CSCE 240 – Exam Two

Due: 11:59pm on Friday, March 24. Late exam submissions will not be accepted.

This is an exam. As you work on these problems, you may use your textbook, class notes, and the recorded lectures. You may ask your instructor clarifying questions. You are not to discuss the problems with other students or seek help from other individuals. All work submitted must be your own. All code submitted will be examined for plagiarism and violations will be reported to the office of Student Conduct and Academic Integrity.

Test all of your code on a Linux lab computer. All source files submitted must compile and run on a Linux lab computer of the instructor's choice. Submissions that do not compile on the Linux workstation will receive no compilation or execution/correctness points.

Problem 1

Deliverables: *length.h* and *length.cc*

Purpose: Create a *Length* class that has a private data member for the length value (a double) and a private data member for length unit of measure (a string). You will include a constructor, accessor and mutator functions for the private data members, a public member function to convert between units of measure, and overloaded +, ==, <, and << operators. Read the comments in the attached *length.h* file for more details.

Specifications:

- The files for this problem are in the attached *problem1.zip*
- All of your code for this problem should be included in *length.h* and *length.cc*
- A Copyright comment with your name must be at the top of *length.h* and *length.cc*
- Ensure that your class will compile, link and run with the initial tests provided in order to earn compilation points and to be eligible for correctness points.
- Attach your completed *length.h* and *length.cc* files (do not zip the files) to the assignment in Blackboard.

Initial Testing:

testlengthparta.cc has been included with some minimal initial tests for the constructor, mutator functions, accessor functions, and convert function. To run these tests, ensure that *length.h*, *length.cc*, *testlengthparta.cc*, and *makefile* are in the same directory, and type ***make testlengthparta*** at the command prompt.

testlengthpartb.cc has been included with some minimal initial tests for the **+**, **==**, **<** and **<<** operators. To run these tests, ensure that *length.h*, *length.cc*, *testlengthpartb.cc*, *makefile*, and *correct-lengthpartb.cc* are in the same directory, and type **make testlengthpartb** at the command prompt.

You are encouraged to include more rigorous tests before submitting your class. This problem will be graded with modified versions of *testlengthparta.cc* and *testlengthpartb.cc*

Points:

style: 1 point

documentation: 1 point

clean compile / link with *testlengthparta.cc*: 0.5 point

clean compile / link with testlengthpartb.cc: 0.5 point

execution / correctness of constructor: 1 point

execution / correctness of accessor and mutator functions: 1 point

execution / correctness of *Convert* function: 1 point

execution / correctness of + operator: 1 point

execution / correctness of == operator: 1 point

execution / correctness of < operator: 1 point

execution / correctness of << operator: 1 point

Problem 2

Deliverable: *pointerproblemfunctions.cc*

Purpose: Implement the *Merge* function whose prototype is given in *pointerproblemfunctions.h*. Read the comments in *pointerproblemfunctions.h* for details about this function.

Specifications:

- The files for this problem are in the attached *problem2.zip*
- No changes are to be made in the attached *pointerproblemfunctions.h* file
- Implement the *Merge* functions in *pointerproblemfunctions.cc* and attach your revised *pointerproblemfunctions.cc* file to the assignment. This is the only file to be submitted for this problem.
- Points will be deducted if your function creates a memory leak or memory leaks.
- Your code should compile and link with the command
  g++ -std=c++17 -I . pointerproblemfunctions.cc pointerproblemdriver.cc
  to see if your *Merge* function correctly merges two *SortedDynamicArray*s
- Attach your completed *pointerproblemfunctions.cc* file only (do not zip the file) to the assignment in Blackboard.

Initial Testing:

"Randomly" generated *SortedDynamicArray*s are created in *pointerproblemdriver.cc* and the *Merge* function is called to merge the two structures. You are encouraged to create more rigorous tests. A revised version of *pointerproblemdriver.cc* will be used to test your function for correctness.

Points:

style: 0.5 point

documentation: 0.5 point

clean compile / link with *pointerproblemdriver.cc*: 1 point

execution / correctness of *Merge* function: 3 points