CMSC 203

Assignment 6

Concepts tested by this program:

Array lists of objects

Arrays of components

Output formatting

Reading from files

JUnit Tests

Inheritance

## The College Football Playoff (CFP) is the new system in American [college football](http://en.wikipedia.org/wiki/College_football) that has determined a national champion for the [NCAA Division I Football Bowl Subdivision](http://en.wikipedia.org/wiki/NCAA_Division_I_Football_Bowl_Subdivision) (FBS) starting in the 2014 season. It replaced the Bowl Championship Series (BCS) previously used that generated much controversy.[[1]](#footnote-1) Unlike the BCS, the new format does not use computer rankings or polls to select the participants. In the playoff four teams play in two semifinal games, with the winners advancing to the new [College Football Championship Game](http://en.wikipedia.org/wiki/College_Football_Championship_Game). Six [bowl games](http://en.wikipedia.org/wiki/List_of_college_bowl_games) — the [Rose Bowl](http://en.wikipedia.org/wiki/Rose_Bowl_Game), [Sugar Bowl](http://en.wikipedia.org/wiki/Sugar_Bowl), [Orange Bowl](http://en.wikipedia.org/wiki/Orange_Bowl), [Cotton Bowl](http://en.wikipedia.org/wiki/Cotton_Bowl_Classic), [Fiesta Bowl](http://en.wikipedia.org/wiki/Fiesta_Bowl), and [Peach Bowl](http://en.wikipedia.org/wiki/Peach_Bowl) — rotate as hosts for the semifinal games. A committee of 12 members including athletic directors, former coaches, administrators and others vote to select and seed the teams.[[2]](#footnote-2) See [www.collegefootballplayoff.com](http://www.collegefootballplayoff.com) for more detailed information.

## You are to write an application to read the committee votes and select and seed the teams. For the sake of simplicity, the input file will include votes on only 10 teams, with one line per committee member. Since the voting is by secret ballot, the votes of each committee member will be anonymous.

## Once the votes have been read, the application will show the rankings of the ten teams, along with their total votes (lower is better). The application will be able to show the detailed votes of each anonymous committee member for each team.

Since the CFP intends to provide weekly rankings as the season progresses, the user will be able to select other voting files and display the results.

## Operation

* When the user selects the Read Votes button, the application reads the committee voting file.
* When the user selects the Rank Teams button, the application selects and seeds the teams and displays the output. If the user selects the Rank Teams button again, a new file will be read in and results displayed.
* When the user selects the Show Details button, the application will display the votes of each committee member for each team, listed in ranked order.
* If the user selects Read Votes again, another file can be read in and repeat the displays
* When the user closes the frame or presses the Exit button, the application exits.

## Specifications

Data Element – CollegeFootballTeam class

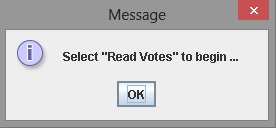
Create a CollegeFootballTeam class that inherits from the Team class (Team.java is supplied). Team.java is a high-level class that would accommodate any team, regardless of sport, professional or amateur. CollegeFootballTeam will also contain instance variables for the team’s conference, the votes for that team, and the sum of votes, and inherit instance variables for the team name, sport, and professional (true/false). It will include a getSumOfVotes method to retrieve the sum of the votes and a getVotes method to retrieve an array of votes, and a setter and a getter for the Conference name. Provide a default no-argument constructor and a three-argument constructor whose arguments are the String name of the team, the String name of the Conference, and an array of integers that will hold the votes for the team.

Data Structure – An ArrayList< CollegeFootballTeam > of CollegeFootballTeam objects.

