Team Members: Dylan Carr, Amanda Cheng

Initial Decisions: We decided to write our program in Java and uploaded our source code into GitHub so that we could easily collaborate on the code.

Part 2 Decisions: We had to rewrite readFile() such that it would read the new textfiles and create separate Student and Teacher objects. Each of the functions we wrote for part 1 now had to accommodate a new ArrayList of Teacher objects as well as the already available ArrayList of Student objects so that it could still print the needed output.

Notes on selected internal architecture: We used an ArrayList to store the list of students once we read in the student file and an ArrayList for the list of teachers as well. The ArrayList holds Student/Teacher objects that are created using each of the given characteristics of each student.

Task Log:

- Read in files: Dylan
- Updating overall main function: Amanda
- Part 1 Functions adjusted to new data structure architecture: Amanda
- NR1/NR3 requirements: Amanda
- NR2/NR4 requirements: Dylan
- Analytics - Grade & Teacher: Dylan, Bus Route: Amanda
- Write Up Part 2: Dylan
- Test Suite: Dylan & Amanda

Testing Notes: We ran into an issue where the Grade: Number Low instruction was printing out only the first student in the file. It took about 20 minutes to debug, and we discovered that it was because the student ArrayList index was not being updated correctly. Each function was relatively quick to write--they each took approximately 5-10 minutes to write.

Part 2 Decisions: Our first decision was to adjust the Student object to match the new students.txt input file format as well as create a new Teacher object to match the new teacher input file. These changes caused us to have 2 arrayLists of objects to store all the data. The old functions had to be adjusted to take in 2 arrayList parameters when needed and loop through them each appropriately.