Development Diary

By: Lavani Somesan

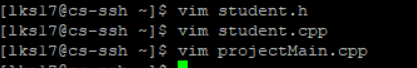
# Identifiable Classes

* Gradebook Class (holds weights and averages of grades, and can display report)
* Students Class (holds students names/records)
* Grades Class (holds students’ grades and can change them)
* **Functions of Student Class**
* Add Student Function
* Remove Student Function
* Get Student Name Function
* Constructor that defaults all students to noName
* Variables: string stuName, string defaultName
* **Functions of Grades Class**
* Add Homework Function
* Add Quiz Function
* Add Exam Function
* Constructor that defaults all grades to <ungraded>
* Variables: defaultGrades
* **Functions of Gradebook Class**
* Change Gradebook Weights
* Change Gradebook Name
* Get Averages
* Get Gradebook Name
* Get Gradebook Weights
* Constructor that defaults gradebook name to noName and weights to pending
* Variables: string defaultName, string defaultWeights, string defaultAverages

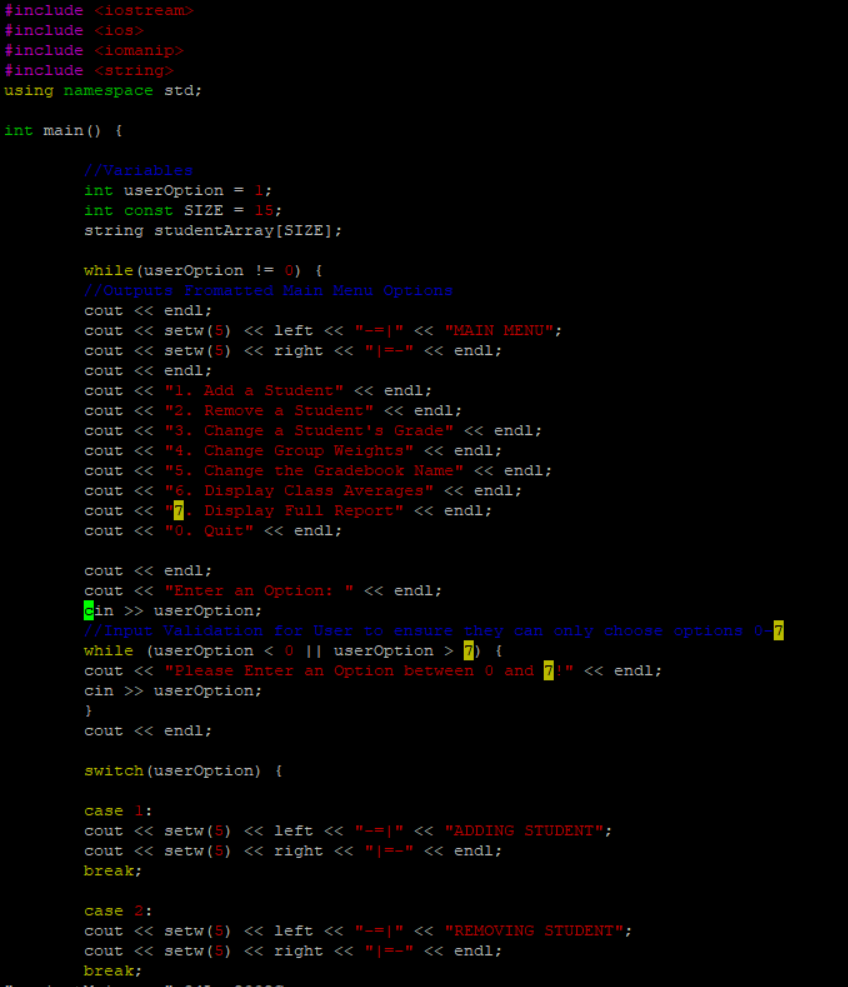
# Steps to Implement Code

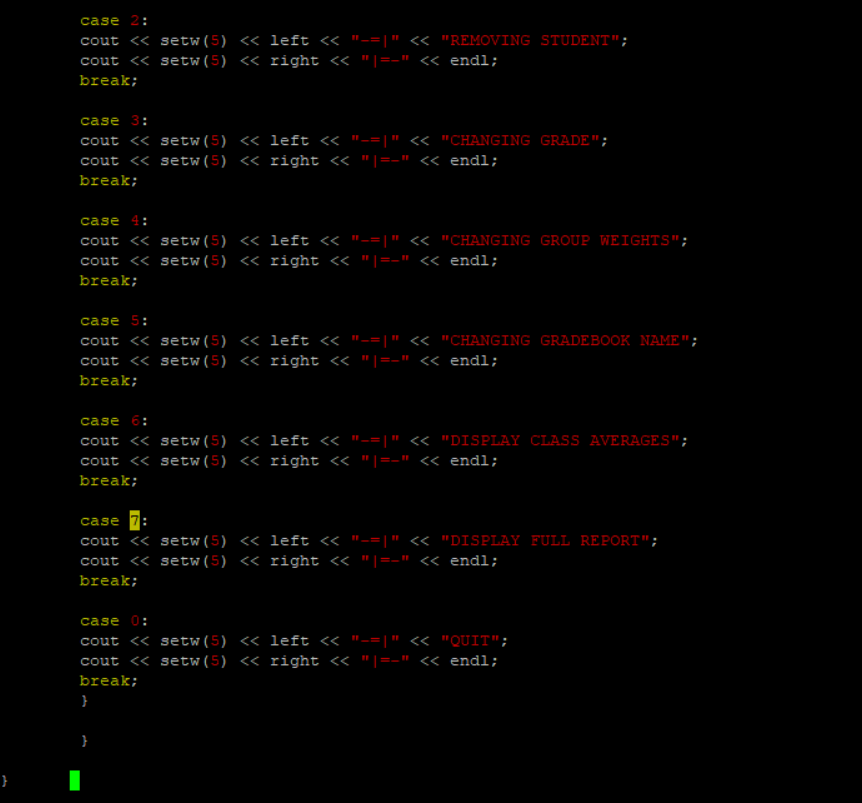
1. First create the student.h file for the student class declaration, then create the student.cpp file for the class definition, then create the main program file.
2. Write the skeleton code for the main program using switch statements for each menu option.
3. Compile the main program and make sure it runs correctly. Make sure output looks correct.
4. Write the student class and its definition. Then tie it to the main program and see if it runs correctly.
5. Next, create the Grade Class which will have a file for definition and declaration. This should hold 5 homework grades, 5 quiz grades, and 3 exam grades for each student that is initially set to ungraded. Make sure when student is removed the grades move up with student.
6. Compile the main program with the Student class and Grades class to make sure it runs correctly.
7. Next, create the Gradebook Class which will store the gradebook name initially set to noName, the Gradebook Weights which are initially set to “…”, and an Average function that calculates the averages for each assignment group and the final Average of the student (set initially to “…”).
8. Compile the main program with all the classes and make sure everything is working correctly.
9. Go back and add any last minute touch ups to the program.

