Building argument:

1. Never even heard of using GitHub to cheat. Didn’t even know you could find academic projects online on Github.I was very confused that mine somehow matched someone from 2019? I was in highschool in Spring 2019. I couldn't even have known anyones repository to copy from? I took OS in Fall 2021 so by the time I may have attempted to borrow a upperclassmen they would have more than likely have graduated from Clemson. I have no idea how I could have even gotten access to said repository.

Things to consider

Versions of projects. Three versions: two do not work one does work

1. These other non functional projects are in the zip file

How do I prove I did not cheat? By proving I worked on it?

- Both projects were made before the final. I have access to both to show I did not successfully submit it once and it worked. I iteratively kept adjusting and fixing issues as I found them. I can even list the issues to the best of my memory if required.
- I have email exchanges with Bennett himself attempting to understand the project as well as professor Sorber himself + technical questions. All of these comms are listed in a folder
- I have notes taken of me brainstorming ideas of how to get it work. I scanned all of my notes related to project-2
- Both were submitted into the auto grader with multiple failed attempts which shows I worked on it over a period of time. 9 total submissions. I have also a screenshot of each submission

Other Points of Discussion

1. I can also explain how the project works and could possibly do it again.
2. I have access to the cmake project I originally used. (I do not use make files)
3. Every project I have is made using cmake so I can show the original cmake file used for the project
4. I have other previous assignments (from any class) I have done for other courses that demonstrate my style (I'll provide any on request but I'll attach just one for now)
5. I can highlight chapters of the book + resources I read online to understand crucial functions in the program (Most of my notes contain small notes for important sections in the book I may use on the project)

Code Style Differences + Actual Code Differences (I don't know how I disprove the evident similarity other than finding obvious styling differences + logic differences)

1. Completely Different Style (I use camel case) and MOSTLY but not always I use braces around if + else + for loops + any other conditional logic branch. All my projects match this style. I attached my project-1 source code so you can see the similarity in style
2. None of the line numbers are the same
3. I didn't use comments because I actually forgot to write them in (Screenshot of email with bennett after submission to confirm this)
4. Spacing is completely different
5. I generally rarely write something as if(x == null) do this; which is another very evident style difference

Anti Cheat Questions + Things to Consider

1. Take into consideration the project description we were given a list of functions to use. That substantially reduces the possible uniqueness as each function requires specific arguments + information. Meaning said function must be used in a very particular way or else it will not properly work. Keep this in mind with each and every function. And also to my knowledge there's no way to do this without linked lists.
2. Note about Moss Similarity Directly from Stanford's website: (I'm assuming the professor used moss)
3. Moss is not a system for completely automatically detecting plagiarism. Plagiarism is a statement that someone copied code deliberately without attribution, and while Moss automatically detects program similarity, it has no way of knowing *why* codes are similar. It is still up to a human to go and look at the parts of the code that Moss highlights and make a decision about whether there is plagiarism or not. One way of thinking about what Moss provides is that it saves teachers and teaching staff a lot of time by pointing out the parts of programs that are worth a more detailed examination. But once someone has looked at those portions of the programs, it shouldn't matter whether the suspect code was first discovered by Moss or by a human; the case that there was plagiarism should stand on its own.In particular, it is a misuse of Moss to rely solely on the similarity scores. These scores are useful for judging the relative amount of matching between different pairs of programs and for more easily seeing which pairs of programs stick out with unusual amounts of matching. But the scores are certainly not proof of plagiarism. Someone must still look at the code