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

`CIS 25 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:

`cis25Fall2023YourNameHwNumber.cpp`

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.

Turn In:

1. **Exercise #1** – Due on **Saturday, November 4, 2023 by 11:00 pm as Email Submission**

Homework Due Dates and Consideration:

Students must submit their homework by the given date and time (e.g., Friday on or before 11:00 pm). The homework submissions will be through emailing the work (programs and others) as attachments based on the specified and required formats and structures— only `*.h`, `*.cpp`, no Zip File and no other file formats such as PDF, Word, etc. as you will get zero0) for these non-coding files!

If the homework submission is after 11:00 pm on the given date, then the homework submission is late. A late homework 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 not count regardless of the reasons, and the student will receive zero.

Note that by submitting work through emails, there are time stamps for the emails. If you disagree with being marked late or not graded, you can **forward** a copy of your original email submission (as proof of what and when you emailed) for consideration.

There will be about 5 to 6 assignments with specified dues dates. Note that assignments may have 6-day time, 10-day time, or 2+ weeks to submit. Every student must start homework as soon as it is available on Canvas—students are responsible with checking Canvas for class information from the Canvas Homepage, Modules, Quizzes, etc.

Assignment hints may be presented in classes with code samples posted on Canvas. everyone will have the same amount of time to submit regardless of the reasons—work on homework/assignments early.

Homework formats/conventions/styles are explained in class or provided through Canvas. Students must follow the coding formats, conventions, and styles to obtain full credit for all assignments. More information will be available 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:
   - Copy of your source file (C++ program→ **`*.cpp`**)—your source file MUST BE NAMED as **`cis25Fall2023YourNameHw2.cpp`** with the cpp extension.
   - Note! Replace "**YourName**" with your own full-name; i.e., **FirstnameLastName**.
   - Copy of output (copy and paste to the end of your program as **`PROGRAM_OUTPUT`** comment block)
   - Copy of **`Logic_Code_Output_Issues`** comment block (as a separate comment block) after the **`PROGRAM_OUTPUT`**.

b) Emailing each package as follows,
   - One email message for each exercise.
   - The SUBJECT line of the message should have the following line:

     `cis25Fall2023YourNameHw3.cpp`

   - Attaching the source file that was created in part a).

   Note 0! "YourName" means FirstnameLastname—no abbreviation!

2. **Reminder!**

   Getting the program to work *<u>is not enough to earn full credit</u>*. **Your program must run correctly and follow all proper conventions and consistent styles as explained in class, and produce the exact textual information/display as shown/required (of course, with appropriate user input data/values for each run/selection!) to receive credit accordingly.**

   Also, your program must work with all reasonable data sets or patterns.

   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 coding convention CPP file.

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

4. **Q.E.D.**

****

More Notes!

- **You are only allowed to use `cout` and `cin` from the `iostream` header.**
- **You are not allowed to use any other classes or syntax structures that are not introduced in class and class meetings — please confirm with the instructor if you have any doubt!**

- **You are not allowed to use `classes` and functions written by someone else.**
- **You must write all classes and functions yourself before using them in the required class work.**
- **All `class` and `type` names must have your first and last names appended.**
- **All local variables must be declared at the top of its function.**
- Except for the member data, 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;
  ```
- **All function names must have the initials of your Firstname and Lastname appended at the end.**
- **All filenames must have your complete Firstname and Lastname appended as required.**
- **The values of the elements from the created array, it any, MUST NOT be changed/modified at all during the analysis steps and process unless they are required to be updated or changed based on the specified tasks.**

Reminder!

- In your program, no GLOBAL DATA are allowed, and you must write all needed functions (no library functions are allowed – Except for **`cin`** and **`cout`** and their functions/manipulators).
- All user-created objects must be dynamic — creating through operator `new`.
- All dynamic allocations must be released/deleted properly — through operator `delete`!
- One `new` ←→ One proper `delete` — syntax errors or/and heavy penalties if this is not followed!
- Smart pointers are created differently and used differently!
- 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 CPP file.

1. Coding Assignment

Exercise #1 – Due Saturday, November 4, 2023 by 11:00 pm through Email Submission

(1) Write a C++ program to display the output described below.

(2) First, the program should display the following information:

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

CIS 25 – C++ Programming
Laney College
Your Name

Information --
  Assignment:               HW #3
  Implemented by:           Your Name
  Required Submission Date: 2023/11/04
  Actual Submission Date:   ____/__/__
```

You need to replace "**`Your Name`**" with your real name and provide the date for the "**`Actual Submission Date:   ____/__/__`**" (using proper and requried date format).

For examples, if your name is **`John Smith`** then `YourName` should be `JohnSmith` throughout all of your work/code as mentioned.

(3) 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 #2 Exercise #1
  Implemented by:           Your Name
  Required Submission Date: 2023/11/04
  Actual Submission Date:   ____/__/__

*****************************************************
*                   MENU – HW #3                    *
*  (1) Calling displayDigitInfoWithArrayYourName()  *
*  (2) Quit                                         *
*****************************************************
Enter an integer for option + ENTER: 6

Wrong Option!