# Step 1:



# Step 2:



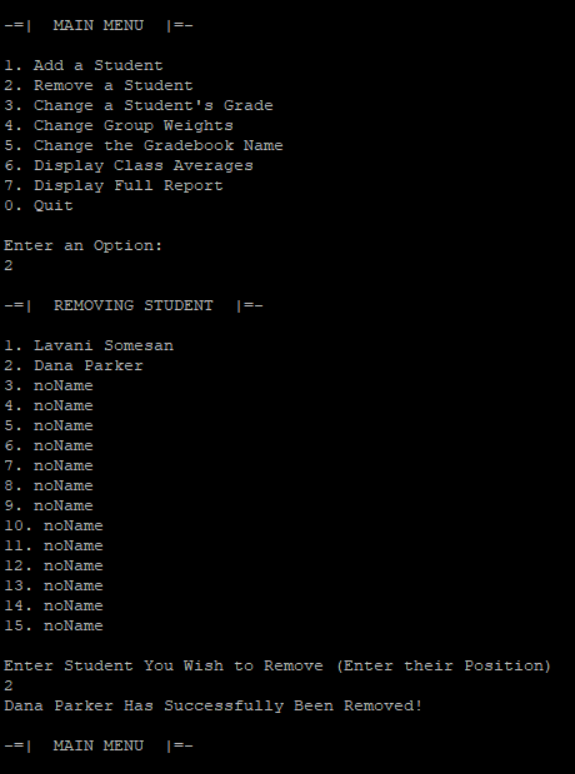


# Step 3:

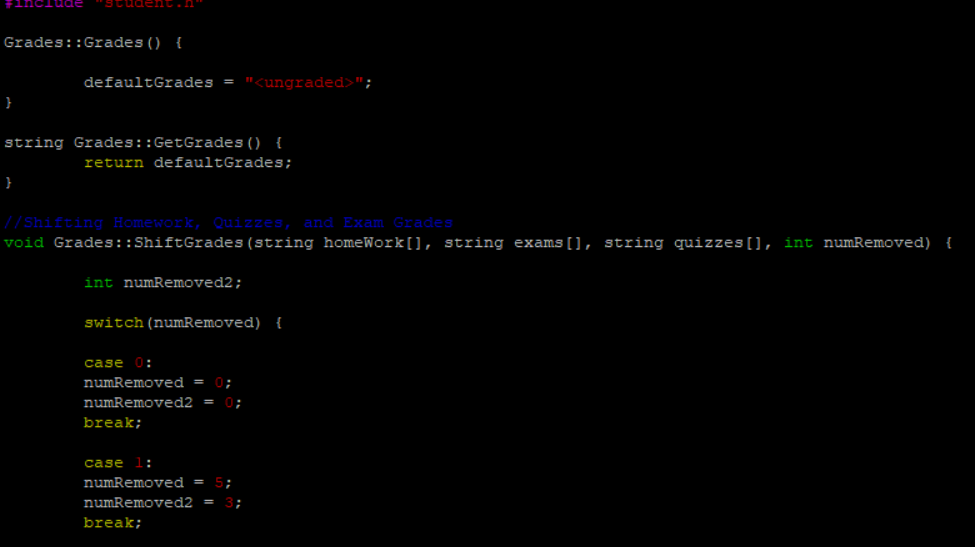
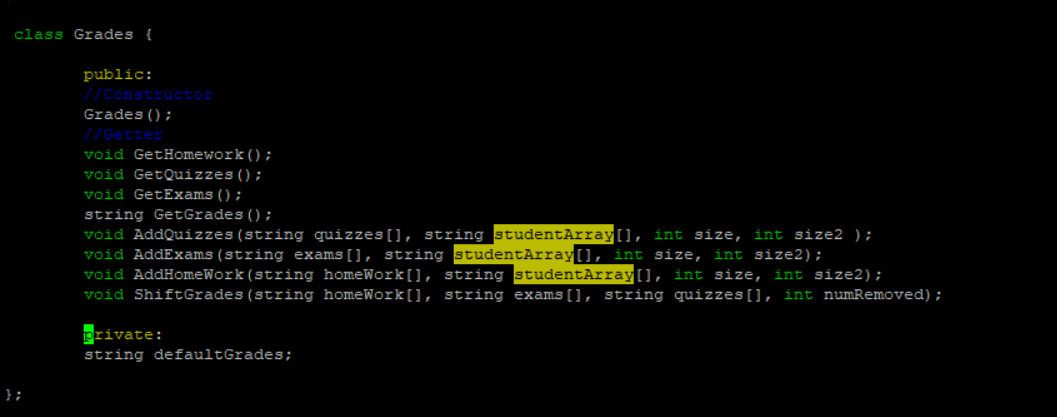
# 

