Football World Cup

**Assignment Specification**

For this assignment you will simulate the playing of a Football World Cup. This is a simplification of the actual FIFA World Cup but the simulation you will develop requires **no prior knowledge** of the rules of football. This section specifies the required functionality of the program.

**Only a text interface is to be used for this program**, there is to be no GUI. More marks will begained for a game that is easy to follow with clear information/error messages.

**System description**

The Football World Cup is played between 4 teams. The teams are ranked from 1 to 4, with 1 indicating the highest ranking and 4 the lowest.

The Cup is played in two stages: Preliminary and Final.

During the Preliminary stage, each team plays the other three teams resulting in a total of six (6) games played. The games during the Preliminary stage occur sequentially and the result of each game is shown on the screen as it is completed. In the Preliminary stage a team gains 3 points for a win, 1 point for a draw, and 0 points for a loss.

At the end of the Preliminary stage the system will display a summary of the results with the teams sorted from first to last, based on the following criteria:

* The teams will be listed from most to least points.
* If teams have the same points, then the team with the higher number of goals scored will be placed higher on the table.
* If teams cannot be separated by any of the previous criteria, then the higher placing in the table will be determined randomly.

After the Preliminary stage is completed, the top two teams play in the Final. The only difference between the final and the preliminary games is that the final must have a winner, the Football World Cup Champions!

At the end of the Final the overall results are displayed on the screen, including the name of the Football World Cup champion team, the Golden Boot Award player, and the Fair Play Award team.

**Class Design**

Your design should have at least the following three classes: Player. Team and Game. A better solution will have more classes, including for example, Menu and RandomGoalsGenerator. For each class you should have a default constructor and a constructor that accepts values for the fields as parameters.

Each class should also have *appropriate* accessor and mutator methods and be able to return its state in the form of a String. Validation of values for fields should also be implemented. You should not allow an object of a class to be set to an invalid state.

**Player**

An object of Player will have at least the following fields:

*name* – of type**String**. The name can contain only alphabetical characters and at most onehyphen, ‘-‘, to accommodate names that may be hyphenated such as Zeta-Jones. There must be a minimum of two alphabetic characters in the name, it should be of a sensible maximum length, and it cannot begin or end with a hyphen. The two players in a team cannot have the same name.

*goals* – of type**int**representing the number of goals scored. This will be used to determine the‘Golden Boot’ award for the top goal scorer.

**Team**

An object of Team will have at least the following fields:

*name* – of type**String**and is the name of the country the team is representing.

*ranking* – of type**int**. This field refers to the team’s relative strength compared to the other teamsin the Cup. No two teams can have the same ranking. The ranking must be between 1 and 4 inclusive (as there will only be four teams in the competition).

*2 players –* of type**Player**. Each team will have two players who are the goal scorers.

Each team will also have a yellow card and red card score. A yellow card is shown to players who have committed serious fouls, while a red card is shown to players who have committed more severe offences and they are sent from the field. The cards are associated with a team, not an individual player. The showing of yellow and red cards is determined randomly, both are rare and some games have no cards shown at all. But there are usually four times as many yellow cards shown as red cards. The total yellow and red card marks will determine a team’s Fair Play score, which will be used to determine the Cup’s Fair Play Award. Each yellow card is worth one mark, each red card is worth two marks. The lower the total marks, the fairer the team.

In addition to the fields mentioned above, an object of class Team must be able to store and/or return other information about its performance in the Cup. You must decide on which of the following should therefore be fields and which do not have to be fields: the number of games the team has played in the Cup, the number of games won, lost, drawn, number of goals scored, and overall points.

**Game**

The Game class will have at least one field, an ArrayList of Team objects. The Game class also has *at least* three methods: *playGame(), playPenaltyShootOut(),* and *displayGameResult().*

The *playGame()* method simulates the playing of a game between two teams. This is done by randomly generating the number of goals scored by each team. The number of goals generated should be in a specified range. The team with the highest ranking will have a greater chance of winning. This will be simulated by giving the higher ranked team a wider range of possible goals as follows:

* Higher ranked team: a goal range of 0 to (5 + a random upset (a random number between 0 and 2))
* Lower ranked team: a goal range of 0 to ((5 – difference in team rank) + a random

upset (a random number between 0 and 2))

The goals will be randomly distributed between the two players.

The *playPenaltyShootOut()* method simulates the playing of a penalty shoot-out, if required for a Final that ends in a draw. One player of each team has five shots at goal. The team whose player has the highest number of goals at the end of the five shots wins. If the score is equal then each player has another shot at goal. This continues until there is a result. The goals scored by a player in a penalty shoot-out are not counted towards the Golden Boot Award.

The *displayGameResult()* method displays on the screen the result of the game at the end of the game, for example:

Game result: Spain 4 vs. Australia 0

Cards awarded: Australia - 1 red card.

The team records (played, won, etc.) are updated after a game is completed.

**System Interface**

When the program starts, the system reads the details of each team from file called ‘teams.txt’. The details include each team’s name and ranking. Once the information is loaded from the file there is no more reading or writing to the file during the actual *running* of the program.

The program then prompts the user for the names of two players in each team who will be the goal scorers. If the name entered is invalid then the user is prompted to re-enter. If the name re-entered is invalid then a default value is allocated, e.g. player-1-Ghana

The system will then offer the user the following menu options and you must display a text based menu that uses the Scanner class to obtain input from the keyboard:

1. Play Preliminary Stage B. Play Final

C. Display Teams D. Display Players E. Display Cup Result X. Close

If the user chooses option ‘A’ the Preliminary stage will be played.

If the user chooses option ‘B’ the Final will be played between the top teams from the preliminary stage, as previously described. If option ‘B’ is chosen but the Preliminary stage has not been played an error message will be displayed. If the final ends in a draw then a penalty shoot-out is played.

If the user chooses option ‘C’ the record of each team is displayed. For example:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Played | Won | Lost | Drawn | Goals | Points | Fair Play Score |
| Australia | 6 | 4 | 2 | 0 | 14 | 12 | 3 |
| China | 6 | 3 | 2 | 1 | 10 | 10 | 0 |
| Ghana | 6 | 3 | 2 | 1 | 9 | 10 | 0 |
| Spain | 6 | 2 | 4 | 0 | 6 | 6 | 4 |

If the user chooses option ‘D’ the players are listed with their number of goals scored. The list does not need to be sorted in any order. For example:

Cahill (Australia) - 8

Rogic (Australia) - 6

Gao (China) - 7

Yu (China) - 3

………..

……….

If the user chooses option ‘E’ the name of the Football World Cup champion team, the Golden Boot Award player, and the Fair Play Award team will be displayed to screen. For example:

Football World Cup Winner: Australia

Golden Boot Award: Ronaldo from Spain

Fair Play Award: Ghana and China

Note that there can be multiple players that win the Golden Boot Award and there can be multiple teams that win the Fair Play Award.

After menu item A to E, the menu is re-displayed on the screen.

If the user chooses option ‘X’, the system should write the same information shown when option ‘E’ is chosen to a file called ‘statistics.txt’. The system will then close.

**Important Notes**

1. Consider carefully your choice of classes and how they interact. You must use appropriate data structures to store the various objects (player, team, etc.) in the program. You must document any additional assumptions you made.
2. You will be required to justify your design and the choice of any data structures used at the interview.
3. Validation of values for fields and local variables should be implemented where appropriate. You should not allow an object of a class to be set to an invalid state (i.e. put some simple validations in your mutator methods).
4. Your program should handle incorrect or invalid input and present the user with relevant error messages. No invalid input should crash the program.
5. Exception handling should be used where appropriate.

**[the end]**