AProject Design/Implementation Document

Academy Awards Project
Warriors of Remote Learning
Ravisher Singh, Bruce Tieu
Due 11/8/2020

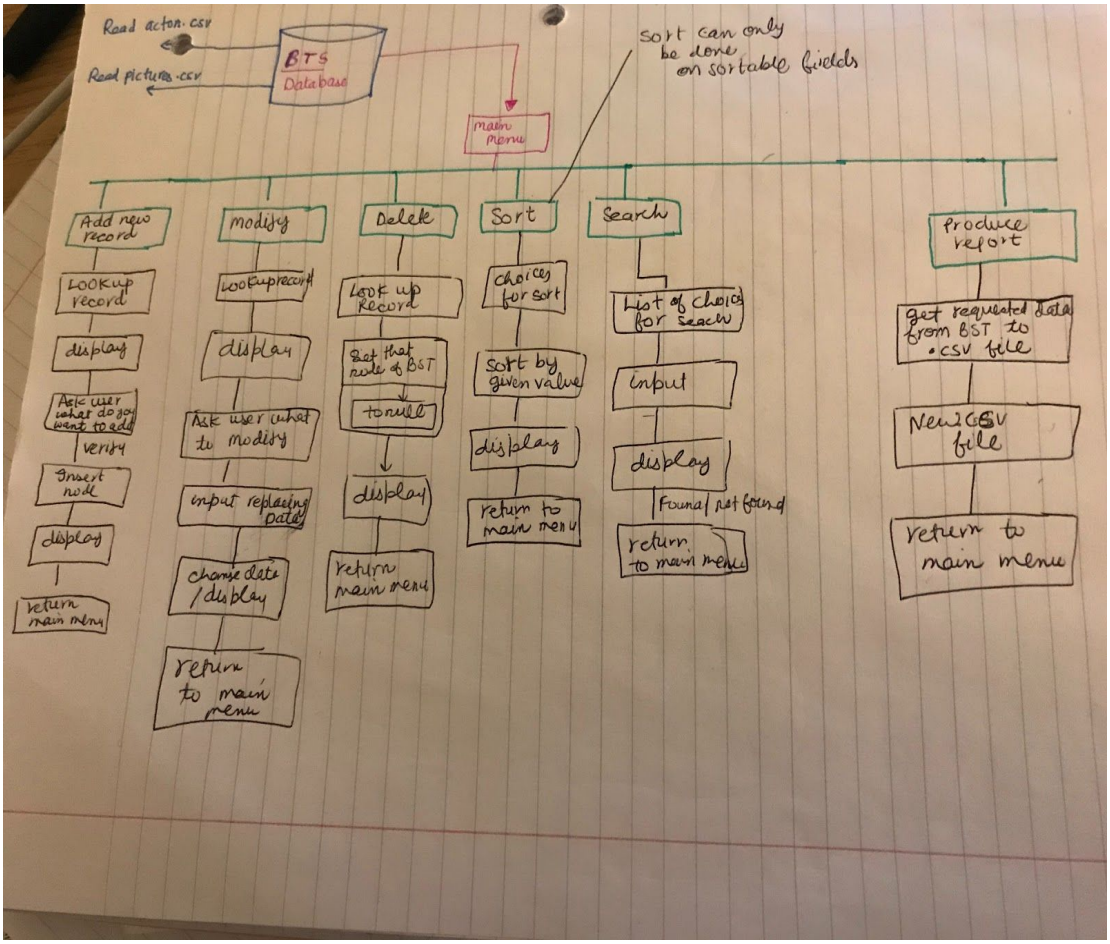
2. Problem Description

We develop a movie database system of academy Awards winners. The database is to handle multiple records, each composed of several fields. There are 2 .csv files, one for actor/actresses and another one for movies. We store them into 2 binary trees. After we read the information from the csv file, we should be able to add, modify, sort, and search entries. It will allow the user to sort records based on the selected keys and produce reports (output) according to predefined criteria.The database will store its information in a file, and it will update the file based on addition & deletion of records and field modifications and also print the updated database to a new csv file.

3. Overall Software Architecture

A brief description of major functions and their main roles in the program. You need to explain how the entire program is constructed and how the functions are related to each other. You don't have to explain every little function. A diagram to display each relation is very useful to get an overall picture. Here's an example of the diagram for a different program but you'll get an idea.

Data structure to use: Binary search tree, it will act as a database. The user will have different options there.



We can maybe have two classes, one for actresses, and the other for pictures. We could also have a parent class called database, and then the child classes will be the actors and pictures.