# Step 4:

# 

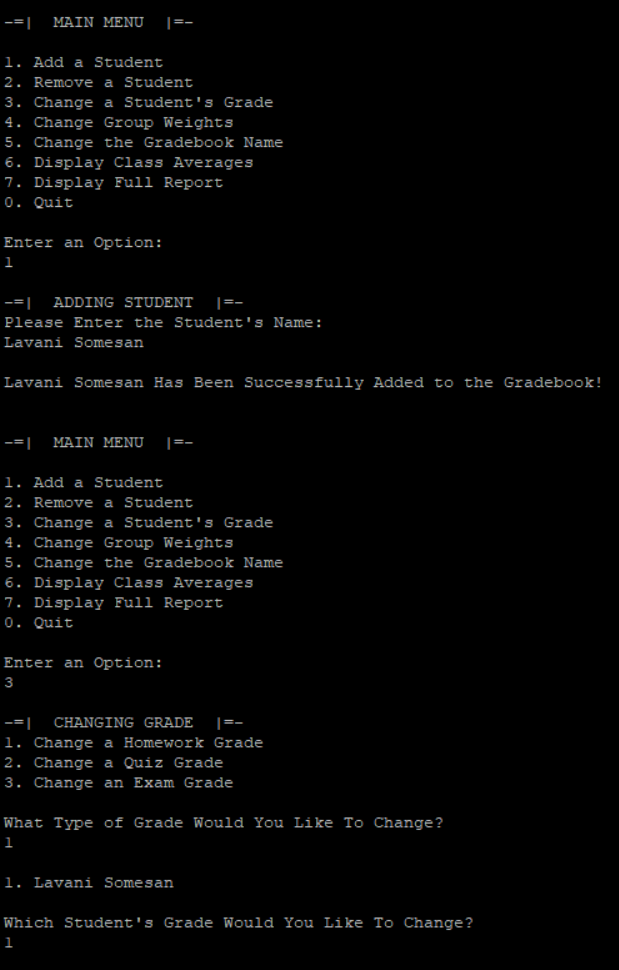


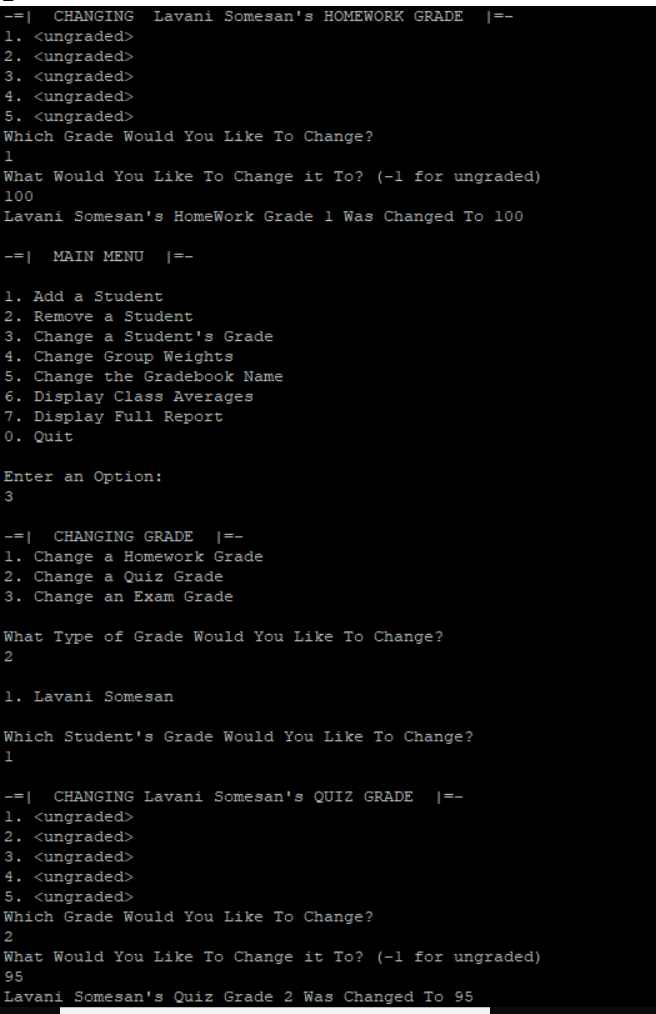
# Step 5:

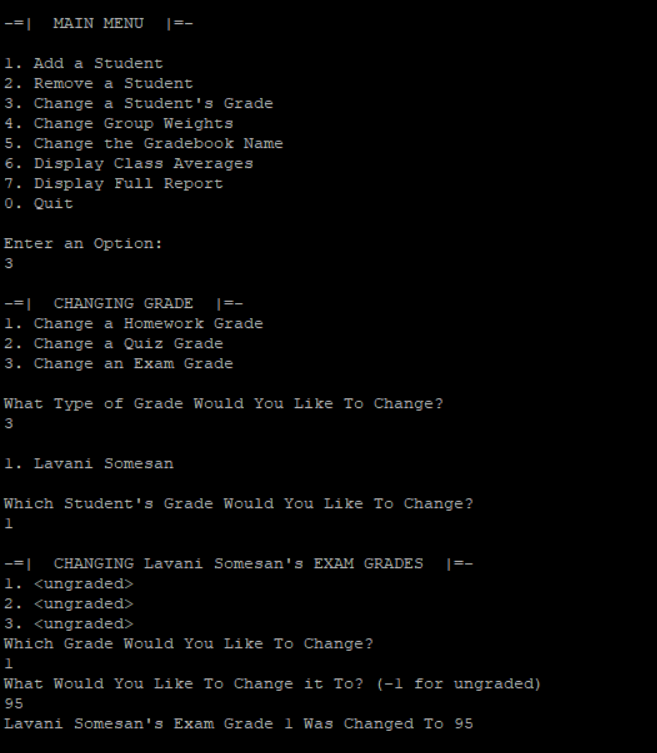


(grades.cpp was too long of a file to snapshot so I took a snap of the beginning of the file)

