COMP 3512 Lab 4

Object Oriented Programming in C++

EDIT: Due 24 hours after your lab ends NO EXCEPTIONS

1 Instructions

Welcome to your fourth COMP 3512 lab. Today we will continue to apply what we have learned about writing good classes in C++ by implementing a simple Vector. I'd like you to set up your Visual Studio project on Github as a private repo, and invite me as a collaborator.

Vectors in an n-dimensional Euclidean space can be represented as coordinate vectors in a Cartesian coordinate system. That is, a vector can be represented with an ordered list of n real numbers (n-tuple). These numbers are the coordinates of the endpoint of the vector, with respect to a given Cartesian coordinate system, and are typically called the scalar components (or scalar projections) of the vector on the axes of the coordinate system.

We will implement a 3-tuple Vector which represents a location in 3-D space. This will give you some more practice defining C++ classes and overloading operators for them.

2 Requirements

Please complete the following:

1. Create a new private repository called COMP_3512_Lab4. Do this from inside Visual Studio:
   (a) Open the Team Explorer window and make sure you are on the Connection tab
   (b) Select Github >Create
   (c) For the name of the project, use COMP_3512_Lab4
   (d) Ensure Git ignore is set to VisualStudio
   (e) Ensure Private Repository is selected (very important)
   (f) Select Create. Don't touch anything while you wait.

2. This creates a new repository in Github and a local clone on your laptop. The local clone of the repository is empty, though. We need to create a Visual Studio project for it.

3. After you created the new repo from inside Visual Studio, a little yellow information window appeared in the Connect pane. It contains a hotlink called Create a new project or solution. Click the hotlink to open the New Project dialog.

4. Ensure Visual C++ is selected in the left pane, and Empty Application in the right name. For the name of the application, use Vector.

5. This will create a Visual Studio 2017 solution inside the new, empty repository. Let's make our first commit and push.

6. Open the Solution Explorer window and make sure you are in solution view, not folder view. If you are in folder view, click the icon to the right of the home icon.

7. Ensure you are using x64 architecture.

8. In the Home pane of the Team Explorer window, select Changes.

9. Enter a commit message ("Vector lab created" or something like that) and then choose Commit All and Push.

10. It's time to build our Vector! Your Vector must meet the following requirements:

    (a) you must commit and push your code after each function has been implemented.

    (b) you must include a header file called Vector.hpp and a source file called Vector.cpp. You must prove that your Vector works by testing it in a main method which is inside a second source file called Vectortester.cpp.

    (c) the Vector only stores three doubles called x, y, and z.

    (d) implement a default constructor which initializes all three values to 0.

    (e) implement a three-parameter constructor that accepts three doubles and assigns them to x, y, and z.

    (f) implement a copy constructor.

    (g) implement three accessors called get_X, get_Y, and get_Z which each returns the corresponding coordinate.

    (h) implement three mutators called set_X, set_Y, and set_Z which each sets the corresponding coordinate.

    (i) implement a function called clear which sets all values in the Vector to 0.

(j) implement a destructor called ~Vector.

(k) implement an overloaded insertion operator so we can print the Vector to std::cout or other streams.

(l) implement the unary increment and decrement operators. You will need a prefix and a postfix version of each. The operators should (respectively) increment or decrement each element's value value in the Vector.

(m) implement the assignment operator using the copy-and-swap algorithm introduced in lecture 9 so we can print the Vector in a printstream.

(n) implement some binary arithmetic operators. Please overload operator+=, operator+, operator -=, and operator-.

(o) implement two version of operator*. The first version should be a friend that accepts two Vector references and return the dot product which is a double. Recall that the dot product is the sum of the products of corresponding elements. The second version should be a member function that accepts a double and multiplies each element in the Vector by the double value.

11. You must test your code in the Vectortester.cpp file. Your code must prove that Vector class, its member functions, and any friend functions also work.

12. Finally, add me as a collaborator on Github. My Github account is chris-thompson. By adding me to your project as a collaborator, you give me permission to access it and view your solution directly. This is how I will mark your lab.

13. **BONUS: overload the operator[] which should accept an integer 0, 1, or 2, and return the value of the double stored in the x, y, or z respectively.**

3 Grading

Your lab will be marked out of 10:

- 4 points for correctness of your Vector implementation
- 2 points for thoroughness of testing in your main function
- 2 points for style (is your code indented, are variable names descriptive, is test output labeled and easy to understand, etc.)
- 2 points for evidence of multiple commits to Github during development.

This is an individual lab. Sharing code or submitting someone else's work is not allowed. By the end of this lab, you should be very comfortable implementing

C++ classes and some of their canonical functions.

Good luck, and have fun!