CMPE 223
Fall 2021
Programming Homework 3

This assignment is due by 23:59 on Friday, 24 December 2021.

You are welcome to ask your HW related questions. You should use only one of these options:

1. Moodle Homework **Question Forum**: HW Question-and-Answer (Q&A) Forum on Moodle is always available. Use the "Forum" link at the course Moodle page.
2. Homework **RECITATION HOURS**: There will be two Q&A RECITATION HOURs on the following days:

- CMPE223-HW3-OfficeHour1: 13 Dec, 06:00-07:00 PM, Zoom ID: 995 8948 4056
- CMPE223-HW3-OfficeHour2: 17 Dec, 06:00-07:00 PM, Zoom ID: 948 8116 5362

Note: Please make sure that you have read the HW document well before participating. However, no HW related questions will be accepted except from the above options.

PROGRAMMING TASK

In this homework, you are supposed to implement a Java program for a movie database by using binary search trees. You must use your own implementation of binary search trees by taking inspiration from your textbook or web. You are not allowed to use any external library or .jar file.

In this movie database, for each movie, you will have an entry in which you keep:

1. The movie title,
2. The day, month, and year the movie was released to theaters,
3. The first name and last name of the director,
4. A list of its cast. For each actor/actress in the cast, you should keep:
   a) The first and last name of an actor/actress, and
   b) The title of the role s/he played in the movie.

In your implementation, you must keep the movie entries in a binary search tree by release year of the movie. Then, to keep the cast of each movie, you should create a binary search tree in which the actors are kept by their names (e.g., if you have five movies in your binary search tree, you will have five more binary search trees: one for the cast of each movie.). To deal with the same year movies, you may define your binary search tree such that for a node, smaller valued items are kept in the left subtree of the node and greater or equal valued items are kept in its right subtree.

Your system will have the following functionalities; the details of these functionalities are given below (see the end of this document for sample inputs/outputs):

1. Add a movie
2. Remove a movie
3. Add an actor/actress to the cast of a movie
4. Remove an actor/actress from the cast of a movie
5. Show the list of movies
6. Show detailed information about a particular movie
7. Query the roles that were played by a particular actor/actress
8. Query the movies which were directed by a particular director

Add a movie: This function adds an entry to the movie database for a given movie whose title, director's name (first name + last name), and release date (day + month + year) are specified as parameters. In this function, the cast is not specified; the actors/actresses that are in the movie will be added later. In this system, the titles of the movies are unique (i.e., no remakes of old movies are considered under the same title). Thus, if the user attempts to add a movie with an existing title, you should overwrite old information.

Remove a movie: This function removes a movie, whose title is specified as a parameter, from the movie database. If this movie does not exist in the database (i.e., if there is no movie with the specified title), you should not allow this operation and give a warning message.

Add an actor/actress: This function adds an actor/actress to the cast of a movie. For that, the title of the movie to which the actor/actress is added, the actor/actress' name (first name + last name), and the title of the role played by the actor/actress are specified as parameters. In this function, you should consider the following issues:

- If the movie with a specified title does not exist in the database, you should not allow the operation and give a warning message.
- In this system, all actor/actress' names are unique within the same cast. Thus, if the user attempts to add an actor/actress with an existing name **within the same cast**, you should overwrite old information. **Here note that an actor/actress can play in different movies.**

Remove an actor/actress: This function removes an actor/actress from the cast of a movie. For that, the title of the movie from which the actor/actress is removed and an actor/actress' name (first name + last name) are specified as parameters. If the movie with a specified title does not exist in the database, you should not allow the operation and give a warning message. Similarly, if the actor/actress is not in the cast of the specified movie, you should not allow the operation and give a warning message. Note that after calling this function, the cast should remain sorted in ascending order.

