CS/CE 1337 – PROJECT 3 – FreePlay Database II

Pseudocode Due: 10/12 by 11:59 PM

Project Due: 10/31 by 11:59 PM

KEY ITEMS: Key items are marked in red. Failure to include or complete key items will incur additional deductions as noted beside the item.

Submission and Grading:

- The pseudocode will be submitted in eLearning as a Word or PDF document and is not accepted late.
- All project source code will be submitted in zyLabs.
    - Projects submitted after the due date are subject to the late penalties described in the syllabus.
- Programs must compile using gcc 7.3.0 or higher with the following flags enabled
    - -Wall
    - -Wextra
    - -Wuninitialized
    - -pedantic-errors
    - -Wconversion
- Each submitted program will be graded with the rubric provided in eLearning as well as a set of test cases. These test cases will be posted in eLearning after the due date.
    - zyLabs will provide you with an opportunity to see how well your project works against the test cases. Although you cannot see the actual test cases, a description will be provided for each test case that should help you understand where your program fails.
- **Type your name and netID in the comments at the top of all files submitted. (- 5 points)**

Objective: Use structures and pointers to create linked lists. Use knowledge of pointers to modify linked lists. Implement a recursive function.

Problem: FreePlay Arcade in Richardson has a primitive flat file database that needs to be updated. This database contains the names, high scores (w/ initials) and the total number of plays of their games. The database also contains the total amount each game would earn if not on free play. You will build a simple interface to interact with the database.

The logic of this project is very similar to project 1. Changes from project 1 are colored blue.

Pseudocode: Your pseudocode should describe the following functions

- Adding a record to the list
- Deleting a record from the list
- Searching the list for a record
- Editing a record in the list
- Printing the list to the database file
- Sorting the list

For each function:

- Determine the parameters
- Determine the return type
- Detail the step-by-step logic that the function will perform

Details:

- The name of the file you submit will be named **`main.cpp`**
- Start the program by prompting the user for the following information in the order listed
    - Database filename
    - Batch filename
    - Log filename
- These would normally be hardcoded in an application, but zyLabs requires a filename for multiple test files
- All records are stored in memory with a linked list of structures (-10 points if not)
- All manipulation to the database will happen within the linked list, not the database file (-10 points if not)
- Expect that some test cases will involve there being no database file or an empty database file at execution
- The interface will provide the following options:
    1. Add a record to the database
    2. Search for a record and display it
    3. Edit a record
    4. Delete a record
    5. Sort records
- **Add a record:** Create a new node and add it to the end of the list
- **Search for a record:** The search term will be a word or phrase. Search through the entries and display the complete record for any game that matches the search term.
- **Edit a record:** With a given game name, the program should update the record and confirm the change by displaying the new record on the screen. The following items can be edited:
    1. High score
    2. Initials
    3. Number of plays
        - If number of plays is edited, the revenue should be recalculated
            - $0.25 per play
- **Delete a record:** User will enter a game name. The program should delete the record from the file. The best way to do this is to delete the record from the linked list.
- **Sort records:** The records can be sorted in ascending order by either name (A -> Z) or number of plays (low -> high).
- Input will come from a batch file. Output will be written to a file.
    - The linked list will be written to a file at the end of the program
    - The new database file will be named `freeplay.dat`
        - No need to remove and rename when a delete is performed
- Revenue is based on a quarter per play
- You can expect all input to be valid.

Database Format: The database will exist in a file. Each record will be on a separate line in the file and each line will have a new line character at the end of the line (except for the last line which may or may not have a newline character). Each record in the database will have the following format. Notice that each field will be separated by a comma and a space. There will be no leading zeroes because the data manipulation will happen in memory instead of the file as it was with project 1.

```
<name>, <high_score>, <initials>, <plays>, <$><revenue><\n>
```

- <name> - may be multiple words
- <high_score> - 1-9 digits
- <initials> - 3 characters – no white space
- <plays> - 1-4 digits
- <revenue> - <1-4 digits><decimal><2 digits>

Input: All input will be file based. Prompt the user to enter the filename for the database file. This would normally be hardcoded in an application, but zyLabs requires a filename for multiple test files. A batch file will be used with the database. Prompt the user to enter the filename for the batch file. Each line in the file will be a command to perform on the database. Each command will be on a separate line in the file and each line will have a new line character at the end of the line (except for the last line which may or may not have a newline character). The format for each option is listed below. There is a single space between fields.

- Add a record
  - `1 "name" high_score initials plays $revenue`
  - The double quotes surround the name so that you know where the end of the name is
  - The values are not required to have leading zeroes
- Search for a record
  - `2 <search term>`
  - Search term may contain spaces
- Edit a record
  - `3 "name" <field number> <new value>`
  - `<field number>`
    - 1 = high score
    - 2 = initials
    - 3 = number of plays
  - The new value is not required to have leading zeroes
  - The double quotes surround the name so that you know where the end of the name is
- Delete a record
  - `4 <name>`
  - Name may contain spaces
- Double quotes are not necessary here since there is no data after the name
- Sort records
  - `5 <name/plays>`
    - A single word should follow the value: `name` or `plays`

Output: Each command in the input file will generate output to a log file. Prompt the user to enter the filename for the log file. After each command output, write 2 blank lines to the file. The output for each command is as follows:

- Add a record
  - `RECORD ADDED`
    `Name: <name>`
    `High Score: <high_score>`
    `Initials: <initials>`
    `Plays: <plays>`
    `Revenue: $<value> - formatted to 2 decimal places`
- Search for a record
  - `<name> FOUND or <name> NOT FOUND`
  - If found
    - `High Score: <high_score>`
      `Initials: <initials>`
      `Plays: <plays>`
      `Revenue: $<value> - formatted to 2 decimal places`
- Edit a record
  - `<name> UPDATED`
  - `UPDATE TO <field> - VALUE <value>`
    - Possible fields: `high score, initials, plays`
  - `Name: <name>`
    `High Score: <high_score>`
    `Initials: <initials>`
    `Plays: <plays>`
    `Revenue: $<value> - formatted to 2 decimal places`
- Delete a record
  - `RECORD DELETED`
    `Name: <name>`
    `High Score: <high_score>`
    `Initials: <initials>`
    `Plays: <plays>`
    `Revenue: $<value> - formatted to 2 decimal places`
- Sort records
  - `RECORDS SORTED BY <name/plays>`
  - Display all records (one per line) in the proper order
    - Line format: same as the database format