

**1.** The first and main obstacle which I had to overcome was honestly just figuring out how to format anything. Now that we needed two cpp files, a main cpp file, and 2 header files, it took me quite a while to figure out where everything went. For example, I still was a little confused on public vs private, as well as getters and setters. After rewatching the lecture recording, as well as looking up tutorials online, I was finally able to figure out the format of the project and the numerous files we needed to provide.

Coding the specifications and data validation rules in the “Setters'' section of my code was pretty easy for the most part, except for a single mistake I made, which I then got stuck on. In the setID function of BloodDonation, I finished my code to count the digits which were in the ID, by using a while loop that divides EmployeeID by 10. Once the while loop was completed, it stored the number of digits looped through in a separate integer which I called validID. The specs require you to set the mID to the EmployeeID if the ID had 6 digits, but because my while loop was changing the actual variable of the Employee ID, my mID at the end of my function was always being set to 0. To fix this problem, I defined a new integer called holderID, which would hold the original value of the EmployeeID, which I could set mID equal to later on. This was mainly the only notable problem I had with the BloodDonation Class.

With the Vacation Account class, there was really just one extremely simple mistake I made ( which I couldn’t figure out for the longest time). This came up in the constructor aspect of the Vacation cpp file, because I wrote EmployeeID = mID, rather than mID = EmployeeID. Rather than changing the value of mID, I was constantly changing the EmployeeID value to 0, so when testing, none of my asserts statements worked. Originally, I thought the problem was in the setter section of my code, which I spent a lot of time trying to fix, until I found the real solution in changing my constructor.

**2.**

BloodDonation bloodDoner1(123456, 29, 201.00);

BloodDonation bloodDoner2(654321, 32, 198.22);

BloodDonation invalidID(1000000, 28, 222.19);

BloodDonation invalidAge(123456, 99, 178.56);

BloodDonation invalidWeight(654321, 55, 23.22);

VacationAccount testVacation(654321);

assert(to\_string(bloodDoner1.getID()) == "123456");

**// testing getID (mID should change)**

assert(to\_string(bloodDoner1.getAge()) == "29");

**// testing getAge (mAge)**

assert(to\_string(bloodDoner1.getWeight()) == "201.000000");

**// testing getWeight (mWeight)**

assert(to\_string(bloodDoner2.getID()) == "654321");

**// testing getID**

assert(to\_string(bloodDoner2.getAge()) == "32");

**// testing getAge**

assert(to\_string(bloodDoner2.getWeight()) == "198.220000");

**// testing getWeight**

assert(to\_string(invalidID.getID()) == "-1")

**// testing an invalid ID test case**

assert(to\_string(invalidAge.getAge()) == "-1");

**// testing an invalid Age test case**

assert(to\_string(invalidWeight.getWeight()) == "-1.000000");

**// testing an invalid Weight test case**

assert(to\_string(testVacation.getBalance()) == "0.000000");

/**/ testing that the initial balance should be 0**

assert(to\_string(testVacation.getID())=="654321");

**// testing getID**

assert(testVacation.addVacationToAccount(bloodDoner2) == true);

**// testing a valid ID match with addVacationToAccount**

assert(testVacation.addVacationToAccount(bloodDoner1) == false);

**// testing an ID mismatch with addVacationToAccount**

assert(to\_string(testVacation.getBalance()) == "4.000000");

**// testing that +4 is added to mBalance**

assert(testVacation.addVacationToAccount(invalidID) == false);

**// testing an invalid ID with addVacationToAccount**

assert(testVacation.addVacationToAccount(invalidWeight) == false);

**// testing an invalid Weight with addVacationToAccount**

assert(testVacation.addVacationToAccount(invalidAge) == false);

**// testing an invalid Age with addVacationToAccount**

cout << "yuh get into it"; // should output this message if asserts work :)