4. Input Requirements

A detailed **list of all external inputs** (from files or keyboard) including a description of the **data type** and **range of valid values** for each input. For input file format and interactive user input, you need to write what data type is used for every field and valid value and length.

File inputs:

1. Read in actor-actresses.csv
    a. Data types:

i. Year: int
ii. Award: string
iii. Winner: boolean?
iv. Name: string
v. Film: string

b. Range of valid values:
i. Year: 1928 - 2016
ii. Award: “Actor In A Leading Role”, “Actor In A Supporting Role”, “Actress In A Leading Role”, “Actress In A Supporting Role”
iii. Winner: 1 (True) or 0 (False)
iv. Film: “... And Justice For All” - “Zobra the Greek”

2. Read in pictures.csv
a. Data types:
i. Name: string
ii. Year: int
iii. Nominations: int
iv. Rating: float
v. Duration: int
vi. Genre1: String
vii. Genre2: string
viii. Release: string
ix. Metacritic: int
x. Synopsis: string

b. Range of valid values:
i. Name: “12 Years A Slave” - “You Can’t Take it With You”
ii. Year: 1927 - 2014
iii. Nomination: 0 - 14
iv. Rating: 0 - 10
v. Duration: 90 - 238
vi. Genre1: “Action”, “Adventure”, “Biography”, “Comedy”, “Crime”, “Drama”, “Musical”, “Western”
vii. Genre2: “Biography”, “Comedy”, “Crime”, “Drama”, “Family”, “Film-Noir”, “History”, “Musical”, “History”, “Romance”, “Sport”, “War”, “Western”, empty string
viii. Release: “January” - “December”
ix. Metacritic: 0 - 100
x. Synopsis: Any string

<u>Keyboard inputs:</u>

1. Choose either the movie or actor database and add a record
    a. Data types: string
    b. Range of valid values: Any string which is a movie or actor / actress
2. Choose either the movie or actor database, search for a record, and modify the fields.
    a. Data types: string
    b. Range of valid values: Any string which is a movie or actor / actress
3. Choose either the movie or actor database and sort by any single (sortable) field
    a. Data types: string
    b. Range of valid values: Any string which is a movie or actor / actress
4. 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 for search.
    a. Data types: string
    b. Range of valid values: Any string which is a movie or actor / actress
5. 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.
    a. Data types: string
    b. Range of valid values: Any string which is a movie or actor / actress

5. Output Requirements

A detailed **list or description of all outputs (**to files) including a description of the **data type** and **range of valid values** for each output.

File Outputs:

1. Choose either the movie or actor database, and print out a .csv file of the latest database (after adds, deletes or modifies).
    a. Pictures database:
        i. Data types:
            1. Name: string
            2. Year: int
            3. Nominations: int
            4. Rating: float
            5. Duration: int
            6. Genre1: String
            7. Genre2: string
            8. Release: string
            9. Metacritic: int

10. Synopsis: string

ii. Range of valid values:

1. Name: “12 Years A Slave” - “You Can’t Take it With You” including database record modification, deletion or addition
2. Year: 1927 - 2014 including database modification, deletion, or addition
3. Nomination: 0 - 14 including database record modification, deletion, or addition
4. Rating: 0 - 10 including database record modification, deletion, or addition
5. Duration: 90 - 238 including database record modification, deletion, or addition
6. Genre1: “Action”, “Adventure”, “Biography”, “Comedy”, “Crime”, “Drama”, “Musical”, “Western” including database record modification, deletion, or addition
7. Genre2: “Biography”, “Comedy”, “Crime”, “Drama”, “Family”, “Film-Noir”, “History”, “Musical”, “History”, “Romance”, “Sport”, “War”, “Western”, empty string including database record modification, deletion, or addition
8. Release: “January” - “December” including database record modification, deletion, or addition
9. Metacritic: 0 - 100 including database record modification, deletion, or addition
10. Synopsis: Any string including database record modification, deletion, or addition

b. Actors-actresses Database

i. Data types:

1. Year: int
2. Award: string
3. Winner: boolean?
4. Name: string
5. Film: string

ii. Range of valid values:

1. Year: 1928 - 2016 including database record modification, deletion, or addition
2. Award: “Actor In A Leading Role”, “Actor In A Supporting Role”, “Actress In A Leading Role”,

"Actress In A Supporting Role" including database record modification, deletion, or addition
3. Winner: 1 (True) or 0 (False) including database record modification, deletion, or addition
4. Film: "... And Justice For All" - "Zobra the Greek" including database record modification, deletion, or addition

Console outputs:

2. Print on console , as per what do user wants to print on screen/file

6. Problem Solution Discussion

We will use a binary search tree to store, search, add, modify, and delete data. There will be separate functionality for each of those operations. For the process of printing the data and sending it to the new csv file we will create a function that takes in a binary search tree and transfers data to the new csv file. There will be a main menu to display the various options to select from. First we read in the file, then the user can add a new record. This is where we will be adding new nodes in the BST. When a user uses the search function, depending upon the search he/she will be able to modify/delete the record. When the user searches for a record using a different criteria we will traverse the BST and add the nodes. Once the fields are sorted the user should be able to print to console or move it to a csv file.

