* (3) Adding 2 Polynomials          *
* (4) Multiplying 2 Polynomials     *
* (5) Displaying 2 Polynomials      *
* (6) Display Resulting Polynomial  *
* (7) Quit                          *
*                                  *
```

Enter an integer for option + ENTER: 1

Initializing 2 Polynomials -

Calling init() -

```
*****
*          init() Submenu          *
* (1) Creating 2 Polynomials      *
* (2) Updating Polynomial 1      *
* (3) Updating Polynomial 2      *
* (4) Displaying 2 Polynomials   *
* (5) Return                     *
*                                  *
```

Enter an integer for option + ENTER: 4

Wrong Option!

```
*****
*          init() Submenu          *
* (1) Creating 2 Polynomials      *
* (2) Updating Polynomial 1      *
* (3) Updating Polynomial 2      *
* (4) Displaying 2 Polynomials   *
* (5) Return                     *
*                                  *
```

Enter an integer for option + ENTER: 2

Not appropriate as there are no Polynomials!

```
*****
*          init() Submenu          *
* (1) Creating 2 Polynomials      *
* (2) Updating Polynomial 1      *
* (3) Updating Polynomial 2      *
* (4) Displaying 2 Polynomials   *
* (5) Return                     *
*                                  *
```

Enter an integer for option + ENTER: 1

Creating 2 Polynomials -

Calling create() -

```
*****
*       create() Submenu       *
* (1) Creating 2 Polynomials   *
* (2) Displaying 2 Polynomials *
* (3) Return                   *
*****
```

Enter an integer for option + ENTER: 1

Removing existing 2 Poly if existing!

Creating 2 new Poly
Creating Poly #1 -

Is there a term (1 : yes, 0 : no)? 1

Enter expo: 3

Creating coeff Fraction:

Enter num: 1

Enter denom: 4

Is there a term (1 : yes, 0 : no)? 1

Enter expo: 0

Creating coeff Fraction:

Enter num: 1

Enter denom: -2

Is there a term (1 : yes, 0 : no)? 0

Creating Poly #2 -

Is there a term (1 : yes, 0 : no)? 1

Enter expo: 4

Creating coeff Fraction:

Enter num: 1

Enter denom: 2

Is there a term (1 : yes, 0 : no)? 1

Enter expo: 2

Creating coeff Fraction:

Enter num: 1

Enter denom: -3

Is there a term (1 : yes, 0 : no)? 1

Enter expo: 0

Creating coeff Fraction:

Enter num: 1

Enter denom: 2

Is there a term (1 : yes, 0 : no)? 0

```
*****
*       create() Submenu       *
* (1) Creating 2 Polynomials   *
* (2) Displaying 2 Polynomials *
```


* (3) Return *

Enter an integer for option + ENTER: 2

Displaying 2 Polynomials -

Poly #1 -

Address: 00A51900

Degree: 3

Number of Terms: 2

Term #1 -

Address: 00A51850

Expo: 3

Coefficient -

Address: 00A51800

num: 1

denom: 4

Term #2 -

Address: 00A51750

Expo: 0

Coefficient -

Address: 00A51700

num: -1

denom: 2

Poly #2 -

Address: 00A52900

Degree: 4

Number of Terms: 3

Term #1 -

Address: 00A52850

Expo: 4

Coefficient -

Address: 00A52800

num: 1

denom: 2

Term #2 -

Address: 00A52750

Expo: 2

Coefficient -

Address: 00A52700

num: -1

denom: 3

Term #3 -

Address: 00A52650

Expo: 0

Coefficient -

Address: 00A52600

num: 1

denom: 2

```
*****
*           create() Submenu           *
* (1) Creating 2 Polynomials           *
* (2) Displaying 2 Polynomials         *
* (3) Return                           *
*****
```

Enter an integer for option + ENTER: 3

Return to previous menu!

```
*****
*           init() Submenu              *
* (1) Creating 2 Polynomials           *
* (2) Updating Polynomial 1           *
* (3) Updating Polynomial 2           *
* (4) Displaying 2 Polynomials         *
* (5) Return                           *
*****
```

Enter an integer for option + ENTER: 5

Return to previous menu!

