CS 260
Spring 2015
Programming Assignment 2

- Write a Python3 program project2.py that operates as described below.
- Your program should accept three command line arguments. The first two arguments are names of input files, and the third argument is the name of an output file.
- All three files will be stored as comma-separated values (csv files format). However, your program must not import Python's csv module. The purpose of this programming assignment is to implement the necessary algorithms and data structures yourself, not just to call predefined functions from a library.
- You can open csv files with an ordinary text file editor or by using Microsoft Excel. Both views will be illustrated in the examples below.
- The first input file contains a database arranged as a 2-dimensionsal table. The first line or row contains column headings. Each remaining line or row contains a data record that consists of plain-text fields separated by commas. You may assume that all rows contain the same number of fields. However, do not assume any maximum number of rows or columns in the database.
- Example: Here are two views of a first input file input0.csv:

```
first_nm,last_nm,gender,cwid,cred_hrs,qual_pts,gpa
John,Roe,M,44444444,40,150,3.75
Jane,Roe,F,66666666,100,260,2.6
John,Doe,M,22222222,50,140,2.8
Jane,Doe,F,88888888,80,280,3.5
Penny,Lowe,F,55555555,40,140,3.5
Lenny,Lowe,M,11111111,100,280,2.8
Denny,Lowe,M,99999999,80,260,3.25
Benny,Lowe,M,77777777,120,90,0.75
Jenny,Lowe,F,33333333,50,90,1.8
Zoe,Coe,F,00000000,50,130,2.6
```

first_nm	last_nm	gender	cwid	cred_hrs	qual_pts	gpa
John	Roe	M	44444444	40	150	3.75
Jane	Roe	F	66666666	100	260	2.6
John	Doe	M	22222222	50	140	2.8
Jane	Doe	F	88888888	80	280	3.5
Penny	Lowe	F	55555555	40	140	3.5
Lenny	Lowe	M	11111111	100	280	2.8
Denny	Lowe	M	99999999	80	260	3.25
Benny	Lowe	M	77777777	120	90	0.75
Jenny	Lowe	F	33333333	50	90	1.8
Zoe	Coe	F	0	50	130	2.6

- The second input file specifies how your program should sort the data from the first input file. Each row of the second file has three fields, as follows. The first field is a column name from the first input file, the second field specifies the direction (ascend or descend), and the third field specifies the data type (string or int or float). The first row denotes the primary key for sorting, the second row (if it exists) denotes the secondary key, etc.
- Example: Here are two views of a second input file input0-7.csv:

```
gender,ascend,string
gpa,descend,float
last_nm,ascend,string
```

gender	ascend	string
gpa	descend	float
last_nm	ascend	string

- The output file should contain the same data as the first input file, with the rows rearranged as specified in the second input file. However, the first line of the output file should still contain the column headings.
- Example: Next suppose we run this command:
  python3 project2.py input0.csv input0-7.csv output0-7.csv
  Finally here are two views of the expected output file output0-7.csv:

```
first_nm,last_nm,gender,cwid,cred_hrs,qual_pts,gpa
Jane,Doe,F,88888888,80,280,3.5
Penny,Lowe,F,55555555,40,140,3.5
Zoe,Coe,F,00000000,50,130,2.6
Jane,Roe,F,66666666,100,260,2.6
Jenny,Lowe,F,33333333,50,90,1.8
John,Roe,M,44444444,40,150,3.75
Denny,Lowe,M,99999999,80,260,3.25
John,Doe,M,22222222,50,140,2.8
Lenny,Lowe,M,11111111,100,280,2.8
Benny,Lowe,M,77777777,120,90,0.75
```

first_nm	last_nm	gender	cwid	cred_hrs	qual_pts	gpa
Jane	Doe	F	88888888	80	280	3.5
Penny	Lowe	F	55555555	40	140	3.5
Zoe	Coe	F	0	50	130	2.6
Jane	Roe	F	66666666	100	260	2.6
Jenny	Lowe	F	33333333	50	90	1.8
John	Roe	M	44444444	40	150	3.75
Denny	Lowe	M	99999999	80	260	3.25
John	Doe	M	22222222	50	140	2.8
Lenny	Lowe	M	11111111	100	280	2.8
Benny	Lowe	M	77777777	120	90	0.75

- Because the size of the database is not known in advance, your program should use an efficient sorting algorithm. Otherwise your program might be too slow for some larger data sizes.

