CS 288 2018S Section 006
Homework 02

Due: At the beginning of class on Wednesday February 14th.

Using an IDE or Text Editor of your choice, create and save a C file that is named, if your name is Harry Houdini, for example, *HW2_HarryHoudini.c* and begins with a block comment containing your name, class and section, and number of the homework assignment. Example:

```
/*
Harry Houdini
CS 288 2018S Section 006
HW 02
*/
```

A college professor has a very simple method to track grades. The method is as follows:

- When an exam is given, the answer sheets are collected in no particular order.
- As time allows, the professor grades the answer sheets, in the order they were collected, and enters the grades in a CSV file keyed by the Student ID.
- Three exams (2 mid-terms and a final) are given per semester.
- At the end of the semester, the professor prints out the three files, manually searches for the grades for each student (one grade per printout), and calculates the final grade to be the average of all three grades rounded to the nearest integer (rounding up if the fractional part is exactly 0.5).
- As the final grades are calculated, the professor enters the results in yet another CSV file to share with the Office of the Registrar.

Besides causing eye strain, this method is time consuming, and error prone. In addition, the registrar's office prefers to have the final grades sorted by Student ID, which the professor is not doing.

Your assignment, if you are willing to accept it, is to write the "Amazing Final Grade Calculator". The program should:

1. Accept the names of four CSV files through command-line arguments. The first three files contain "Student ID, Grade" records, and the fourth is the name of the CSV output file to receive the final grades.

2. Read the input files one by one. Every time a "Student ID, Grade" pair is read, create a node, and insert it into a singly linked list via the SLL_insert() function. The SLL_insert() function inserts the nodes into the list in ascending order based on the Student ID. The function implementation serves as proof of your newly found C Pointers wizardry.

   By the time the program finishes reading the input files, the length of the list should be triple the number of students because each student will have three consecutive nodes. The program must be able to support an outrageous number of students (hundreds of thousands).

3. The final grades can now be easily calculated by averaging three consecutive nodes at a time. Specifically, by popping three nodes from the linked list, calculating their average, and writing the output in "Student ID, Final Grade" CSV format to the output file.

If you do everything correctly as described above, the output file will be sorted by Student ID. The registrar's office staff are ecstatic, and they can't wait to put their hands on the sorted final grades. Don't disappoint them!

Sample Session, Input, and Output Files

Command Line

```
$ amazingCalculator midterm01.csv midterm02.csv finalexam.csv finalgrade.csv
```

Sample Input Files:

midterm01.csv	midterm02.csv	finalexam.csv
10012,85	10728,25	10728,80
11981,100	10012,75	11981,70
10728,65	11981,95	10012,65

Sample Output File:

finalgrade.csv
10012,75
10728,57
11981,88