CSC369-01
Assignment: 3
Name: Winton Gee

Q1[10].
Your goal is to print the names and addresses of students that have taken all the top N most difficult classes.

General summary
- Determine the top N most difficult classes
- Determine which students took the most difficult classes

Determine the top N most difficult classes
- TopNMapper:
    - TreeSet t
    - (1, “CSC365, 1”) =>
        - No output
        - Add Record(CSC365, 1) to t
        - Make sure size(t) <= N
            - If size(t) > N, then remove last record
        - Cleanup
            - For (el : t) context.write(null, el);
    - Class Record
        - String className;
        - Integer classDifficulty;
        - Int compareTo(other)
            - Res = classDifficulty.compareTo(other.classDifficulty);
            - If (res == 0) return className.compareTo(other.className);
            - Return res; // desc so do not multiply by -
- TopNReducer
    - Treeset t
    - (null, Record(CSC365, 1)) (null, Record(CSC469, 2)) …
    - reduce(key.values)
        - For (el : values)
            - t.add(el)
            - If size(t) > N, then remove last record
        - Cleanup
            - For (el : t) context.write(null, el);

Determine which students took the most difficult classes
Mapper
- (1, “John, 123 Main, 233 223 5566, (CSC365 CSC369 CSC469)”) =>
    - Only need information about name, address, classes
        - (“John, 123 Main, (CSC365 CSC369 CSC469)”)

Reducer

- Store the result of mapper somewhere and use it for the reducer
- The reducer doesn't really have much to do besides printing out information of the students that have taken all N of the difficult classes
    - If (student has taken all of the top N most difficult classes)
        - print requested info using the context write

Q2[10].

The problem is to print the N students with the highest GPA.

General Summary

- Determine/Calculate the gpa of all students
- Print the information of students with top N highest GPA
- GPA => Sum values then divide by count of values

Answer

TopNMapper:

- TreeSet t
- (1, "John, 123 Main, 233 223 5566, ((CSC365 A) (CSC369 A) (CSC469 B))") =>
    - No output
    - Convert grades to their int value : gpa, A = 4, B = 3, …
    - Add Record(studentName, gpa) to t
    - Make sure size(t) <= N
        - If size(t) > N, then remove last record
    - Cleanup
        - For (el : t) context.write(null, el);
- Class Record
    - String studentName;
    - Double gpa;
    - Int compareTo(other)
        - Res = gpa.compareTo(other.gpa);
        - If (res == 0) return studentName.compareTo(other.studentName);
        - Return res; // desc so do not multiply by -

TopNReducer

- Treeset t
- (null, Record(John, 3.95))…
- reduce(key.values)
    - For (el : values)
        - t.add(el)
        - If size(t) > N, then remove last record
    - Cleanup
        - For (el : t) context.write(null, el);

Q3[10].
The problem is to print the top N most popular classes (i.e., classes with the highest enrollment).

General Summary

- Job 1: Mapper -> Combiner -> Reducer
- Job 2: Mapper -> Reducer

Answer

Job 1

Mapper

- (1, “3, CSC354”) => (CSC354, (1)) // 3 would be the student id, (1) is count

Combiner/Reducer

- (CSC354, 1) (CSC354, 1) … (className, count) => (className, sumCount)

Job 2

TopNMapper:

- TreeSet t
- (1, “CSC354 2”) =>
    - No output
    - Add Record(className, count) to t
    - Make sure size(t) <= N
        - If size(t) > N, then remove last record
    - Cleanup
        - For (el : t) context.write(null, el);
- Class Record
    - String className;
    - Integer count;
    - Int compareTo(other)
        - Res = count.compareTo(other.count);
        - If (res == 0) return className.compareTo(other.className);
        - Return res; // desc so do not multiply by -

TopNReducer

- Treeset t
- (null, Record(CSC354, 2))…
- reduce(key.values)
    - For (el : values)
        - t.add(el)
        - If size(t) > N, then remove last record
    - Cleanup
        - For (el : t) context.write(null, el);