CSCI 2421 Final Project

For many movie lovers, actors and directors, the annual Academy Awards are the highlight of the year.
Everyone dresses up, they walk the red-carpet, listen to long and boring speeches and generally pat themselves on the back…but have you ever wondered which movies are the top movies, or who has received the most awards.
Well you could ask google, but we are going to do our own data analysis.

The purpose of this final project is to help you think about the design of a somewhat complicated project, then implement and test your code.
We want you to start first with the design.
Read these requirements and make a design document (answering the questions provided).
Create a design document complete with the objects and flow of data, as well as a decision on the best data structures to use for each component.
Then you will be ready to implement the code and test.
Please don't leave things until the last two weeks.
Get started now, and please ask your instructor and TA for help BEFORE you get too lost.
Get the big picture done first. Worry about the structure and implementation of the major functionality.
Then if you have time, work on the little details, and minor error checking.

So now onto the requirements. ….And the Oscar goes to….

For your Final Project you will develop a simple database system. The database is to handle multiple records, each composed of several fields.
The database will store its information to a file, addition and deletion of records, field modifications, and it will allow users to sort records based on the selected keys, and produce reports (output) according to predefined criteria.

Some definitions:

1. A database is a collection of information, or data, that you can organize, update, sort, search through, and print as needed. A database does not just hold information; you use a database to organize and analyze information so that you understand its significance.
2. A database file consists of one or more records. Each record holds all the information about one subject item. In C++, the *class data type* provides an efficient way to represent and manipulate records.
3. Each piece of information in a record is called a field. Fields store the data that has been entered or calculated. In C++, fields are nothing more than the member variables defined for a particular class.

Requirements

Given the requirements as a rough specification, you are to design the classes and implement the database. So you can consider the requirements below as an outcome from a meeting with a client. You are in full control of the choice of data structures **(except the main data structure of a Binary Search Tree, more detail below)**, algorithms, internal file format, and detailed user interface scheme.
Requirements are listed with R:
You are designing and implementing a database for the Academy Award winners.
R1/R2: You are to read in information from two files.

R1: You will read in actor-actress.csv which is formatted as a csv file (Ask the user for the file name). Then place the items in a Binary Search Tree, sorted by name.

This is a common format and is comma separated (instead of being on separate lines). So you will have comma's between the values. Blank values will just have a comma noting to go to the next field. (so you may have value,, indicating a blank field.) Each line ends in a newline, not a comma. If you want to view the file, often this will be opened by a spreadsheet unless you specifically open it with a text editor. Do not open it with Microsoft Word, as this may change the format. Consider using getline with three parameters, as an easy way to read in files The first line of a CSV file notes the data descriptions as follows:
Year,Award,Winner,Name,Film

The Winner field has a one if they won and a zero if they did not win.

R2: Then you will read in information about the movies (called pictures by the Academy of Motion Pictures) that have won best picture award. Place these items in a Binary Search Tree, sorted by name. This is also formatted as a .csv file

The first line of pictures.csv contains the data fields including:
name,year,nominations,rating,duration,genre1,genre2,release,metacritic,synopsis

R3/R4: Choose either the movie or actor database and add a record

R5/R6: Choose either the movie or actor database, search for a record, and modify the fields.

R7/R8: Choose either the movie or actor database, search for a record and delete the fields.

R9/R10: Choose either the movie or actor database and sort by any single (sortable) field

R11/R12: Choose either the movie or the actor database and do a complete search on any "complete" searchable field. It is unlikely that you would have an exact match on an entire description, so that would not be listed to search.

R13/R14: Choose either the movie or the actor database and do a partial search on any searchable field. A partial search is any substring within a field.

R15/R16: Choose either the movie or actor database, ask for a file name, and print out a .csv file of the latest database (after adds, deletes or modifies). Remember that the first line of a .csv file lists the name of the fields separated with commas, ending in a newline. Then the following lines are the information from the fields separated with commas, ending in a newline.