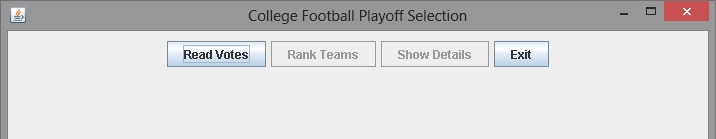
Data Manager – CollegeFootballPlayoffManager class

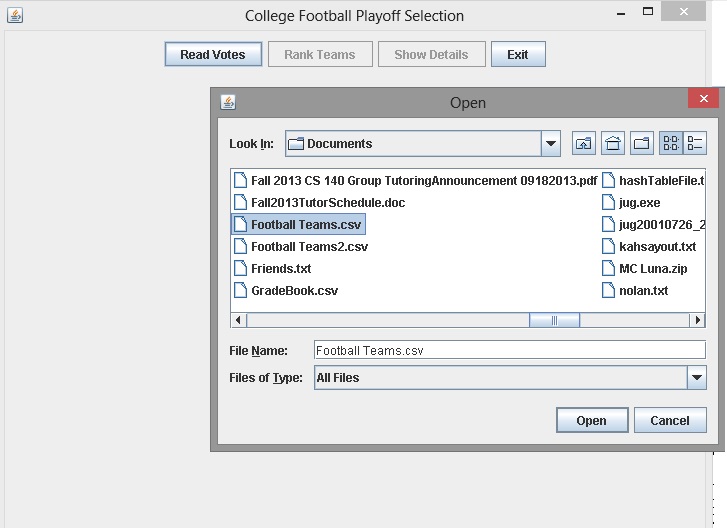
Create a CollegeFootballPlayoffManager class. It will contain an ArrayList< CollegeFootballTeam > of CollegeFootballTeam objects. Data Managers are in charge of adding to the data structure, deleting from the data structure, updating the data structure, retrieving information from the data structure and any other methods that involve the data structure as a whole. This Data Manager will provide methods to read the input file (read Votes) and return an ArrayList< CollegeFootballTeam > of teams in order of their voted ranking (rankTeams). **Follow the Javadoc that I gave you for your CollegeFootballPlayoffManager class so that the JUnit test for this class will work correctly**. Add any additional methods as required for your design.

GUI Driver

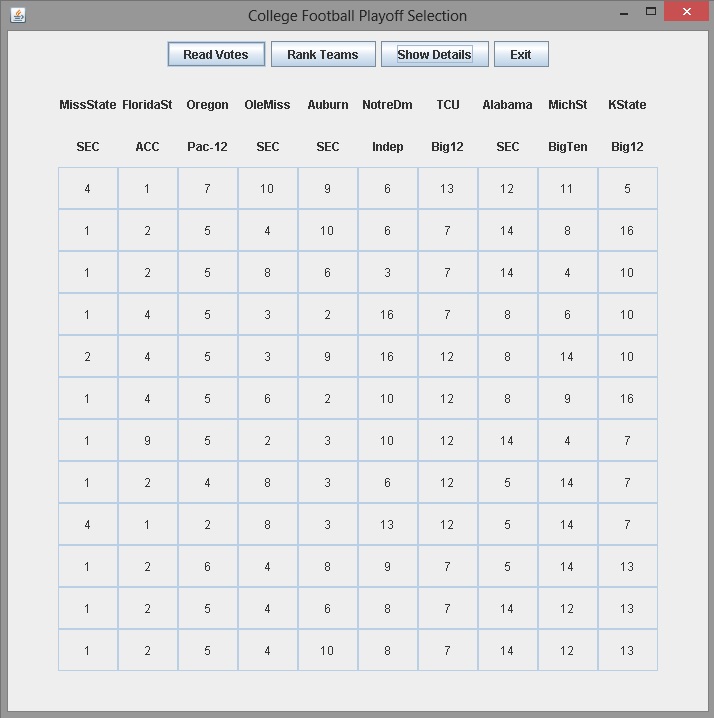
* It will contain a CollegeFootballPlayoffManager object.
* Use a JOptionPane at the beginning of the application to prompt the user to choose a file to read. 

When the application starts, only the relevant buttons are activated.

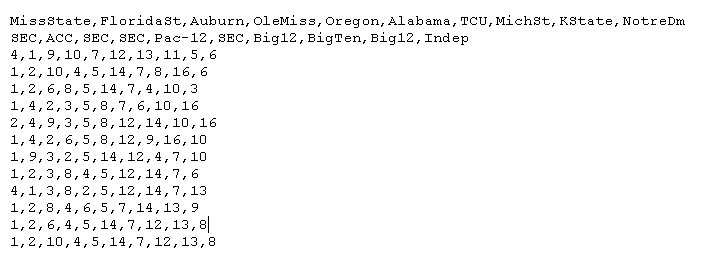


Use the JFileChooser class to select a file to read. 