```
*****
*           MENU - HW #3                *
* (1) Creating/Updating 2 Polynomials  *
* (2) Evaluating 2 Polynomials         *
* (3) Adding 2 Polynomials              *
* (4) Multiplying 2 Polynomials         *
* (5) Displaying 2 Polynomials          *
* (6) Display Resulting Polynomial     *
* (7) Quit                             *
*****
```

Enter an integer for option + ENTER: 2

Evaluating Polynomials -

Enter the value (Fraction) to be evaluated with -

num: 1

denom: 1

Poly #1 at x = (1/1): (-1/4)

Poly #2 at x = (1/1): (2/3)

```
*****
*           MENU - HW #3                *
* (1) Creating/Updating 2 Polynomials  *
* (2) Evaluating 2 Polynomials         *
* (3) Adding 2 Polynomials              *
* (4) Multiplying 2 Polynomials         *
* (5) Displaying 2 Polynomials          *
* (6) Display Resulting Polynomial     *
* (7) Quit                             *
*****
```

Enter an integer for option + ENTER: 3

Adding 2 Polynomials -

// TO BE UPDATED

```
*****
*           MENU - HW #3           *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials      *
* (3) Checking for Palindrome       *
* (4) Adding 2 Polynomials          *
* (5) Multiplying 2 Polynomials     *
* (6) Displaying 2 Polynomials      *
* (7) Display Resulting Polynomial  *
* (8) Quit                          *
*****
```

Enter an integer for option + ENTER: 1

Initializing 2 Polynomials -

Calling init() -

```
*****
*           init() Submenu          *
* (1) Creating 2 Polynomials        *
* (2) Updating Polynomial 1         *
* (3) Updating Polynomial 2         *
* (4) Displaying 2 Polynomials      *
* (5) Return                        *
*****
```

Enter an integer for option + ENTER: 2

Updating Polynomial #1

```
*****
*           update() Submenu        *
* (1) Adding one term              *
* (2) Removing one term            *
* (3) Displaying updated Polynomial *
* (4) Return                       *
*****
```

Enter an integer for option + ENTER: 1

Adding 1 term -

Enter expo: 2

Creating coeff Fraction:

Enter num: 1

Enter denom: 1

```
*****
*           update() Submenu        *
* (1) Adding one term              *
* (2) Removing one term            *
* (3) Displaying updated Polynomial *
* (4) Return                       *
*****
```

Enter an integer for option + ENTER: 3

Displaying Poly #1 -

Address: 00A51900

Degree: 3

Number of Terms: 3

Term #1 -

Address: 00A51850

Expo: 3

Coefficient -

Address: 00A51800

num: 1

denom: 4

Term #2 -

Address: 00A51650

Expo: 2

Coefficient -

Address: 00A51600

num: 1

denom: 1

Term #3 -

Address: 00A51750

Expo: 0

Coefficient -

Address: 00A51700

num: -1

denom: 2

* update() Submenu *

* (1) Adding one term *

* (2) Removing one term *

* (3) Displaying updated Polynomial *

* (4) Return *

Enter an integer for option + ENTER: 4

Return to previous menu!

* init() Submenu *

* (1) Creating 2 Polynomials *

* (2) Updating Polynomial 1 *

* (3) Updating Polynomial 2 *

* (4) Displaying 2 Polynomials *

* (5) Return *

Enter an integer for option + ENTER: 5

Return to previous menu!

