Griffin Cloonan
Landon Woollard
Nick Ponce

Lab 1: Part 2 Write Up

Programming Language: Python

Internal Architecture:

- Python File Objects — used to handle the contents of the students.txt CSV
- Python List Objects — Used to store lines containing query matches for later printing results

Task Log:

	Assigned	**Start Time**	**End Time**	**Work Hours**
User Input Parsing	Landon	(4/4/18) 2 PM	(4/5/18) 3 PM	2
Command Switch Cases	Landon	(4/4/18) 2 PM	(4/5/18) 3 PM	2
File Parsing	Nick	(4/4/18) 2 PM	(4/4/18) 3 PM	1
Line Query Function	Nick	(4/6/18) 2 PM	(4/7/18) 9 PM	3
Result Output	Nick	(4/4/18) 2 PM	(4/4/18) 3 PM	1
Test Suite	Griffin	(4/4/18) 2 PM	(4/9/18) 9:30PM	3
Write Up	Griffin Landon Nick	(4/4/18) 2 PM	(4/9/18) 9:30PM	2
Modifying Menus to accommodate new options	Landon	(4/11/18) 2 PM	(4/11/18) 3PM	1
Two-File Modification	Nick	(4/11/18) 2 PM	(4/11/18) 9 PM	2
New Result Output	Griffin Nick	(4/12/18) 12 AM	(4/15/18) 9 PM	3

Analysis Functionality and Output	Griffin Landon Nick	(4/15/18) 3 PM	(4/15/18) 9 PM	4
Debugging	Griffin Landon Nick	(4/11/18) 9 PM	(4/15/18) 9 PM	3
Additional Write Up	Griffin Landon Nick	(4/15/18) 4 PM	(4/15/18) 9 PM	1

Testing:

R4. S[tudent]: <last name>

- Find matching student last name
- Print last name, first name, grade, classroom assignment of student
- Print last name, first name of teacher

R5. S[tudent]: <last name> B[us]

- Find matching student last name
- Print last name, first name, bus route

R6. T[eacher]: <last name>

- Find matching teacher last name
- Print last name, first name of student

R7. G[rade]: <number>

- Find matching GPA number
- Print last name, first name of student

R8. B[us]: <number>

- Find matching bus route number
- Print last name, first name, grade, classroom of student

R9. G[rade]: <number> H[igh] or G[rade]: <number> L[ow]

- Find matching grade number with highest/lowest GPA
- Print last name, first name, GPA, teacher, bus route of student

R10. A[verage]: <number>

- Find all students with matching grade number and computer average GPA
- Print grade level and average GPA

R11. I[nfo]

- For each grade (from 0 to 6) compute the total number of students in that grade.
- Report the number of students in each grade in the format:
    - <Grade>: <number of students>

      sorted in ascending order by grade.

Issues Encountered:

- Python's raw_input() function throws an exception upon EOF, causing the program to non-gracefully fail with our test scripts since each test input file ended with an EOF. We solved this problem by catching that exception and gracefully ending the program.

- Communication issues on expected output caused tests to fail, but we quickly realized that the result outputted had been changed after communicating with the person implementing the tests.
- The input files were not standard throughout and our assumption that we could remove the newline characters and carriage return characters from everything at the end of a line was incorrect, because the teacher input file did not contain a newline at the end. We solved this by letting python's conversion functions deal with that when we converted the string to an int and then back.

Modifications Decisions For Part II:

Modifying the existing code for Part 1 of Lab 1 to fit the separation of the input file into two input files in Part 2 was rather simple. The original implementation had only one querying function which queried for a list of the students based on some criteria. To modify it we simply added a function that did the same for teachers and used it with the classroom criteria (a sort of foreign key) in the original student function to keep the format of the returned information the same for the rest of the existing code. We used these two functions to implement the new output functions.

The query language is two simple functions that function as described below:

```
list_of_student_data = queryStudentsByCriteria(criteria, search_type)
```

Where the function takes some single instance of student data as shown below and its type of search which is denoted by a correlated int also shown below in the format.

Format: StLastName(0), StFirstName(1), Grade(2), Classroom(3), Bus(4), GPA(5), TLastName(6), TFirstName(7)

```
list_of_teacher_data queryTeacherByCriteria(criteria, search_type)
```

Where the function takes some single instance of teacher data as shown below and its type of search which is denoted by a correlated int also shown below in the format.

Format: TLastName(0), TFirstName(1), Classroom(2)

Final Notes:

We decided to take the user query incrementally, rather than all at once. First, we ask our user for the type of search they would like to make (Student, Teacher, Bus, etc.). Once the initial choice has been made, syntax to complete the query is presented and the user only needs to finish the details. This simplify the user input parsing by breaking it into very clear stages, limiting the number of input errors to check. Furthermore, we added a '-t' interpretation flag to suppress all non-result print statements in order to get rid of user info that wasn't necessary for testing. In part 2, as we were updating our menu to have an option for "Analysis" we ran into the problem of the 'A' option already being used for "Average." We decided to use the second letter, 'N' for Analysis.