In our tests that were run, we found that while some of the class was functional, there were some problems and some weird interactions with the functions. We’ll detail some of the issues below.

**idForName**: There is something strange going on in the idForName function. When you call it you get a really strange ID that was never added. We aren’t entirely sure but it is either a problem in idForName or addUser. For example, when you add the name “Van” along with user ID 12345, you would expect the integer value returned from idForName(“Van”) to return 12345 but we get the seemingly random integer 57.

Testing that it wasn’t the name specifically, we tried another pair of values “Calvin” with 6789. The resulting integer was a really large value 4294967173. Seeing as the ID appears to work for other function calls (addPhoneNumbers and addGrade), we would assume that the problem is in the idForName function. Additionally, the problem might be that it is not dereferencing the value and taking the address (57 would be a strange address, however.)

**idForNameNegative**: This was a check to see what would happen if we input a negative number. Similar to the previous test, though, it looks like something is still wrong so it is returning random numbers back to us that aren’t close to what we put in.

**mapTest**: Map test tests the usage of the map data structure. The way we interpreted the implementation was that you should be able to have two users with the same name as long as they have different user ID’s. This doesn’t work out in reality, though, as you will get the most recently added user’s ID number.

**removeList**: There appears to be a problem within the function that only removes one user from the vector we put in. We tried inputting 4 names and 2 names and it returned 3 and 1 names in students respectively. This might be an issue with the function not iterating through the vector and removing names – it might be scraping off the first name from the vector and just removing that name it finds.

**fullRecord**: Interestingly, while the other methods accept literals, this method would not accept literals and had to have objects created. Even after creating them, there was still some cases where the program would not read the input to the problem. After one compilation, the program was working, and without adjusting anything, the next compilation rendered the program with errors.

Does not happen all the time, however, so this might be an error with the way objects are being parsed in.

Both of us sat together while we worked on the program, taking turns writing functions for the tests.