Show the list of movies: You should list all movie entries in the movie database on the screen in the following format. If the database does not contain any movie, you should display `---none---`. Your system should display movies in **ascending** order by release year.

```
Title, release year, director name (for the 1st movie)
Title, release year, director name (for the 2nd movie)
Title, release year, director name (for the 3rd movie)
. . .
```

Show detailed information about a particular movie: You should display all of the information about a movie whose title is specified as a parameter. The output should be in the following format where the actor/actress' names should be sorted in alphabetically ascending order. If the movie with a specified title does not exist in the database, you should display `---none---` after the title. Your system should display actor/actress names in **ascending** order by actor name.

```
Title
Release day / release month / release year
Director name
Actor/actress name, title of his/her role (for the 1st actor/actress)
Actor/actress name, title of his/her role (for the 2nd actor/actress)
Actor/actress name, title of his/her role (for the 3rd actor/actress)
. . .
```

Query the roles that were played by a particular actor/actress: You should list all roles played by the actor/actress whose name (first name + last name) is specified as a parameter. The output should include the role, movie title, and the year, and should be in the following format. Note that if the actor/actress did not play a role in any movie, you should display `---none---` after the name of the actor/actress. Your system should display movies in **descending** order by release year.

```
Actor/actress name
Role, movie title, release year (for the 1st role)
Role, movie title, release year (for the 2nd role)
Role, movie title, release year (for the 3rd role)
. . .
```

Query the movies which were directed by a particular director: You should list all movies directed by the director whose name (first name + last name) is specified as a parameter. The output should include the titles and the release dates of the movies, and should be in the following format. Note that if the director did not direct any movie, you should display `---none---` after the name of the director. Your system should display movies in **descending** order by release year.

```
Director name
Movie title, release day / release month / release year (for the 1st movie)
Movie title, release day / release month / release year (for the 2nd movie)
Movie title, release day / release month / release year (for the 3rd movie)
. . .
```

The name of the class must be `MovieDatabase`, and must include following public member functions. These functions will be used to test your code. You can also define additional classes in your solution.

```
public class MovieDatabase {
      MovieDatabase();

      public void addMovie(String movieTitle, String directorFirstName,
                           String directorLastName, int releaseDay,
                           int releaseMonth, int releaseYear);
      public void removeMovie(String movieTitle);
      public void addActor(String movieTitle, String actorFirstName,
                           String actorLastName, String actorRole);
      public void removeActor(String movieTitle, String actorFirstName,
                              String actorLastName);
      public void showAllMovies();
      public void showMovie(String movieTitle);
      public void showActorRoles(String actorFirstName,
                                 String actorLastName);
      public void showDirectorMovies(String directorFirstName,
                                     String directorLastName);
};
```

Example Test Code:

Below is an example of the test code that can be used for testing your programs. Of course, we will use other test codes in grading. Thus, do not forget to test your program with different test codes as well.

```
public static void main(String args[])
{
     MovieDatabase md;

     md.showAllMovies();

     md.addMovie("Eyes Wide Shut", "Stanley", "Kubrick", 22, 10, 1999);
     md.addMovie("Family Plot", "Alfred", "Hitchock", 9, 4, 1972);
     md.addMovie("Psycho", "Alfred", "Hitchock", 20, 5, 1960);
     md.addMovie("Sweet and Lowdown", "Woody", "Allen", 26, 1, 1999);
     md.addMovie("Midnight in Paris", "Woody", "Allen", 30, 9, 2011);
     md.addMovie("Barton Fink", "Coen", "Brothers", 21, 8, 1991);
     md.addMovie("The Interpreter", "Sydney", "Pollack", 22, 4, 2005);
     md.addMovie("Psycho", "Alfred", "Hitchock", 20, 5, 1960);

     md.showAllMovies();
     md.removeMovie("Midnight in Paris ");
     md.showAllMovies();
     md.showMovie("Eyes Wide Shut");
     md.addActor("Barton Fink", "John", "Turturro", "Barton Fink");
     md.addActor("Barton Fink", "John", "Goodman", "Charlie Meadows");
     md.addActor("Barton Fink", "Judy", "Davis", "Audrey Taylor");
     md.addActor("Barton Fink", "Michael", "Lerner", "Jack Lipnick");
     md.addActor("Eyes Wide Shut", "Tom", "Cruise", " Bill Harford");
     md.addActor("Eyes Wide Shut", "Nicole", "Kidman", "Alice Harford");
     md.addActor("Eyes Wide Shut", "Madison", "Eginton", "Helena Harford");
     md.addActor("Eyes Wide Shut", "Jackie", "Sawaris", "Roz");
     md.addActor("Eyes Wide Shut", "Sydney", "Pollack", "Victor Ziegler");
     md.addActor("Midnight in Paris", "Woody", "Allen", "Woody Allen");
     md.addActor("The Interpreter", "Nicole", "Kidman", "Silvia Broom");
     md.addActor("The Interpreter ", "Sean", "Penn", "Tobin Keller");
     md.addActor("The Interpreter ", "Earl", "Cameron", "Zuwanie");
     md.showMovie("Barton Fink");
     md.showMovie("Eyes Wide Shut");
     md.removeActor("Eyes Wide Shut", "Jackie", "Sawaris", "Roz");
     md.showMovie("Eyes Wide Shut");
     md.showActorRoles("Nicole", "Kidman");
     md.showActorRoles("Judy", "Davis");
     md.showDirectorMovies("Alfred", "Hitchock");
     md.showDirectorMovies("Stanley", "Kubrick);

     System.exit(0);
}
```