The following rules apply to all projects in this course:

- We now provide a virtual machine so you can test your program in the same environment where it will be graded. This server is "`cs260.ua.edu`". Regardless where you primarily develop your program, you must use this server to test your projects before submitting. Login using your Bama userid and password. You might first need to install the SSH secure shell and file transfer clients, for example at http://helpdesk.ua.edu/~recovery/ssh.htm. Also, if you wish to access the server from off-campus, you'll need to install a VPN client, which is available at https://mybama.ua.edu on the Tech tab.
- During grading, your program will be tested on the `cs260.ua.edu` server. Example scripts and input/output files will be provided on this server in the /projects directory.
- We provide a common platform that all students must use to test their programs before submitting, and we will use this same platform to evaluate each submitted program. Your program will NOT be evaluated based on how it performs in any other environment, because we will not attempt to replicate different environments used by each individual student.
- In previous semesters, some students did not follow the instructions and/or did not adequately test their programs before submitting, so they lost points for errors that they could have easily detected and corrected. This semester, it is required to test the final version of your program on the server before you submit it. All the scripts and input/output files are provided on the server in the /projects directory. When you run the script, it will produce a file "checksums.txt" that you must submit along with your program. When grading your project, we will use the contents of this file to verify that you actually did test the final version of your program on the server.
- You are permitted to verbally discuss ideas with classmates or other people, but sharing any code is strictly forbidden. **Do not look at your classmates' or anybody else's code, and do not show anybody your code!** However, you are encouraged to verbally discuss the projects with your classmates, provided nobody involved is looking at any code during the discussion.
- You must write all your own code for each project. Do not take code from any other source. If you happen to find a similar program online or in a book, you are NOT permitted to copy portions of that program into your program. The purpose of the projects is to learn how to write programs for data structures and algorithms, not to practice googling for solutions.
- I**f your code is too similar to another student's code or to anybody else's code, you will receive an invitation to discuss the situation in the Dean's office.** We use the MOSS system which detects plagiarism based on programs that are too similar. The penalty for plagiarism is typically a score of 0 on the project or a grade of F in the course.
- You should strive to make each program as efficient as possible. During testing, we will allow each program to run for a reasonable amount of time, but if your program takes much longer to run than we think it should, we will eventually decide your program is too inefficient (or possibly in an infinite loop), and in this case we may terminate the execution of your program before it ends.

- Place all your program’s source files and the file “checksums.txt” (generated by the script) into a single zip file, give it a name of the form LastName_FirstName_ProjectNumber.zip (example: Doe_John_1.zip), and upload this zip file using Blackboard. Make sure your zip file includes the latest version of all your source files. Your zip file should contain the file “chesksums.txt” and all the source files needed to run your program, and these files should all be located at the top level of your zip file. The zip file should not contain any other subfolders. Do not submit projects via email.
- Each assignment will be due by 11:59pm on the due date specified on Blackboard. Please aim to submit your project earlier than the due date in case you encounter unforeseen circumstances at the last minute. There will be a substantial deduction for late submissions. Points will also be deducted if you do not follow all the instructions exactly, because this can make it difficult or impossible to grade your project.
- After the projects are graded, your score will be uploaded onto Blackboard, along with a grading sheet that summarizes how your score was calculated. Also, any additional input/output files used when grading the projects will be placed in the /projects directory on the `cs260.ua.edu` server.
- To completely understand your project’s score, you will likely need to run your program on the server using the new input/output files. If you think your project grade was computed incorrectly, you must inform us **within at most one week** from when your score is posted on Blackboard. You must stop by one of the instructors’ offices during office hours and show us that your program runs correctly using the script and input/output files located on the server. If a mistake was made when grading your program, we will be glad to correct it.
- However, note that the following statements (which we hear quite frequently) will NOT cause us to increase your project score:
  - “I spent many hours working on the project, so I want a higher score.”
  - “My program contains over 1000 lines of code, so I want a higher score.”
  - “I didn’t follow all the instructions because I thought they were optional.”
  - “My program shouldn’t be tested on any valid inputs that I forgot to consider.”
  - “I didn’t understand the project description, and I didn’t ask for clarification.”