1. **Reflection**
   1. Testing Techniques
      1. What were the **software testing techniques** that you employed in this project? Describe their characteristics using specific details.
         1. The foundation for what type of testing I did was white box testing where I tested the code with an intimate knowledge about the code itself. The first technique I used was to test my statement coverage to ensure my executable statements were able to run. Then I moved on to branch coverage. This is where I tested each “branch” in my code, i.e. the if else statements.
      2. What are the **other software testing techniques** that you did not use for this project? Describe their characteristics using specific details.
         1. The one white box technique that I did not need to use was path coverage. This is because I did not have complex enough code to need to cover linear independent paths through my code.
   2. Mindset
      1. Assess the mindset that you adopted working on this project. In acting as a software tester, to what extent did you employ **caution**? Why was it important to appreciate the complexity and interrelationships of the code you were testing? Provide specific examples to illustrate your claims.
         1. The way that I employed caution with my code was by assuming it was going to not work until I proved it did work. It is important to go into coding with a mindset that it is going to be a complex web of logical links. While you may be able to understand what you are writing you may not understand its affects on the overall web. So it is important to work without assumption and with caution. I did this for my data maps. When I was originally testing my maps they did not work because I did not understand enough about them. I researched more on how they work and was able to fix the situation by creating my own maps in the tests.   
            A picture containing text

            Description automatically generated
      2. Assess the ways you tried to limit **bias** in your review of the code. On the software developer side, can you imagine that bias would be a concern if you were responsible for testing your own code? Provide specific examples to illustrate your claims.
         1. I do not think I actively tried to limit bias in my code. I passively limited bias in my code by acting with caution. Once again, I reference my data maps. I originally wanted to use arrays because I thought that they were going to be easier to work with. However, after further research I decided that data maps would be more efficient, even if they would be harder to work with.
      3. Finally, evaluate the importance of being **disciplined** in your commitment to quality as a software engineering professional. Why is it important not to cut corners when it comes to writing or testing code? How do you plan to avoid technical debt as a practitioner in the field? Provide specific examples to illustrate your claims.
         1. Being disciplined and committed to quality as a software engineer is very important. You can not cut corners or allow quality to suffer because you do not want to do the right thing, or you are low on time. It will reflect poorly on you, your team, and your company. For me personally I especially take it hard if it reflects poorly on me. To me my work reflects my ethic, so to do poorly is to fail. In the future I plan on preventing this by having diagnosed OCD and ADHD which give me a pathological need to have my code be as good as I can make it.