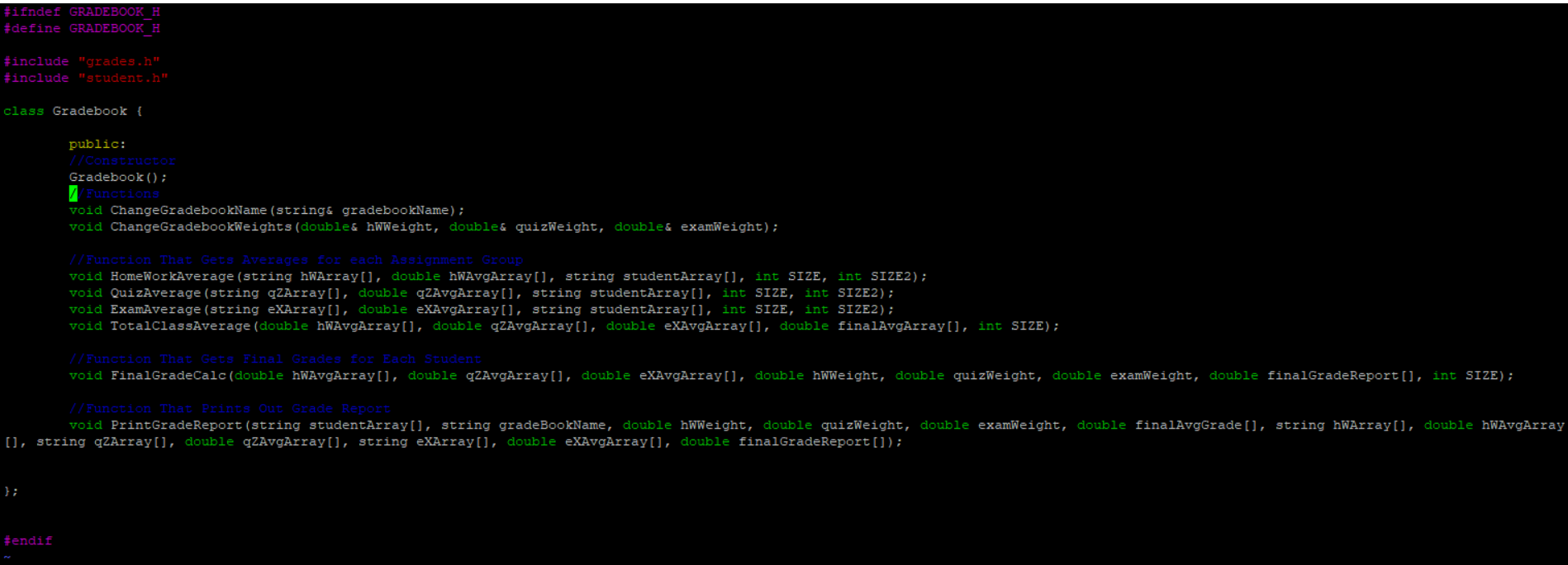
# Step 6:

