EECE/CS 115 Programming Assignment 5 (25 points)

Goals

1. More work with functions.
2. Learn how to use pointers.
3. Learn how to use classes.
4. Learn how to overload class functions.
5. Working with git.
6. Working with CMake.

Assignment

In this assignment you will be extending the 3D Vector class we started in lecture 5.

Requirements

1. You must turn in 2 files – Vector3D.h, Vector3D.cpp.
2. You will be extending the Vector3D class we started in class, so you can modify the existing files that are on github at https://github.com/scanavan/FRI_IASA_115.git.
3. Use the main.cpp file in the Assignment 5 zip file that is provided to test your program. If you have written the functions correctly, the output will look the same as figure 1.
4. Vector3D.h needs to go in the include directory, and Vector3D.cpp needs to go in the source directory.
5. You need to create the following class functions and constructors:
    a. Vector();
    b. Vector(float x, float y, float z);
    c. Vector(const Vector* vec);
    d. Vector* operator+(const Vector &rhs);
    e. Vector* operator-(const Vector &rhs);
    f. Vector* operator*(float scalar);
    g. Vector* operator/(float scalar);
    h. void operator+=(const Vector &rhs);
    i. void operator-=(const Vector &rhs);
    j. void operator*=(float scalar);
    k. void operator/=(float scalar);
    l. void operator=(const Vector& rhs);
    m. void operator-();
    n. bool operator==(const Vector &rhs);
    o. bool operator!=(const Vector &rhs);
    p. float Magnitude();//length of vector – **HOW DO YOU COMPUTE THIS?**
    q. void Normalize();//create unit vector (length is 1) – **HOW DO YOU DO THIS?**
    r. friend std::istream& operator>>(std::istream& is, Vector& rhs);
    s. friend std::ostream& operator<<(std::ostream& os, const Vector& vec);

6. Rules for creating your class:
    a. Your code must be modular (if you can use another class function inside another one, you must use it – you must use Magnitude and the overloaded /= inside Normalize)
    b. You can call the variables whatever names you want, however, they need to make sense.
    c. You must name the functions exactly as documented in step 7.
    d. Stream operators must be friends of the class.
    e. You can print the Vector data in the << operator anyway you like, but it must be neat and readable.
    f. Every function in this class can be written in <= 3 lines of code. If any of the functions are > 5 lines of code, you will lose points **(don't write spaghetti code that would normally be > 5 lines of code that you have condensed into < 5)**.
    g. You will need to use the constructors inside some of the functions. Make sure to use the correct one in each.
    h. Only create the functions listed in step 3, remove any extra that you have.
7. See Figure 1 for what your output should look like.

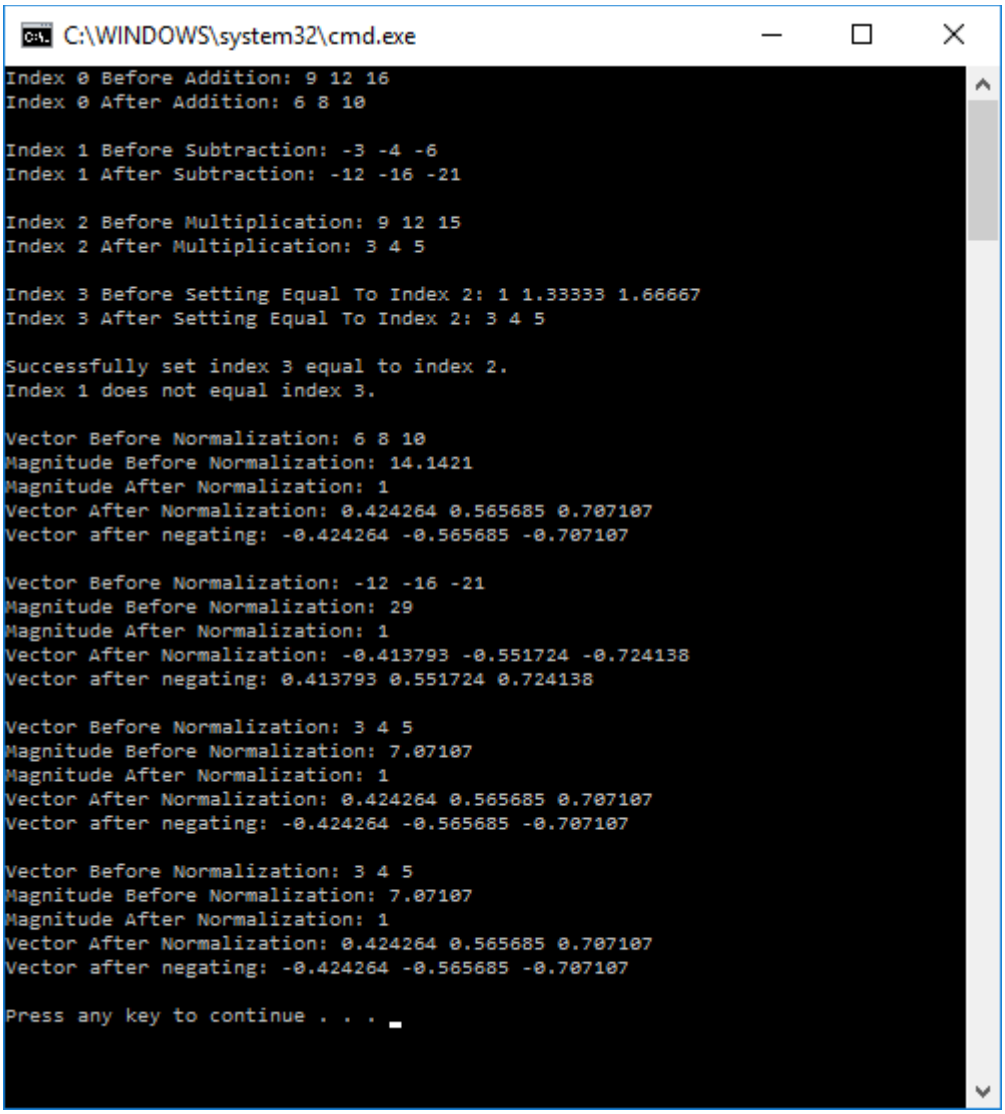

Figure 1. Correct output for assignment 5.

8. **Zip up the 2 files (Vector3D.h, Vector3D.cpp) – Assignment5_firstInitialLastName.zip (e.g. Assignment5_scanavan.zip).**

Grading (25 total points)

- 1 point: Correct submission format
- 15 points: Correct class functions
- 5 points: Correct use of pointers
- 4 points: Code modularity, length of functions (total lines for each function <= 5)