CS102: Introduction to Computer Science
Summer 2014
Program #4

You are going to expand the program discussed in class that reads student records from a specified text file, computes final averages, sorts the students, and outputs the calculated averages to another specified text file. You will add the following functionality:

1. You will add a field to the "student" structure; this will be a "char" that records the student's letter grade ('A', 'B', 'C', 'D', or 'F'). **10 Correctness points**

2. The compute_averages function will be expanded as follows:
    a. The user will be asked to enter a decimal fraction between 0 and 1 indicating the percentage of the final average determined by the test score (the last value in each row of the input file). The rest of the grade is determined by the homework scores, with each homework score weighted equally. **10 Correctness points**
    b. After the final average is computed, the appropriate letter grade will also be recorded in the student's record. The cutoffs for A, B, C, and D are 90.0, 80.0, 70.0, and 60.0 respectively; any final average lower than 60 gets an F. **10 Correctness points**

3. The sort_students function will ask the user if student records should be sorted according to names or final averages. If the latter, the records should be sorted from the highest average to the lowest average. (Alternatively, if you find it simpler, your program can include two separate sort functions for the two cases. In this case, you can prompt the user for instructions on how to sort in "main" and then call the appropriate function.) **10 Correctness points**

4. The display_students function will be expanded as follows:
    a. Each student's letter grade will be written to the output file after their final average (separated by a space). **10 Correctness points**
    b. At the end of the file, messages will indicate counts of each letter grade. A blank line should be written before these messages. The counts should be computed as the file is written. (You can do this elegantly with a small array of integers to store counts and very few lines of code.) **10 Correctness points**

5. Your input_students function will not assume that there are four homeworks per student. The first line of the input file will include two integers separated by a space. The first integer will still indicate the number of student records that follow. The second integer will indicate the number of homeworks per student. (Note: I consider this conceptually to be the most difficult feature to add, although I only had to add or change about ten lines of code to make this work. I will discuss hints in class. If you can't get this part to work, just leave it out, and you can still achieve up to a 90 on the assignment. I'll have a version of my test file with the second integer left out, and I should be able to change the constant NUM_HW in your code to test your program with the test file.) **10 Correctness points**

I will provide a single sample input file and two sample output files created using two different sets of input from the user, as will be explained in class. Your program should output EXACTLY the same thing as mine, assuming that I use the same input file and I enter the same input. Messages in the output file should be phrased the same way as mine, the same formatting should be applied, etc.

CS102: Introduction to Computer Science
Summer 2014
Program #4

Your homework will be graded out of 100 points with the following breakdown:

- **Correctness**: You should follow all instructions exactly as stated above. I will compute your total Correctness score according to the breakdown indicated on the first page of this document. This is an individual assignment (i.e., you should not collaborate with anyone else). **70 points**.
- **Elegance and Efficiency**: You should use the concepts we have learned in class to write your code in a simple, elegant manner. Avoid unnecessary processing. Do not use sets of separate variables when you can use a simple array. Avoid repeated code. **15 points**.
- **Format**: Your program should use proper indentation and other spacing which makes the code readable and easy to understand. **10 points**.
- **Comments**: Edit the comment at the top of the program explaining what the updated program does, and also include your name. Edit comments above each function, if appropriate. Add or edit comments within functions as appropriate. **5 points**.

Submitting assignments: Email me your code (to **CarlSable.Cooper@gmail.com**) as an attachment. Do NOT attach the executable (only send the source file). Please state in your e-mail which environment you used to develop the code (e.g., Cygwin, Quincy, Ubuntu, etc.). This program is due the night of Thursday, August 7, before midnight.