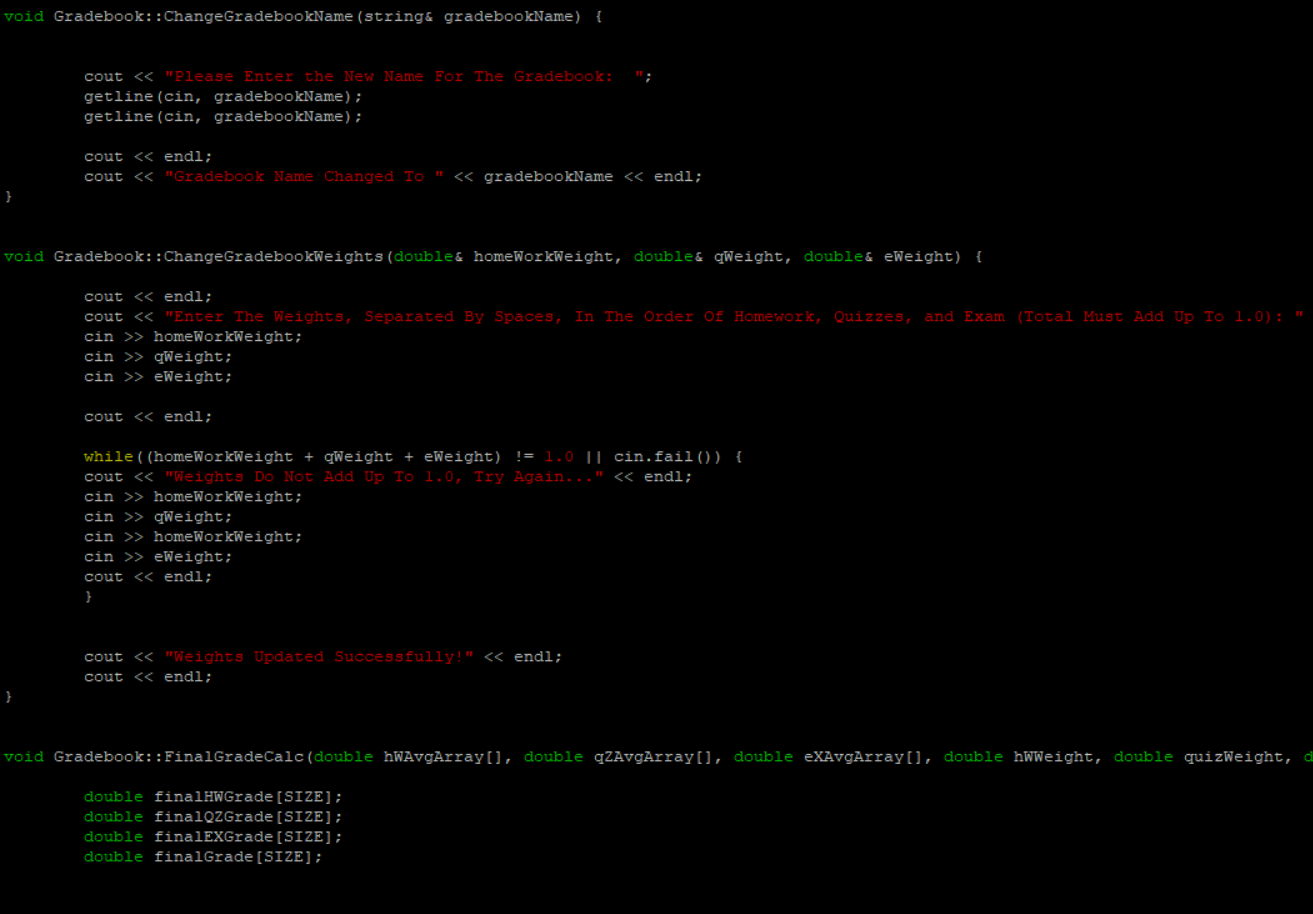






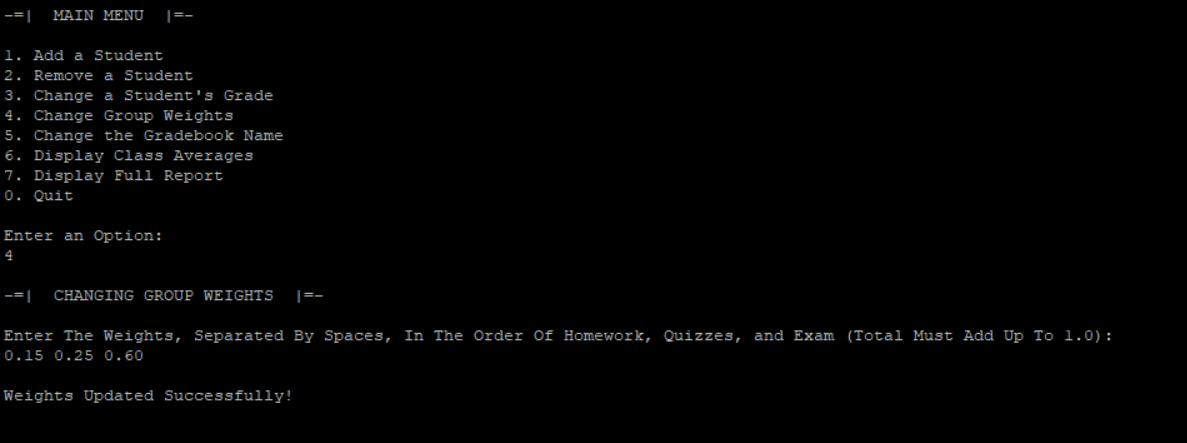
# Step 7:

