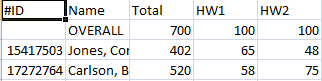
# CS 211 PA #5

At certain universities, professors often teach large classes in which an individual TA is responsible for grading a given section's programming assignment. Assuming that each TA fills in the Excel form correctly (a rather large assumption), the instructor must combine each section's grades into a master gradebook file that can be published to students. Less technical professors combine the gradebook by hand. However, this process can be cumbersome and is most certainly error prone! Your task for this assignment is to automate this process by writing a C++ application that automatically combines several CSV-based gradebooks into a single master gradebook.

## CSV File Format

Each CSV contains a single section's grades for a given category (e.g. homework). All files have a similar format. The top row contains headers and the 2nd row contains the maximum grade for each assignment. Similarly, the first column is always the student's ID number, the 2nd the student's name, and the 3rd is their overall score for the grading category. Rows after the third row contain marks for a given task. Here's an example:



### Obtaining a list of CSV files

Your program should prompt the user for a list of CSV files to parse. Each file will be in the format "<category>\_<number>.csv". For example: "homework\_1.csv", "homework\_2.csv", and "exams\_2.csv" are examples of potential file names. **Note that <category> can be any string**. Listed previously are strings that make sense, but your homework should be able to handle something like "sdfsrwe\_1.csv" just fine! Also, be sure to use <category> as the separator for the summary CSV file (discussed later).

## Expected Output

Your program should generate two CSV files: one summary file with overall grades and one details file that contains the grading breakdown for all sections.

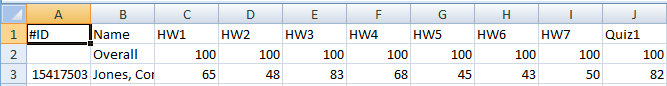
### Summary CSV File

The summary CSV file should contain all students, an overall class grade (in percent), and a points breakdown for each grading category. Here's an example screenshot of a summary file whose individual CSV files had a "Homework" and "Quizzes" <category> as inputs:



### Details File

The details file should essentially aggregate all of a student's records into a single file. In other words, it should include a detailed breakdown of every student's marks in the individual CSV files. Here's an example:



## Header Comment, and Formatting

1. Be sure to modify the file header comment at the top of your script to indicate your name, student ID, completion time, and the names of any individuals that you collaborated with on the assignment.
2. Remember to follow the basic coding style guide. A basic list of rules is included with this document.

## Deliverables

Your final project and reflection must be as a ZIP file through Canvas no later than midnight on Monday, December 5, 2016. A partial check-in of your progress will occur during lab on November 14. Remember that your submission must either contain a CodeBlocks or Visual Studio project file!

# Reflection Essay

In addition to the programming tasks listed above, your submission must include an essay that reflects on your experiences with this homework. This essay must be at least 350 words long. Note that the focus of this paper should be on your reflection, ***not*** on structure (e.g. introductory paragraph, conclusion, etc.). The essay is graded on content (i.e. it shows deep though) rather than syntax (e.g. spelling) and structure. Below are some prompts that can be used to get you thinking. Feel free to use these or to make up your own.

* Describe a particular struggle that you overcame when working on this programming assignment.
* Conversely, describe an issue with your assignment that you were unable to resolve.
* Provide advice to a future student on how he or she might succeed on this assignment.
* Describe the most fun aspect of the assignment.
* Describe the most challenging aspect of the assignment.
* Describe the most difficult aspect of the assignment to understand.
* Provide any suggestions for improving the assignment in the future.

## Grading Criteria

Your assignment will be judged by the following criteria:

* [25] November 14 Partial checkin. In order to receive full credit, you must demonstrate a program that:
  + Obtains a list of files from the user
  + Reads the list of files into one or more data structures (at minimum a vector, ideally a hashtable)
  + Contains one or more UML diagrams of the classes that you use or intend to use in your project.
* [5] Your final project runs and compiles
* [50] Your final project passes my test cases
* [10] Your reflection essay satisfies the requirements as specified earlier in this document.
* [15] Your project contains good structure. In other words, your program intelligently uses classes when appropriate and generally conforms to good OOP design (i.e. everything isn't slapped into main).