Weighted Average Calculation Project

In this project, I will be writing a C++ program to calculate a student’s final course average based on grades for their homework assignments, labs, and exams.

I will also cover the setup process and environments I use in order to write the program.

# Outline

As with any project, it is always important to start with an outline of what will need to be done in order to achieve the required goal. Below is my outline for this project, which this documentation will reflect:

* Planning the project
* Creating a GitHub repository for the project
* Including libraries
* Initializing and declaring variables
* Writing the program
  + Reading user input
  + Working with data from the user
  + Formatting output
* Debugging
  + Ensure project requirements are satisfied
  + Verify that output is accurate
  + Polishing the details
* Final thoughts

# Planning the project

Before I even start writing code, let’s first analyze the project requirements and determine what needs to be done to satisfy those requirements.

## Program description

*For this project, you will write a C++ program that calculates a student’s final course average based on the grades for their homework assignments, laboratory assignments, term paper, midterm exam, final exam and class participation.*

Right away, I know I will need to create several sets of variables, each for the various types of work assigned to students. I am not worried about the identifiers or the type of variables I will be using just yet.

## Assignments and exams

*There are four (4) homework assignments, each worth a maximum of 15 points. The homework average is 10% of the final grade for the course.*

*There are four (4) laboratory assignments, each worth a maximum of 25 points. The laboratory average is 20% of the final grade for the course.*

*The midterm and final exams are each worth a maximum of 100 points. The midterm and final exams are 25% of the final grade for the course.*

Now I have the specifics for the assignments. I know how many assignments there are, their point values, and the percentages of the overall class grade, respectively.

## Term paper

*The term paper is graded on a letter grade scale (A, B, C, D, & F). For the purpose of calculating the student’s average, letter grades are assigned the following numerical values: A=95, B=85, C=75, D=65, & F=55. If the user enters a letter grade other than A, B, C, D, or F a numerical grade of zero will be assigned to the paper. The term paper is 15% of the final grade for the course.*

Notice that the term paper is graded differently. Instead of a typical point system where an integer is read and used as the score for a particular assignment, a character is read from the console instead. That character is then mapped to an integer internally using the scale specified. This paper is worth a percentage of the overall class grade like all other assignments in this class.

## Participation

*Class participation must be a boolean value (0 {false}, 1 {true}). A true is successfully participating in the class. Class participation is 5% of the final grade for the course.*

This is just a simple bool variable. The value is true if a student attends class, and false if the student misses class.

## Input format

*The user will input their sample grade elements into the zyLab program input box. An entry for each of the grade elements: 4 homework assignments, 4 laboratories, midterm exam, final exam, term paper and class participation boolean (0 or 1).*

This paragraph detailing the format of the input is important as it specifies the order in which I need to read values in from the console. I will need to make sure I follow this order exactly, otherwise calculations made within the program at runtime will not be correct.

## Fulfilling project requirements

Various requirements are stated in the project which must be fulfilled throughout the program:

*Your code must utilize numeric constants for grade weighting factors (i.e., homework average of 10%, laboratory average of 20%, etc).*

*Calculate the homework and laboratory averages.*

*Calculate the final course average.*

*Output a neat summary of the student grades.*

*Use the calculated Final Average to assign a Letter Grade based on the following table:*

*A table with numbers and symbols

Description automatically generated*

As specified here, const variables must be used in the program. Not only will this satisfy the project requirements, but I also won’t be able to accidentally modify grade weighting factors, causing the calculations to be wrong.

At the end of the program, I must also use the iomanip library to make a nice display of all calculated values to print to the console.

# Creating a GitHub repository for the project

As with all projects, it is important to create a repository to be able to track file creations and modifications. This way, if something doesn’t work, it is possible to revert versions. This also serves as a way of working on the same project with multiple devices.

## Creating the repository

To start, let’s first create a new repository through GitHub’s web client. I have added the C++ .gitignore template so that machine-specific C++ files are ignored by git automatically. I also added a README.md file, because why not?

A screenshot of a computer

Description automatically generated

Now I can use git for everything else. Let’s switch over to the Windows Terminal and finish setting up the project.

## Cloning the repository

A good starting point for working with any repository is to clone it using git clone. However, before doing that, it would be a good idea to first change into a suitable directory instead of cloning a repo to the home directory. Here, I created a new folder called Schueckler-Projects, made specifically for projects like these:

A computer screen shot of a program

Description automatically generated

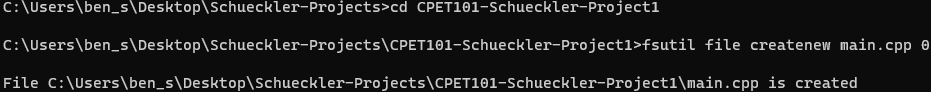
Next, I will change to the directory I just created, and run git clone to clone the repo to the newly created Schueckler-Projects directory:

A computer screen shot of a program

Description automatically generated

## Creating the main.cpp file

Currently, there’s no C++ source file in this directory. I will fix that by changing into the repo directory and use fsutil file createnew to make a new main.cpp file. I don’t want any contents written to this file just yet, so I’ll initialize the size of this file to 0:



In order for changes to apply to this file, I must also add it to git by calling git add:



## Opening the repo with [CLion](https://www.jetbrains.com/clion/)

All that is left to do before I start coding is to open CLion, an IDE made specifically for C++. This is what I like to use to write and test C++ programs as it is a fast, intelligent, and very well-built IDE.

Now I am finally ready to start writing the code!

# Including libraries

I will be making use of three libraries in this program.

As with most programs designed to be run from the command line, I will include iostream to be able to use cout and cin, as well as any other functions under this library. This allows the program to read and write to the console.

iomanip will be used to format the output into the requested format in order to fulfill the project’s code requirements.

The map library will be included as well in order to handle converting letter grades into integer grades and back. A map in C++ is similar to a dictionary in Python. I will talk more about the usage of this library later on.

Here are all of the libraries included at the top of the file, along with the rest of the empty C++ template:

A screenshot of a computer program

Description automatically generated

Notice how the include statements are grayed out. This is because CLion recognizes that I haven’t used these libraries yet and is suggesting I should remove unused libraries. This warning can be ignored as these libraries will soon have uses within the program, and CLion will dismiss those warnings soon anyway after I’ve used those libraries somewhere in the code.

# Initializing and declaring variables