Project #: 3
Semester: Spring 2022_____
Name: Christopher LaFave

I. Requirements: Restate the problem specification and any detailed requirements in your own words.
Implement functions in a pre-made class to that they will work as intended.

II. Design: How did you attack the problem? What choices did you make in your design, and why? Show class diagrams for more complex designs.
I took each function one at a time. I would look at the function description, think through how to solve the problem, attempt an implementation, and then test and debug it until it worked.

III. Security Analysis: State the potential security vulnerabilities of your design. How could these vulnerabilities be exploited by an adversary? What would be the impact if the vulnerability was exploited?
Because this class uses pointers, I wouldn't be surprised if I accidentally implemented them in a way where more knowledgeable people than me could mess with memory through the code.

IV. Implementation: Outline any interesting implementation details in your solution.
I got stuck with the assignment operator and looked up how other people did it, and somebody used the swap method I implemented. It's really cool they way it just swaps the objects, and that the temporary object gets rid of the old data when it goes out of scope.

V. Testing: Explain how you tested your program, enumerating the tests if possible. Explain why your test set was sufficient to believe that the software is working properly, i.e., what were the range of errors for which you were testing.
I would primarily test them with the ZyLab tests, but if I couldn't figure out what was going on I would use the test program and set breakpoints and stuff like that until the ZyLab passed.

VI. Summary/Conclusion: Present your results. Did it work properly? Are there any limitations? NOTE: If it is an analysis-type project, this section may be significantly longer than for a simple implementation-type project.
It works properly! All of the tests passed. If there are limitations (which there probably are) I am not aware of them.

VII. Lessons Learned: List any lessons learned. For example, what might you have done differently if you were going to solve this problem again?
I learned a lot more about how constructors and destructors work, and how to mess with linked lists. I am now much more comfortable implementing definitions of functions in a class.