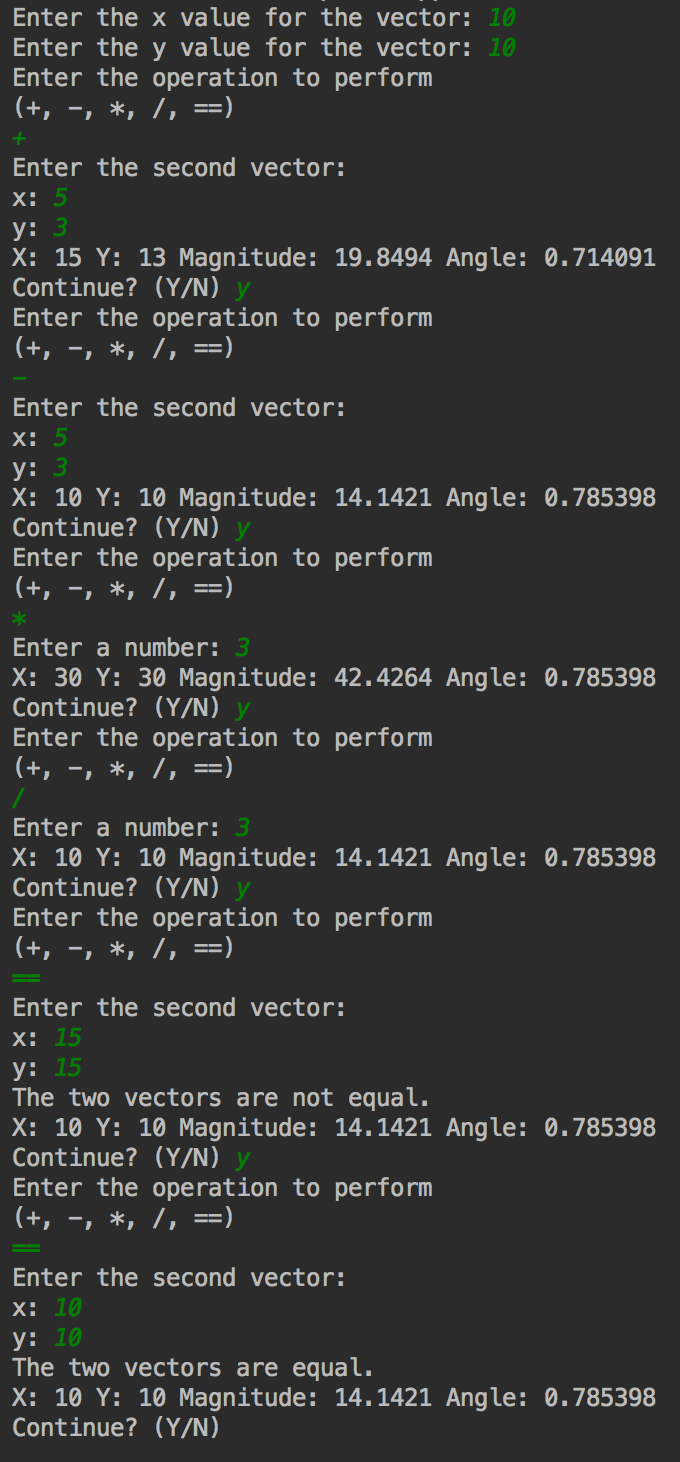
Kristian Snyder

Data Structures CS2021 Section 001

Prof. Charles Zimmer, Prof. Karthik Vadambacheri

Lab 03



Task 4: Result of all operations

**Central Concepts:**

Successful class design was the main objective to reach in this lab. There were many ways to perform operations on vectors and create a valid class. However, considering how to organize the member functions to manipulate attributes and finding redundancy reduced time taken and code complexity, while still having maintainable and extensible code.

Exposure to the standard library is also a significant portion of the project. There are many functions available that can help with number manipulation and calculations, such as abs() and sin() that make calculating things like the magnitude and angle far easier. The goal isn’t necessarily to learn how they work, but to learn that they exist and in what cases they would be useful.

Both goals are important to any professional career because careful planning time and use of all resources available will overwhelmingly result in a cleaner, more maintainable product. When so many examples of software are crippled by excessive technical debt, being aware of what will hurt maintainability and dramatically increase complexity is important.

**Class Design:**

My first thought on how to implement a vector was to determine what the minimum state of each instance would have to be. I prefer to leave as little complexity in the member variables as possible so that I can limit the amount of state checking necessary and create more complex data representations in the member functions. By doing this, less work goes into maintaining the state. Therefore, I only stored the x and y components of a vector. Everything else can be calculated. Redundancy in state can increase prevalence of bugs and adds effort to refactoring.

The first roadblock I ran into was in disambiguating the constructor taking x and y and the constructor taking a magnitude and angle. Since all parameters were type double, I added a parameter to the magnitude constructor called radians, a Boolean that would allow the constructor to be passed the angle in degrees or radians. It was largely an unnecessary feature, but was a clean disambiguation.

**Task 1 Rationalization:**

As stated above, the only member variables I created were x and y. These are the only two variables necessary to do vector calculations; everything else can be calculated from them. I could have stored the angle and magnitude, but I went for simplicity instead to make the class easier to read.

Getters and setters are just for easy access to change the vector’s x and y magnitude and direction.

The print method exists to meet the requirements for task 1. I included the magnitude since it can be useful to visualize a vector by its total length, not the length of its components.

**Changes for Task 2:**

When I was implementing the print functionality, I thought that having a dedicated method to get the magnitude for the vector would be a good idea, so I added the function getMagnitude to the header file and implemented it.

**Changes for Task 3:**

I didn’t have to make any changes besides visual reorganization and grouping of the class definitions and implementation. I placed all the operator overloads together and moved the constructors to the bottom of the declaration.