Output of the Example Test Code:

For the test code that is given above, you will see the following output on the screen.

```
---none---

INFO: Movie Eyes Wide Shut has been added
INFO: Movie Family Plot has been added
INFO: Movie Psycho has been added
INFO: Movie Sweet and Lowdown has been added
INFO: Movie Midnight in Paris has been added
INFO: Movie Barton Fink has been added
INFO: Movie The Interpreter has been added
ERROR: Movie Psycho overwritten

Psycho, 1960, Alfred Hitchock
Family Plot, 1972, Alfred Hitchock
Barton Fink, 1991, Coen Brothers
Sweet and Lowdown, 1999, Woody Allen
Eyes Wide Shut, 1999, Stanley Kubrick
The Interpreter, 2005, Sydney Pollack
Midnight in Paris, 2011, Woody Allen

INFO: Movie Midnight in Paris has been removed

Psycho, 1960, Alfred Hitchock
Family Plot, 1972, Alfred Hitchock
Barton Fink, 1991, Coen Brothers
Sweet and Lowdown, 1999, Woody Allen
Eyes Wide Shut, 1999, Stanley Kubrick
The Interpreter, 2005, Sydney Pollack

Eyes Wide Shut
22/10/1999
Stanley Kubrick

--none--

INFO: John Turturro has been added to the movie Barton Fink
INFO: John Goodman has been added to the movie Barton Fink
INFO: Judy Davis has been added to the movie Barton Fink
INFO: Michael Lerner has been added to the movie Barton Fink
INFO: Tom Cruise has been added to the movie Eyes Wide Shut
INFO: Nicole Kidman has been added to the movie Eyes Wide Shut
INFO: Madison Eginton has been added to the movie Eyes Wide Shut
INFO: Jackie Sawaris has been added to the movie Eyes Wide Shut
INFO: Sydney Pollack has been added to the movie Eyes Wide Shut
ERROR: Movie Midnight in Paris does not exist
INFO: Nicole Kidman has been added to the movie The Interpreter
INFO: Sean Penn has been added to the movie The Interpreter
INFO: Earl Cameron has been added to the movie The Interpreter
```

```
Barton Fink
21/8/1991
Coen Brothers
John Goodman, Charlie Meadows
John Turturro, Barton Fink
Judy Davis, Audrey Taylor
Michael Lerner, Jack Lipnick

Eyes Wide Shut
22/10/1999
Stanley Kubrick
Jackie Sawaris, Roz
Madison Eginton, Helena Harford
Nicole Kidman, Alice Harford
Sydney Pollack, Victor Ziegler
Tom Cruise, Bill Harford

INFO: Jackie Sawaris has been removed from the movie Eyes Wide Shut

Eyes Wide Shut
22/10/1999
Stanley Kubrick
Madison Eginton, Helena Harford
Nicole Kidman, Alice Harford
Sydney Pollack, Victor Ziegler
Tom Cruise, Bill Harford

Nicole Kidman
Silvia Broom, The Interpreter, 2005
Alice Harford, Eyes Wide Shut, 1999

Judy Davis
Audrey Taylor, Barton Fink, 1991

Alfred Hitchock
Family Plot, 9/4/1972
Psycho, 20/5/1960

Stanley Kubrick
Eyes Wide Shut, 22/10/1999
```

WHAT TO HAND IN

A zip file for both parts containing:

- The Java sources for your program.

