Lab 1 (Part 1 & 2) Writeup

Team Members: Kaito Trias, Karla Sunjara, Nadia Wohlfarth

Initial Decisions (Part 1): We decided from the start that we would program in Python. One of the reasons for this decision was because we haven't used this programming language since early CS classes and we wanted to refresh our skills. Additionally, we enjoy coding in this language and it is useful to know well for job interviews so we figured it is beneficial to brush up now. We also decided to use GitHub because we know that this tool is used widely in industry and we should get practice with it and feel confident so that we can be successful after graduation. Also it is convenient and great for collaborating and we have learned git commands in our classes.

Initial Decisions (Part 2): For part two of the lab, we decided to use our part 1 code and make modifications. This is because the way we structured our part 1 lends itself to be modified to read in two files instead of 1, we edited functions, and added functionality. Also, due to the vagueness of the Analytics requirement for part 2, we discussed how we should implement it. We decided that we would add three commands: Analytics: G, Analytics: T, and Analytics: B which calculate the averages based on grade, teacher, and bus route respectively.

Notes on selected internal architecture (Part 1): We spent some time thinking about the data structures we would need for this program on Monday and we originally created a Student class. We were utilizing this class to store the different data components that made up a student in the data sample we were given. We later realized that this implementation was not necessary for the program. Instead we just directly read from the file. The data structure we are using is a Map for the info command. Since every grade needs to have a total number of students, for every grade we lookup in the map if there is a number associated with the grade. The map works as a counter for every grade. We also originally wrote a test script and then decided to test manually.

Notes on selected internal architecture (Part 2): For part 2 we used dictionaries for the Analytics part with grade being the key and the value being a list of numbers of students and GPA, teacher being the key - value remains same, and bus route being the key - value remains same.

Task Log (Part 1): All three of us brainstormed our plan of attack for this lab during the lab period of our first class (Friday, 9/20 from 4-5pm). Nadia was designated to work on creating the writeup. Our next working session was on Monday during class (9/23) and Nadia and Karla pair programmed starting with reading in the file, beginning to parse, and the specific inputs from the command line. Kai worked Tuesday morning on the program for 2 hours and added the student, instructor, and average functions. Nadia, Karla, and Kai worked Tuesday evening as well (9/24). Karla and Nadia worked on creating and documenting all of the test cases during lab and outside of lab for an hour on Wednesday(9/25). Also during this time Kai fixed bugs that were found through testing our code. On Thursday (9/25) Karla spent another 1.5 hours making sure the tests worked after several code changes and creating the tests.out deliverable. We all spent 1.5 hours adding finishing touches, cleaning up our code, created the README, and zipped all of the files(9/26).

Task Log (Part 2): Nadia, Karla, and Kai worked on Sunday (9/29) from 1-4. During this session we made decisions on how we will program, and we pair programmed. Karla contributed mainly to the Analytics section, Kai contributed by adding new functionality described in part 2, and Nadia made initial modifications for handling 2 files and wrote the test script. Then, the three of us worked on Monday(9/30) during lab time to manually test our code and add finishing touches. On Sunday (10/6) Karla spent an hour adding more tests to the script and updated the readme to match the new functionality. The three of us reviewed the writeup, manually tested our program one last time and added finishing touches during lab on Monday (due date).

Notes on testing (Part 1): Karla and Nadia performed manual testing on the code and then developed the test script on Tuesday (9/24). We used the test driven development approach and came up with the majority of our test cases before the program was finished. While manually testing our program, we realized we had issues with the empty line implementation so we went back and corrected it to accommodate the empty space. When we were manually testing we tried to come up with every edge case we could (No colon, misspelled last name, multiple last names in one command, numbers when a string is expected, lowercase last names, etc.). We realized that we had a bug when we tried to type Grade or G as a command. We found that we had a typo in our code (a variable name) which was causing this error! Our testing also helped us find and fix small bugs in our Quit and Info commands, as well as correctly handle cases in which students.txt was not in the directory or was an empty file.

Notes on testing (Part 2): Nadia added tests for the Additional Functionality: Extended Search (NR1 - 4) and also for the Additional Functionality: Analytics (NR5). Due to the similarity of the code from part 1, we did not discover any major bugs during testing. The kinds of errors we ran into all mainly dealt with output spacing.

Additional Part 2 Writeup Requirements:

- Decisions you made on how to modify your Part 1 code to accommodate new input data. Which parts of the code were affected?

  We changed the validations for checking if list.txt is a valid file. In part 1, we stripped each line with a comma delimiter and checked if there were 8 values in the list. Now we check if there are 6 to accomodate for the new text file.

  We refactored the code so that finding a teacher was a separate function. The function reads teachers.txt and returns a list of teachers that have the same room number. The logic that calls the function then prints the teachers names in the list.

  We changed accepting one file into accepting two files to be read into our program. This affected multiple functions but was a minor change. Functions affected: find_teacher(classroom), student_command_with_b(last_name), student_command_without_b(last_name), find_student(classroom), teacher_command(user_input), grade_command(user_input), bus_command(user_input), average_command(user_input), and info_command().

- Syntax and semantics of any additions to the query language, complete with examples : Here are examples of the queries for the NR1-5 requirements.

  <u>NR1-4 requirements</u>

  NR-1: N: <number>

  - Ex. N: 101

  NR-2: NT: <number>

  - Ex. NT: 101

  NR-3: GT: <number>

  - Ex. GT: 3

NR-4: E

<u>NR-5 Cases</u>

Analytics: G

Analytics: T

Analytics: B

Overall, we had a really harmonious group and worked well together!