COSC2430 hw2: Sorting Algorithms with Linked List

1. Introduction

Given a list of credentials as input, you will need to implement a C++ program to add these input objects into a linked list. Within the linked list, your program needs to perform different adding, removing and sorting operations base on the given commands. This homework will focus on linked list implementation and simple sorting techniques. When submit your assignment, please name the folder on the server as "hw2".

2. Input files

- The input file will contain a list of credentials (ranging from 0 to 100).
- Each credential represents a node in the linked list and should be added one by one to the end of the linked list.
- Each credential will have four attributes: id, username, score, and grade.
    - Note: id will always contain 4 digits ranging from 0 to 9. username will always contain lowercase alphabet character (a – z), no spaces or special character included. score will range from 0 to 100. grade is given between A, B, C, D, and F.
- The formatting of each credential is as follow:

```
[id:value;username:value;score:value;grade:value]
```

- Valid credential should have all attributes present and appear in this order: id, username, score, grade.
    - Example of valid credential:

```
[id:1234;username:spongebob;score:100;grade:A]
```

    - Example of invalid credential:

`[id:1234;username:steve;grade:C]` – missing attribute: score

`[id:1234;grade:B;score:85;username:batman]` – out of order

- Invalid credential should be ignored.
- The input will <u>not</u> contain any empty lines or blank spaces.
- In the case when the input is empty, continue to process the command.
- While reading the input, \n and \r should be removed before processing string.
- Input might contain duplicate id credential or duplicate username credential, please read section 5 below on how to process duplicate cases.

3. Command files

- There will be three types of command: `Add`, `Remove`, and `Sort`. The commands should be executed in the order that they appear (latter command should always run based on the former command's result).

    - `Add (index) [credential]`
        - The `Add` command will be followed by an integer inside parenthesis (represent the index in the linked list to be added) and then followed by a credential.
        - If index = 0, meaning the credential should be added at the beginning of the linked list.
        - If the index = size of the linked list, meaning the credential should be added at the end of the linked list.
        - If the index > size of the linked list, meaning index out of bound, in this case the credential should not be added to the linked list.

        - Ex: `Add (0) [id:1234;username:tom;score:50;grade:F]`
            add this credential at the beginning of the list

        - Ex: `Add (3) [id:1234;username:tom;score:50;grade:F]`
            add this credential at the third index of the list, if not out of bound

        - Add command might contain duplicate id credential or duplicate username credential, please read section 5 below on how to process duplicate cases.

    - `Remove (attribute:value)`
        - The `Remove` command will be followed by an attribute and its value (placed inside parenthesis).
        - Every credential that contains the attribute with the matching value should be removed from the linked list.
        - Ex: `Remove [grade:C]`
            remove every credential that contain value "C" in attribute "grade".

    - `Sort (attribute)`
        - Sort the linked list base on the given attribute (placed inside parenthesis).
        - Sort ascending for id.
        - Sort alphabetically for username and grade.
        - Sort descending for score.
        - When sorting if two values are similar, don't swap their position.

- The command will <u>not</u> contain any empty lines or blank spaces.
- While reading the command, \n and \r should be removed before processing string.

4. Output files

- The output file should display every credential in your linked list after executing all the command in the command file.
- Each credential will be on its own line.

5. Adding Operations

- When adding a credential to the linked list, the credential might contain duplicate id or duplicate username.

- In the case when the credential contains duplicate id, update the previous credential's attributes with the new credential's attributes.
    - Ex: If the credential `[id:2468;username:onion;score:75;grade:C]` is in the linked list when adding `[id:2468;username:lettuce;score:50;grade:F]`, since the two credential have matching id, update the username, score, and grade of the credential in the linked list to the latter credential. The username for id:2468 should be updated to lettuce, the score should be updated to 50, and the grade should be updated to F.

- In the case when the credential contains duplicate username (but doesn't have matching id), the credential should be ignored and not to be added to the linked list.

- When adding from both the input and command files, always check if the credential has all the attributes present and following the order of id, username, score, grade. Invalid credential should be ignored.

6. Requirements

Homework is individual. Your homework will be automatically screened for code plagiarism against code from the other students and code from external sources. Code that is copied from another student (for instance, renaming variables, changing for and while loops, changing indentation, etc. will be treated as copy) will be detected and result in "0" in this homework. The limit is 50% similarity.

7. Turn in your homework

Homework 2 needs to be turned in to our Linux server, follow the link here
https://rizk.netlify.app/courses/cosc2430/2_resources/

Make sure to create a folder under your root directory, name it "hw2" (case sensitive), copy all your .cpp and .h file to this folder, "ArgumentManager.h" need to be included as well.

PS: This document may have typos, if you think something illogical, please email TAs for confirmation.