Name: Marc Luzuriaga

UID: 205916585

Project Name: Project 6

**Notable Obstacles**

The biggest obstacle that I overcame in this project was determining how to create a function that checked whether an EmployeeID was valid or not. More specifically, I struggled with creating the validation of counting the integers. As a solution, I created a while loop to check the integer with a termination condition of greater than 9. If the ID was greater than 9, there would be a counter that would be incremented by 1.

Another obstacle that I overcame was determining how I could use functions from another header file. Although I used the #include header file, there seemed to be an error. I then learned that you had to create a function prototype at the top of the function again.

**Test Cases**

1.

BloodDonation doner12(910291, 21, 101);

VacationAccount account12(910291);

BloodDonation doner13(910290, 21, 101);

BloodDonation doner11(910290, 21, 101);

VacationAccount account11(910290);

assert(account11.getID() == 910290);

assert(account11.addVacationToAccount(doner11) == true);

assert(account11.getBalance() == 4);

assert(account11.addVacationToAccount(doner9) == false);

assert(account11.getBalance() == 4);

assert(account11.addVacationToAccount(doner12) == false);

assert(account11.getBalance() == 4);

assert(account11.addVacationToAccount(doner11) == true);

assert(account11.getBalance() == 8);

assert(account11.addVacationToAccount(doner11) == true);

assert(account11.getBalance() == 12);

assert(account11.addVacationToAccount(doner11) == true);

assert(account11.getBalance() == 16);

assert(account11.addVacationToAccount(doner13) == true);

assert(account11.getBalance() == 20);

The purpose of this test case is to check if the addVacationToAccount function was working properly and detecting whether the account had a weight, ID,or age of -1 and whether the names matched

2.

BloodDonation doner10(99999.99, 20, 100.9999999999999);

VacationAccount account10(99999.99);

assert(account10.addVacationToAccount(doner10) == false);

assert(doner10.getID() == -1);

assert(doner10.getAge() == -1);

assert(doner10.getWeight() == -1);

The purpose of this test case is to check an edge case on whether the function was using the correct type. This assert help me catch a mistake where I used an integer for weight instead of a double type.

3.

BloodDonation doner6(0, 66, 280.000000001);

VacationAccount account6(0);

assert(account6.addVacationToAccount(doner6) == false);

assert(doner6.getID() == -1);

assert(doner6.getAge() == -1);

assert(doner6.getWeight() == -1);

The purpose of this test case was to check whether or not my program caught an initialization of BloodDonation where the age was less than or equal to 65 or less than or equal to 280