7. Data Structures

Candidates:

1. Vector:
    a. Advantages: Not only are vectors easy to implement, but vectors can dynamically resize an array so it is good for adding additional elements into the vector if needed. It is also easy to remove elements and access elements. For time complexities, it takes O(n) time to add and remove elements in the array. If the index is known, then constant time access is achieved; ie O(1).
    b. Disadvantages: Vectors can be time and memory expensive, especially when the amount of data to be stored grows substantially.
2. Binary Search Tree
    a. Advantages: It is more efficient than an array when it comes to inserting, deleting, and looking up elements since the time

complexities for these actions are O(log(n)). We can also traverse the tree in a variety of ways (inorder, preorder, and postorder) which is helpful for returning data in a specific order. Furthermore there are many types of binary trees such as a B+ tree, red-black tree, etc which can be used for different purposes.

b. Disadvantages: The binary search tree may not always be balanced. In this case, it could be more costly to balance the tree as nodes are inserted / removed.

3. For searching results I considered array, vector, linked list, stack and heap. Arrays are not dynamic so cannot be used, stacks are very hard to sort, heaps does not directly delete the elements. Vector is the most eligible choice here because of ease of delete and modification of elements.

8. User Interface Scheme

User interface scheme should show the menu items at top level and items in sub menus and how to navigate through menus.

Possible Scheme:

Loop:

a. Read a file
    i. Pictures.csv
    ii. Actor-actresses.csv
b. Add a record
    i. Choose database
        1. Pictures
            a. Name?
            b. Year?
            c. Nominations?
            d. Rating?
            e. Duration?
            f. Genre1?
            g. Genre2?
            h. Release?
            i. Metacritic?
            j. Synopsis?
        2. Actor-actresses
            a. Year?
            b. Award?
            c. Winner?

            d. Name?
            e. Film?
    c. Search a record
        i. Pictures database
            1. Complete (exact) search
                a. Search again within these results?
                b. New search (return to menu)?
            2. Partial search
                a. Search again within these results?
                b. New search (return to menu)?
        ii. Actor-actresses database
            1. Complete (exact) search
                a. Search again within these results?
                b. New search (return to menu)?
            2. Partial search
                a. Search again within these results?
                b. New search (return to menu)?
    d. Modify a field
        i. Pictures database
            1. Choose a field
        ii. Actor-actresses database
            1. Choose a field
    e. Sort a field
        i. Pictures database
            1. Choose a field
        ii. Actor-actresses database
            1. Choose a field
    f. Delete a record
        i. Pictures database
        ii. Actor-actresses database

9. Status of Application

Compiled on Xcode and CSE grid.

10.Test Cases may be?

Test Scenario 1: Verify that actors-actresses.csv file contains the correct data in each field.

Test case 1: “Year” field contains the string “Twenty twenty”

Test result 1: Invalid data (this field must only be integers; handle with exception; throw out data)

Test case 2: “Award” field contains a value which is other than “Actor In A Leading Role”, “Actor In A Supporting Role”, “Actress In A Leading Role”, “Actress In A Supporting Role”.
Test result 2: Invalid data (this field is only limited to these four choices; throw out data)

Test case 3: “Winner” field contains a movie name or actor name
Test result 3: Invalid data (this field contains only a 0 or 1; anything else should be thrown out)

Test case 4: “Film” field contains a number like 7.
Test result 4: Invalid data (this field should only be a string with a certain number of characters)

Test case 5: Not test case 1 - 4
Test result 5: Data is valid

Test Scenario 2: Verify that the pictures.csv file contain the correct data in each field.