*****************************************************
*                   MENU – HW #3                    *
*  (1) Calling displayDigitInfoWithArrayYourName() *
*  (2) Quit                                        *
****************************************************
Enter an integer for option + ENTER: 1

Calling displayDigitInfoWithArrayYourName() -

  While displayDigitInfoWithArrayYourName() is running –

    How many integer(s)? 5

    Enter integer #1: -29
    Enter integer #2: 129142571
    Enter integer #3: 52
    Enter integer #4: -25904
    Enter integer #5: -245766

    There is/are
      1 negative and odd integer(s)
      2 negative and even integer(s)
      1 positive and odd integer(s)
      1 positive and even integer(s)

    The digit(s) is/are found as follows,
      0 seen 1 time(s)
      1 seen 3 time(s)
      2 seen 6 time(s)
      4 seen 3 time(s)
      5 seen 4 time(s)
      6 seen 2 time(s)
      7 seen 2 time(s)
      9 seen 3 time(s)

    The most seen digit(s) is/are
      2 seen 6 time(s), and
        2 is found in
          -29
          129142571
          52
          -25904
          -245766

****************************************************
*                    MENU – HW #3                  *
*  (1) Calling displayDigitInfoWithArrayYourName() *
*  (2) Quit                                        *
****************************************************
Enter an integer for option + ENTER: 2

Calling displayDigitInfoWithArrayYourName() -

  While displayDigitInfoWithArrayYourName() is running –

    How many integer(s)? 5
    Enter integer #1: 129142571
    Enter integer #2: -2914258
    Enter integer #3: -29258
    Enter integer #4: 22581
    Enter integer #5: -1134311

    There is/are
      1 negative and odd integer(s)
      2 negative and even integer(s)
      2 positive and odd integer(s)
      0 positive and even integer(s)

    The digit(s) is/are found as follows,
      1 seen 9 time(s)
      2 seen 8 time(s)
      4 seen 3 time(s)
      5 seen 4 time(s)
      7 seen 1 time(s)
      8 seen 3 time(s)
      9 seen 3 time(s)

    The most seen digit(s) is/are
      1 seen 9 time(s), and
        1 is found in
          129142571
          -2914258
          22581
          -1134311

****************************************************
*                    MENU - HW #3                  *
*  (1) Calling displayDigitInfoWithArrayYourName() *
*  (2) Quit                                        *
****************************************************
Enter an integer for option + ENTER: 2

Calling displayDigitInfoWithArrayYourName() -

  While displayDigitInfoWithArrayYourName() is running -

    How many integer(s)? 4

    Enter integer #1: 1221
    Enter integer #2: -1221
    Enter integer #3: -2222
    Enter integer #4: 1111

    There is/are
      1 negative and odd integer(s)
      1 negative and even integer(s)
      2 positive and odd integer(s)
      0 positive and even integer(s)

    The digit(s) is/are found as follows,
      1 seen 8 time(s)
```

```
          2 seen 8 time(s)

        The most seen digit(s) is/are
          1 seen 8 time(s), and
            1 is found in
              1221
              -1221
              1111

          2 seen 8 time(s), and
            2 is found in
              1221
              -1221
              -2222

    ****************************************************
    *                    MENU - HW #3                  *
    *  (1) Calling displayDigitInfoWithArrayYourName() *
    *  (2) Quit                                        *
    ****************************************************
    Enter an integer for option + ENTER: 2

    Have fun!
```

At least, your program should have and use the following functions,

`displayCodingStatementYourName()`

`displayClassInfoYourName()`

`runMenuHw3YourName()`

`displayDigitInfoWithArrayYourName()`

where `YourName` must be replaced by your first name and your last name. For examples,

If your name is **`John Smith`** then `YourName` should be `JohnSmith` throughout all of your work/code as mentioned.

The sample run will have the options and values selected by the user. While testing, you must run your program to produce the required output.

(4) Save the work with

`cis25Fall2022YourNameHw3.cpp`

The above output should be copied and added to the end of the code in the PROGRAM_OUTPUT comment block. No manual manipulation to the output is allowed; you will get zero (0) for the whole work if any manual manipulation is found!

(5) Email the source code (your program) above using the SUBJECT LINE of

`cis25Fall2023YourNameHw3.cpp`

Again, THE Notes!

- **You are only allowed to use `cout` and `cin` from the `iostream` header.**
- **You are not allowed to use any other classes or syntax structures that are not introduced in class and class meetings — please confirm with the instructor if you have any doubt!**
- **You are not allowed to use `class`es and functions written by someone else.**
- **You must write all classes and functions yourself before using them in the required class work.**
- **All `class` and `type` names must have your first and last names appended.**
- **All local variables must be declared at the top of its function.**
- Except for the member data, 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;
  ```
- **All function names must have the initials of your Firstname and Lastname appended at the end.**
- **All filenames must have your complete Firstname and Lastname appended as required.**
- **The values of the elements from the created array, if any, must not be changed/modified at all during the analysis steps and process unless they are required to be updated or changed based on the specified tasks.**

Reminder!

- In your program, no GLOBAL DATA are allowed, and you must write all needed functions (no library functions are allowed – Except for **`cin`** and **`cout`** and their functions/manipulators).
- All user-created objects must be dynamic — creating through operator `new`.
- All dynamic allocations must be released/deleted properly — through operator `delete`!
- One `new` ←→ One proper `delete` — syntax errors or/and heavy penalties if this is not followed!
- Smart pointers are created differently and used differently!
- 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 coding convention CPP file.