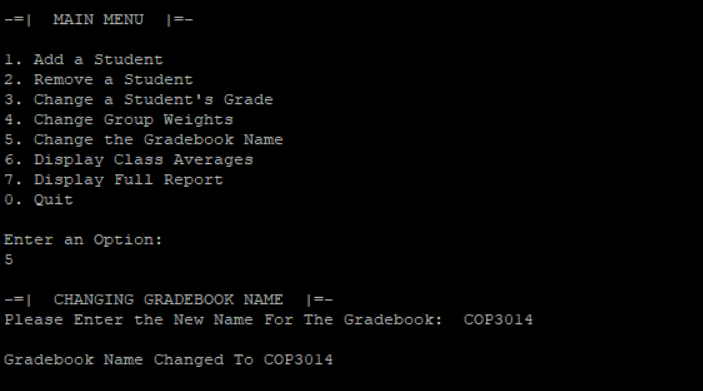


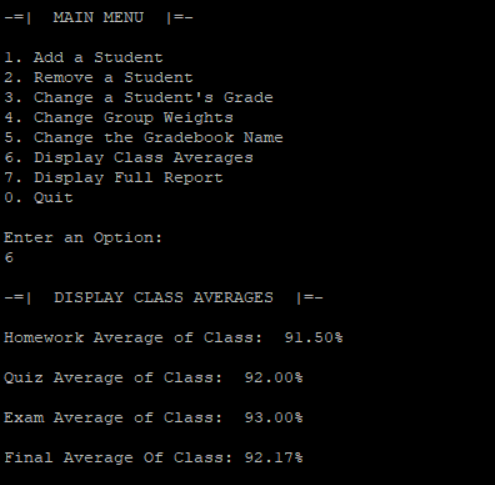


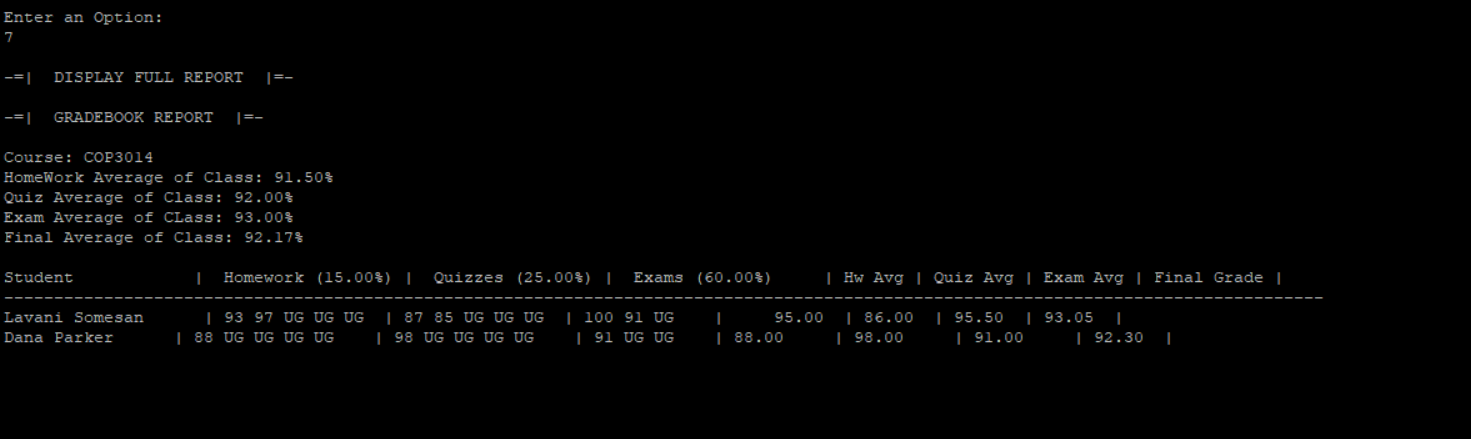
(gradebook.cpp is way too long to screenshot)

# Step 8:









# Changed During Process:

I kept all the classes that I identified with in the beginning. I changed some of the functions and what they do. For the Grades class I had to add a function to shift the grades up when a student is removed. For my Gradebook class I did add a lot of specific functions to it like FinalGradeAverage() and TotalClassAverage(). I also didn’t need that many variables and setters in this project since I was mainly passing arrays back and forth using pointers.