ET580 PROJECT 1: DYNAMIC ARRAY CONTAINER CLASS

GENERAL INSTRUCTIONS

SUBMISSION REQUIREMENTS

1. Store all project files in a **Project_1** folder which includes the **documentation.txt** file.

   Test the final submission with the following commands to make sure it works:
   make; ./prog; make clean

2. Submissions are eligible for credit if the following is true:

   a. The project will compile and run as submitted. Block comment problem code.

   b. Issue tracking files will include a complete list of all task and bug resolution.

   c. Each student will have a significant commit history with clear messages.

   d. Each function in a .cpp file is commented to identify authors and editors.

   e. Each student will attend a team interview to discuss the project.

IMPLEMENTATION REQUIREMENTS

1. Submitted code is limited to the concepts and commands used in class.
2. Submitted algorithms will implement the provided pseudocode without deviation.
3. Objects will be passed by *reference*.
4. Data will be passed by *constant reference* unless the data is to be modified.
5. *Assertions* will be used to test and validate function parameter values.
6. *Cout/Cin* usage is limited to console interaction functions, such as *print* or *main*.

PHASE I

1. Read the **Group Instructions** document for implementation details.

2. Create the following files within a folder named **Project_1** in the group repository:

   driver_phase1.cpp
   Container.h
   Container.cpp
   Files.h
   Files.cpp
   data.csv
   Makefile

3. Add *preprocessor directives* to all header files and *include* statements as needed. Prepare the *Makefile*.

4. Store 5 or more comma delimited container values sorted from smallest to largest in the *data.csv* file as a single row of comma-delimited text.

5. Code the *class*, *data members* and *core member functions* for *Container.h/.cpp*.

6. Implement the *read* function in *Files.h/.cpp* to read values into the container.

7. Program *driver_phase1.cpp* to test **all** phase 1 functions.

PHASE II

1. Add this file to the program:

   driver_phase2.cpp

2. Update the *Makefile* to compile the program with the new driver file.

3. Implement the specified *extended member functions* for *Container.h/.cpp*.

4. Copy *driver_part1.cpp* content to *driver_part2.cpp*. Add code to *driver_part2.cpp* to test **all** phase 2 functions.

PHASE III

1. Add these files to the program:

   Algorithms.h
   Algorithms.cpp
   driver_phase3.cpp

2. Update the *Makefile* and include statements to compile with the new files.

3. Core the specified functions in *Algorithms.h/.cpp*.

4. Copy *driver_part2.cpp* content to *driver_part3.cpp*. Add code to *driver_part3.cpp* to test **all** phase 3 functions.