Ashley Moreno
Lab 1 Part b Write-Up

Team Members
None

Initial Decisions
For my implementation, I decided to use python as my programming language. I am comfortable with the language and its built-in libraries are ideal for tasks like file handling and data manipulation. The program was developed in a Unix-based environment. I used the text editor Vim and all debugging was done directly in the terminal by running the Python script and checking outputs or errors.

Internal Architecture
List (list):

- Used to store collections of sequential data, such as student records
- Example: students list holds individual student details like last name, grade, classroom, etc.

Dictionary (dict):

- Maps unique keys (e.g., classroom number) to values (e.g., list of teachers)
- Example: teachers dictionary maps classrooms to lists of teachers for fast lookups.

Set (set):

- Ensures uniqueness and removes duplicates
- Example: Used to eliminate duplicate teacher names when outputting teachers for a classroom.

Usage in Key Functions:

- find_by_last_name(last_name): Uses a list to search for students and a dictionary for efficient teacher lookup
- find_teachers_by_classroom(classroom): Uses a dictionary for quick teacher retrieval and a set to remove duplicates

Task Log

All tasks completed by Ashley Moreno

Task Name	**Time to Complete**
Project Design & General Set up	**~ 20 min**
Parsing & Loading Data	**~ 1 hr**

Re-implementing R4-R11	**~ 1 hr 30 min**
Implementing NR1-NR4	**~ 2 hr**
Implementing NR5	**~ 1 hr 10 min**
Completing Main() Calls	**~ 25 min**
Test Suite & Output	**~ 25 min**
Lab Write-Up	**~ 30 min**
README & Submission	**~ 15 min**

Modifications to Part A

Separation of Input Files:

Instead of reading all data from a single students.txt file, the program reads student information from list.txt and teacher information from teachers.txt. This required creating two separate functions: read_students(filename) and read_teachers(filename).

Mapping Teachers to Classrooms:

In the new format, each classroom has its own teacher(s), and this data is stored in the teachers dictionary, mapping classroom numbers to a list of teacher names (to account for co-teachers).

Affected Parts of the Code:

- Input Handling:
  - The previous students.txt reading code was removed and replaced with separate reading functions for list.txt (student data) and teachers.txt (teacher data).
- Teacher Lookup:
  - The code now uses a teachers dictionary to map classroom numbers to teachers, ensuring that students are associated with their respective teachers dynamically based on their classroom number.
- Classroom/Teacher Queries:
  - Any command that outputs teacher information (e.g., ‘S <lastname>’, ‘G <grade>’) was updated to look up the teacher(s) for the relevant classroom from the teachers dictionary.

New Commands:

- C <classroom_number>: Find students in a classroom.
- TC <classroom_number>: Find teacher(s) for a classroom.
- GT <grade_number>: Find teachers for a grade.
- E: Display classroom enrollments.
- GGPA: GPA by grade with averages.
- TGPA: GPA by teacher with averages.
- BGPA: GPA by bus route with averages.

Testing

When: Testing was performed throughout the development process, especially after completing each major functionality.

Who: Ashley Moreno

Duration: Approximately 2 hours were likely spent testing and debugging the entire program.

Bugs Found:

- Input validation- needing quotes around command
- String and list concatenation issues
- Duplicate entries in teacher lists
- Command overlaps (e.g., conflicts between T and TGPA commands)

Time to Fix Bugs: 1-2 hours

Final Notes
I was still unable to fix the input validation bug. While the program works as expected, still having to put quotes around each command. All else works.