```

*****
*                MENU - HW #3                *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials *
* (3) Adding 2 Polynomials *
* (4) Multiplying 2 Polynomials *
* (5) Displaying 2 Polynomials *
* (6) Display Resulting Polynomial *
* (7) Quit *
*****
Enter an integer for option + ENTER: 4

```

Adding 2 Polynomials -

// TO BE UPDATED

```

*****
*                MENU - HW #3                *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials *
* (3) Adding 2 Polynomials *
* (4) Multiplying 2 Polynomials *
* (5) Displaying 2 Polynomials *
* (6) Display Resulting Polynomial *
* (7) Quit *
*****
Enter an integer for option + ENTER: 6

```

Displaying the resulting Polynomial -

Address: 00A5B000

Degree: 3

Number of Terms: 3

Term #1 -

Address: 00A5B350

Expo: 4

Coefficient -

Address: 00A5B300

num: 1

denom: 2

Term #2 -

Address: 00A5B250

Expo: 3

Coefficient -

Address: 00A5B200

num: 1

denom: 4

Term #3 -

Address: 00A5B150

Expo: 0

Coefficient -

Address: 00A5B100

num: 2

denom: 3

```
*****
*           MENU - HW #3           *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials      *
* (3) Adding 2 Polynomials          *
* (4) Multiplying 2 Polynomials     *
* (5) Displaying 2 Polynomials      *
* (6) Display Resulting Polynomial  *
* (7) Quit                          *
*****
Enter an integer for option + ENTER: 4
```

Multiplying 2 Polynomials -

Address: 00A5B000

Degree: 7

Number of Terms: 7

Term #1 -

Address: 00A5C750

Expo: 7

Coefficient -

Address: 00A5C700

num: 1

denom: 8

Term #2 -

Address: 00A5C650

Expo: 6

Coefficient -

Address: 00A5C60

num: 1

denom: 2

Term #3 -

Address: 00A5C550

Expo: 5

Coefficient -

Address: 00A5C500

num: -1

denom: 12

Term #4 -

Address: 00A5C450

Expo: 4

Coefficient -

Address: 00A5C400

num: -7

denom: 12

Term #5 -

Address: 00A5C350

Expo: 3

Coefficient -

Address: 00A5C300

num: 1

denom: 8

Term #6 -

Address: 00A5C250

Expo: 2

Coefficient -

Address: 00A5C200

num: 2

denom: 3

Term #7 -

Address: 00A5C150

Expo: 0

Coefficient -

Address: 00A5C100

num: -1

denom: 4

```
*****
*           MENU - HW #3           *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials      *
* (3) Adding 2 Polynomials          *
* (4) Multiplying 2 Polynomials     *
* (5) Displaying 2 Polynomials      *
* (6) Display Resulting Polynomial  *
* (7) Quit                          *
*****
```

Enter an integer for option + ENTER: 6

Displaying the resulting Polynomial -

Address: 00A5C000

Degree: 7

Number of Terms: 7

```
*****
*           MENU - HW #3           *
* (1) Creating/Updating 2 Polynomials *
* (2) Evaluating 2 Polynomials      *
* (3) Adding 2 Polynomials          *
* (4) Multiplying 2 Polynomials     *
* (5) Displaying 2 Polynomials      *
* (6) Display Resulting Polynomial  *
* (7) Quit                          *
*****
```

Enter an integer for option + ENTER: 7

All objects have been removed/deleted through calls to free()!

Have fun!

`cis27Spring2024YourNameHw3.c`

`cis27Spring2024YourNameHw3Utility.h`

`cis27Spring2024YourNameHw3Utility.c`

`FractionYourName.h`

`FractionYourName.c`

`FractionUtilityYourName.h`

`FractionUtilityYourName.c`

`PolyTermYourName.h`

`PolyTermYourName.c`

`PolyTermUtilityYourName.h`

`PolyTermUtilityYourName.c`

`PolyNodeYourName.h`

`PolyNodeYourName.c`

`PolyNodeUtilityYourName.h`

`PolyNodeUtilityYourName.c`

In the above filenames, please provide `YourName` with full firstname and lastname without spaces.

- (5) The **PROGRAM OUT** and **Logic_Code_Output Issues** comment blocks should be added to the application driver, which is `cis27Spring2024YourNameHw3.c`.
- (6) No manual manipulation to the output is allowed; you will get zero (0) for the whole work if any manual manipulation is found!
- (7) Please use the **Logic_Code_Output Issues** to provide any observations and comments as needed.
- (8) Email the source files above using the SUBJECT LINE of
`cis27Spring2024YourNameHw3.c`