Test case 1: “Name” field contains non string values like -12312
Test result 1: Invalid data (this field must only be strings)

Test case 2: “Year” field contains a string
Test result 2: Invalid data (this field must only be integers)

Test case 3: “Nomination” field contains a string
Test result 3: Invalid data (this field must only be a float or double)

Test case 4: “Duration” field contains a string value
Test result 4: Invalid data (this field must only be an int)

Test case 5: “genre1” field has an int value
Test result 5: Invalid data (this field must only have string values)

Test case 6: “genre2” field has an int value

Test result 6: Invalid data (this field must only have string values)

Test case 7: “metacritic” field has a string value
Test result 7: Invalid data (this field must only have int or float values)

Test case 8: “release” field has an int value
Test result 8: invalid data (this field must have only string values)

Test case 9: “synopsis” field has an int value
Test result 9: invalid data (this field must only have string values)

Test scenario 3: Verify that a record is added.

Test case 1: Add a record to the pictures database.
Test result 1a: Record does not show up in the csv output. This is a failed test case.
Test result 1b: Record shows up in the csv output. This is a successful test.

Test case 2: Add a record to the actor-actresses database.
Test result 2a: Record does not show up in the csv output. This is a failed test case.
Test result 2b: Record shows up in the csv output. This is a successful test.

Test scenario 4: Verify that a record is found.

<u>Exact searches:</u>

Test case 1: User searches exactly for the actor name “Abigail Breslin” in the actor-actresses database.
Test result 1a: A record is returned with the name “Adolph Caesar”. This is a failed test.
Test result 1b: At least a record or no record (if it doesn’t exist) is returned containing the name “Abigail Breslin”. This is a successful test.

Test case 2: User exactly searches for the film name “Titanic” in the pictures database.
Test result 2a: A record containing the film name “The English” is returned. This is a failed test.
Test result 2b: At least a record or no record (if it doesn’t exist) is returned containing the film name “Titanic”. This is a successful test.

<u>Non existent records:</u>

Test case 3: User searches for a name that is not in the actor-actresses database.

Test result 3a: No records are returned. This is a successful test.
Test result 3b: A record that is not supposed to be returned is returned. This is a failed test.

Test case 4: User searches for a name that is not in the pictures database.
Test result 4a: No records are returned. This is a successful test case.
Test result 4b: A record that is not supposed to be returned is returned. This is a failed test.

Partial searches:

Test case 5: User partially searches for a record with film names containing “House” in the actor-actresses database.
Test result 5a: Invalid results are returned - this is a test failure.
Test result 5b: At least one record or no records (if it doesn’t exist) which contain the substring “house” are returned. This is a successful test.

Test case 6: User partially searches for a record with synopses with the substring “struggle” in the pictures database.
Test result 6a: Completely different records with no sub string of “struggle” are returned. This is a failed test.
Test result 6b: At least one record or no records (if it doesn’t exist) of synopses have the substring “struggle” in it. This is a successful test.

Test scenario 6: Verify that a record is modified.

Test case 1: User changes the year of the film “The Last Command” from 1928 to 1930 in the actor-actresses database.
Test result 1a: After searching for the record again, the year is still 1928 or something completely different. This is a failed test.
Test result 1b: After searching for the record again, the year is not 1930. This is a successful test.

Test case 2: User changes the genre of a film from “Comedy” to “Horror” for a record in the pictures database
Test result 2a: After searching for the record again, the genre is still “Comedy” or something completely different. This is a failed test.
Test result 2b: After searching for the record again, the genre is now “Horror”. This is a successful test.

Test scenario 7: Verify that a field can be sorted by ascending or descending order.

Test case 1: User sorts the “name” field or any other field by ascending order in the actor-actresses database.
Test result 1a: The field is not sorted in ascending order. This is a failed test.
Test result 1b: The field is sorted in ascending order. This is a successful test.

Test case 2: User sorts the “name” field or any other field by descending order in the actor-actresses database.
Test result 2a: The field is not sorted in descending order. This is a failed test.
Test result 2b: The field is sorted in descending order. This is a successful test.

Test case 3: User sorts the “metacritic” field or any other field by ascending order in the pictures database.
Test result 3a: The field is not sorted in ascending order. This is a failed test.
Test result 3b: The field is sorted in ascending order. This is a successful test.

Test case 4: User sorts the “metacritic” field or any other field by descending order in the pictures database.
Test result 4a: The field is not sorted in descending order. This is a failed test.
Test result 4b: The field is sorted in descending order. This is a successful test.

Test scenario 8: Verify that a record is deleted.

Test case 1: Delete a record in the pictures database.
Test result 1a: Record still shows up in the database. This is a failed test case.
Test result 1b: Record does not show up in the database. This is a successful test.

Test case 2: Delete a record to the actor-actresses database.
Test result 2a: Record still shows up in the database. This is a failed test case.
Test result 2b: Record does not show up in the database. This is a successful test.