* When the user selects the Rank Teams button, display the teams by name in order (#1 through #10).
* When the user selects the Show Details button, display the 10 teams by name and the votes of each (anonymous) committee member.



Data File – The data files will be text files with two header lines and one line per committee member. The first line will contain the team names, and the second line will contain the corresponding conference names. Each subsequent line will be a sequence of 1 through 10, separated by commas. There will be several data files, labeled by week (the actual committee will vote seven times during the season, starting on Oct 28th.)



Deliverables:

CheckList

Java files - The src folder with your driver (javafx application), data element, data manager and Junit Test (.java) files

Javadoc files - The doc folder with your javadoc for student generated files

UML Class Diagram (an image, not the proprietary format, must be a .jpg)

Deliverable format: The above deliverables will be packaged as follows. Two compressed files in the following formats:

LastNameFirstName\_AssignmentX\_Complete.zip [a compressed file containing the following]

UML.jpg

Assignment 6 Checklist (filled in with YES or NO or ?)

doc [a directory] *please include the entire doc folder with the javadoc for student*

*generated files*

file1.html (example)

file2.html (example)

src [a directory] *contains your driver (javafx application), enumerated class, data*

*element, data manager and Junit Test (.java) files*

File1.java (example)

File2.java (example)

File\_Test.java (example)

LastNameFirstName\_AssignmentX\_Moss.zip [a compressed file containing only the following]

*contains .java file which includes the driver (javafx application), enumerated*

*class, data element, data manager and Junit Test (.java) files – NO FOLDERS!!*

File1.java (example)

File2.java (example)

**Program Grade Sheet Assignment #6**

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DOCUMENTATION Total 25 points

CheckList for Assignment 6 is included and completed 1 pt \_\_\_\_\_

Javadoc for all user created classes 6 pts \_\_\_\_\_

Test Cases: JUnit Test Class

Add to the public tests for your CollegeFootballPlayoffManager class 5 pts \_\_\_\_\_

Create a Test for your Team class 5 pts \_\_\_\_\_

UML Diagram 4 pt \_\_\_\_\_

Lessons Learned 4 pt \_\_\_\_\_

In 3+ paragraphs, highlight your lessons learned and learning experience from working on this project. How did you do? What have you learned? What did you struggle with? How will you approach your next project differently?

PROGRAMMING Total 75 points

Internal class documentation (within source code) using Javadoc 5 pts \_\_\_\_\_

Description of what class does

Author’s Name, @author

Methods commented properly using Javadoc

Description

@param, @return

Compiles and Runs without runtime errors or warnings 10 pts \_\_\_\_\_

Accuracy

Public tests – JUnit test given to you and your tests/additions 10 pts \_\_\_\_\_

Student Tests – CollegeFootballPlayoffManager and Team 4 pts \_\_\_\_\_

Private tests 10 pts \_\_\_\_\_

Program Details

Data Element – Team class 10 pts \_\_\_\_\_

1. Contains instance variables with setters and getters
2. method to retrieve the total votes for the team
3. toString method to print the name and total votes of the team

Data Structure – uses ArrayList<Team> to hold the list of team instances 4 pts \_\_\_\_\_

Data Manager – CollegeFootballPlayoffManager 12 pts \_\_\_\_\_

1. arraylist of Team objects
2. method to add to the data structure (addTeam)
3. method to retrieve from the data structure (getTeams).
4. constructor and a toString method
5. Follow the Javadoc for the CollegeFootballPlayoffManager class

GUI – Frame and Panel 10 pts \_\_\_\_\_

1. Displays an appropriate title
2. Use the ActionListener and define actionPerformed
3. Uses a CollegeFootballPlayoffManager object
4. Reads from a file - uses a FileChooser
5. Displays top four teams and shows their pairing
6. Displays teams and total of votes for each team
7. Provides an Exit button that closes the application
8. Program user interface

Clear to user how data is to be entered

Output is easy to understand

Total 100 pts \_\_\_\_\_

1. <http://en.wikipedia.org/wiki/Bowl_Championship_Series> [↑](#footnote-ref-1)
2. <http://en.wikipedia.org/wiki/College_Football_Playoff> [↑](#footnote-ref-2)