**CSCI 221: Computer Programming II**

**Programming Assignment 3 and 4 - see notes below detailing what needs to be completed for each assignment**

**Programming Assignment Overview: Let’s Play Ball!**

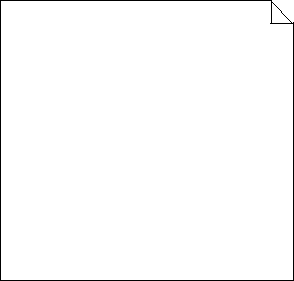
In this assignment, you will write a program to read in the salary data from baseball players, store the data in an array of objects, and provide several methods to query the data. The data is provided to you in a comma-delimited text file which is part of a bigger database. We will only use the Salaries.csv file provided. You need to implement the following UML:

- Salaries: PlayerSalary[ ]

PlayerSalaries

*The Constructor needs to read in the comma-delimited data file, create an array of PlayerSalary objects, and store the data in it.*

*comparePlayersInYear returns a PlayerSalary array of size two in which the highest paid player is*



+PlayerSalaries(String filename)

+averageSalaryInYear(int year): int

+findPlayerInYear(String player, int year): PlayerSalary

+highestSalaryInYear(int year): PlayerSalary

+lowestSalaryInYear(int year): PlayerSalary

+highestSalaryInTeamInYear(String team, int year): PlayerSalary

+lowestSalaryInTeamInYear(String team, int year): PlayerSalary

+highestSalaryInLeagueInYear(String leag, int year): PlayerSalary

*in position 0 of the array.*

*The other methods should be intuitive.*

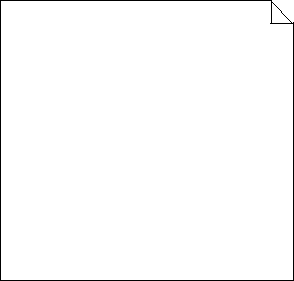
+lowestSalaryInLeagueInYear(String leag, int year): PlayerSalary PlayerSalary

* yearID: int
* teamID: String
* leagueID: String

|  |  |  |
| --- | --- | --- |
|  | |  |
| SalaryLookUp | | |
|  | | |
| +main(String args[ ]): static void | | |
|  |  | |

* playerID: String
* salary: int

+toString, getter methods for each instance variable



*The main method reads in the data and stores it as described above, then provides a text-based user interface via the console.*

*Present a menu of numbered options (reflecting the methods in PlayerSalaries) to the user (see menu expected next page). Whichever option the user chooses, the program then asks for year, player, and/or team info as needed. Print the returned info using toString in the PlayerSalary object when needed. If any method is asked for data that is not known (example a year that is incorrect or out of range, or a team or player ID that does not exist) that method should throw an exception with an appropriate message). The main method should catch and handle that exception and the menu loop should continue until the user chooses to quit the program. If the user chooses a menu number out of range, the program should print an appropriate message and the menu loop should continue. The program terminates when the user chooses the Quit option. You MAY assume that the data entered are of the appropriate type – example, when an int is expected, an int is entered.*

Please enter your choice:

1: See average salary in a specific year

2: See a specific player’s salary in a specific year 3: See the highest salary in a specific year

4: See the lowest salary in a specific year

5: See the highest salary on a specific team in a specific year 6: See the lowest salary on a specific team in a specific year 7: See the highest salary in the league in a specific year

8: See the lowest salary in league in a specific year

9: Compare salaries of two specific players in a specific year 10: Quit.

------------

**Guidance for completing this assignment.**

Homework 3 – Complete parts 1 - 3

Part 1:

Complete the PlayerSalary class. The constructor is written, you must complete the toString and getter methods, and a main method that verifies that the PlayerSalary class works properly.

Part 2: Complete the PlayerSalaries constructor by reading the csv file provided, and storing the data for each player into a PlayerSalary instance and storing that instance into the PlayerSalaries instance variable (PlayerSalary[ ]) in the PlayerSalaries class.

Part 3: Implement all of the PlayerSalaries methods, without worries about throwing exceptions.

Homework 4 – Complete parts 4 – 5

Part 4: Add the exception throwing to each PlayerSalaries method.

Part 5: Complete the SalaryLookUp class by writing the main method to generate the menu, prompt for input, and catch exceptions as necessary which may be returned by the PlayerSalaries methods.

**Follow the same folder-building process required of HW1. Submit this zipped folder. Documentation – follow the usual practices for documenting Java programs.**