- The Java sources should be WELL DOCUMENTED as comments, as part of your grade will be based on the level of your comments.

- You should test your Java source files on (if) available Moodle VPL environment to ensure your code solution's correctness before submitting. VPL simply tests your program's output by checking against given sample input. You should pass that task's VPL test case successfully.

- A **maximum-3 pages** PDF report document that explains your own answers for programming task in a clearly readable PA report format (refer to **PA REPORT FORMAT** section).

- For given task, only code or report submission will not be graded. In other words, you should submit both correct code solution and its related report for the task in order to be graded.

PA REPORT FORMAT

A programming assignment report is a self-description of a programming assignment and your solution. Please note that if you do not have correct code solution for the task, you should not report about that since it will not be graded. The report must not be hand-written. You may use a word processor or the on-line editor of your choice and prepare as a PDF document. The report must be grammatically correct and use complete English sentences. Each report should include the following sections, in the order given:

Information (%5): This section includes your ID, name, section, assignment number information properly.

Problem Statement and Code Design (%30): Include a brief summary of the problem and/or your sub-tasks to be completed in this assignment. You should show your modular design rationale by creating a *structure chart* that indicates your top-down, stepwise refinement of the problem solution. You may create the structure chart using available graphical tools like MS PowerPoint, SmartDraw etc.

Implementation and Functionality (%40): Since you have modular source code, you should describe each sub-module (program) in this section. Each sub-module should include names and types of any input/output parameters as well as the *pseudocode* algorithm that used for completing its task. By this way, you give meaning to each chart boxes from the previous section.

Testing (%15): You should provide a tester class that is able to identify key test points of your program. This class should be able to generate additional (apart from the given sample input/output) test data for the purpose of being clear on what aspects of the solution are being tested with each set. This section should also include a description of any program *bugs* that is, tests which has incorrect results. You should write these to describe your tests, summarize your results, and argue that they cover all types of program behavior.

Final Assessments (%10): In this final section, you should briefly answer the following questions:

- What were the trouble points in completing this assignment?
- Which parts were the most challenging for you?
- What did you like about the assignment? What did you learn from it?

IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:59 on Friday, December 24th.

2. You should upload your homework to Moodle before the deadline. No hardcopy submission is needed. You should upload files and any additional files if you wrote additional classes in your solution as a single archive file (e.g., zip, rar).

3. The standard rules about late homework submissions apply (20 points will be deducted for each late day). Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You ARE NOT ALLOWED to modify the given method names. However, if necessary, you may define additional data members and member functions.

5. Your classes' name MUST BE as shown in the homework description.

6. The submissions that do not obey these rules will not be graded.

7. To increase the efficiency of the grading process as well as the readability of your code, you have to follow the following instructions about the format and general layout of your program.

8. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//---------------------------------------------------------
// Title: Scheduler tester class
// Author: Name/Surname
// ID: 2100000000
// Section: 1
// Assignment: 1
// Description: This class tests the …
//-----------------------------------------------------
```

9. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to your classes, functions, declarations etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
 void setVariable(char varName, int varValue)
//---------------------------------------------------------
// Summary: Assigns a value to the variable whose
// name is given.
// Precondition: varName is a char and varValue is an
// integer
// Postcondition: The value of the variable is set.
//---------------------------------------------------------
{
      // Body of the function
}
```

10. Indentation, indentation, indentation...

11. This homework will be graded by your TA, İbrahim İleri. Thus, you may ask him your homework related questions through HW forum on Moodle course page. You are also welcome to ask your course instructor İsmail Bora Çelikkale for help.