Database overall management

1. Use a text based menu for users to choose available features. Program should have a main menu at the beginning and sub menus depending on the task.
2. Each component of the overall program should be fairly modular.
    a. Each menu item, for example, should be a separate function. The menu feature should be handled by a separate function and not by main( ).
3. Program should be fairly fault tolerant of user input (both when the user is entering data, and making menu choices). Provide appropriate user prompts and on-screen directions
4. Split the program into multiple files based on the roughly categorized functionality.

Data Retrieval and Modification

1. Users should be able to search records based on the field information in two modes: exact and contains. For example, search "Justin". Then under the search sub menu, users have to pick the search mode (exact or contains) and the field. (Fields should be listed out in another menu, so the user doesn't need to remember a specific field name.
2. Quite often, searches may generate a relatively big output. Users should be able to search again within the search result (secondary search) or start all over again from scratch (new search).
3. Since the entire data is structured in a Binary Search Tree of names, any search (except name of movie or actor) will have to traverse the entire tree and search through the designated fields in every node of the tree.
4. There should be no restriction to the number of records in the database. So, in other words, you should not consider a fixed array for the record data structure.

Submission Guideline

You need to submit following items (all zipped together):

1. Source code with reasonable comments
2. Makefile that works (and is tested) on the csegrid.
3. Readme.txt file noting status of what works and what doesn't
4. A single final report that includes:
   - Summary of provided functions. This should be matched with the requirements
   - Design that shows the overall program structures, and the explanation of key algorithms. A description of user interface scheme is required to explain the menu items at top level and items in sub menus and how to navigate through menus. A detailed instruction and sample skeleton is available from **Design Document.**
   - Accurate status of the program, what's done, and what's not completely implemented.
   - Accurate status of testing on the csegrid.
   - The final report should be in MS Word, or PDF format.

Grading Criteria

A. All work MUST be your own. Any and all help received from another person (other than the TA or instructor), or any source (other than the textbook) must be documented in comments. We reserve the right to give you a zero for the final project if we feel you did not do your own work.

B. Submitting a working program that provides all of the required features will result in a maximum grade of 90%.

C. Documentation explained above will result in the additional 10% of the grade. Note if you spend time getting it right for HW7, you may not have anything but update the status for the final project.

D. Any or all of the following will result in point deductions of up to 5% for *each* infraction.
   1. Poor and/or inconsistent programming style. This includes the following:
      a. Improper use of indentation.
      b. Overuse of global variables.
      c. Failure to keep functions modular and reusable (possibly applicable to other programs).
      d. Insufficient comments.
   2. Insufficient menu prompts
   3. Program is not *reasonably (not absolutely)* fault tolerant.
      a. Test to ensure that your program cannot be crashed or sent into an infinite loop by a user who is not following directions.

E. Partial credit may be awarded.

    1. You may get partial credit for non-working modules (functions) by explaining (in the final document under status) where you think the problem lies.

F. Submitting a program that does not compile on csegrid.ucdenver.pvt may result in a deduction of at least 20%. Additional points will be lost for each required feature that is not adequately addressed or if we can not easily determine whether your code is correct. Usually if it doesn't compile you are looking at a 50% penalty since it is difficult to see functionality if it isn't running.

Extra Challenge / Extra Credit

You can get up to 20% extra credit if you earned at least 80% on the project. Some of you may want an extra challenge to boost your abilities and have some interesting resume material. If you want an additional challenge, you can incorporate the nominations.csv file (which has the same format as the actress-actors file but adds many categories and about 10,000 records. In addition to incorporating the nominations, you should provide options for three or four statistics. Look at the data to decide what you could provide. Examples: List of Directors and number of nominations (sorted by number of nominations) or top 20 words in titles….be creative with the data. Your data structures will become very important to work with this many records. If you do the extra credit, please note this in your documentation and make it clear in the running of the program. Note: Since this is extra credit, it needs to meet a higher standard for full extra credit).