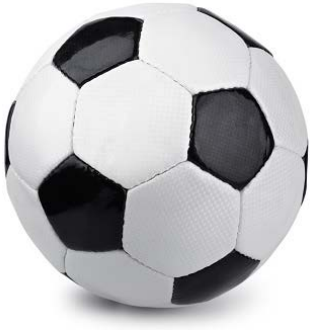
Java Foundations
Practices - Section 8: The Soccer League

Overview



It's been a brutally cold and snowy winter. None of your friends have wanted to play soccer. But now that spring has arrived, another season of the league can begin. Your challenge is to write a program that models a soccer league and keeps track of the season's statistics.

There are 4 teams in the league. Matchups are determined at random. 2 games are played every Tuesday, which allows every team to participate weekly. There is no set number of games per season. The season continues until winter arrives.

The league is very temperature-sensitive. Defenses are sluggish on hot days. Hotter days allow for the possibility of more goals during a game. If the temperature is freezing, no games are played that week. If there are 3 consecutive weeks of freezing temperatures, then winter has arrived and the season is over.

Tasks

Write a program that models a soccer league and keeps track of the season's statistics. Carefully consider what data should be stored in an array and what data should be stored in an `ArrayList`. Design classes with fields and methods based on the description of the league. You'll also need a test class that contains a main method. All fields must be `private`. Provide any necessary getters and setters.

Teams

Each team has a name. The program should also keep track of each team's win-total, loss-total, tie-total, total goals scored, and total goals allowed. Create an array of teams that the scheduler will manage.

Print each team's statistics when the season ends.

Games

In a game, it's important to note each team's name, each team's score, and the temperature that day. Number each game with integer ID number. This number increases as each game is played. Keep track of every game played this season. This class stores an `ArrayList` of all games as a field.

Your program should determine scores at random. The maximum number of goals any one team can score should increase proportionally with the temperature. But make sure these numbers are somewhat reasonable.

When the season ends, print the statistics of each game. Print the hottest temperature and average temperature for the season.

Scheduler

Accept user input through a `JOptionPane` or `Scanner`. While the application is running, ask the user to input a temperature. The program should not crash because of user input. If it's warm enough to play, schedule 2 games. Opponents are chosen at random. Make sure teams aren't scheduled to play against themselves. If there are 3 consecutive weeks of freezing temperatures, the season is over.

Sample Output:

```
run:
Too cold to play.
Too cold to play.
Too cold to play.
Season is over


*********RESULTS*********



Team 1
Wins: 1, Losses: 1, Ties:0
Points Scored: 9, Points Allowed: 9

Team 2
Wins: 1, Losses: 1, Ties:0
Points Scored: 8, Points Allowed: 8

Team 3
Wins: 0, Losses: 1, Ties:1
Points Scored: 6, Points Allowed: 9

Team 4
Wins: 1, Losses: 0, Ties:1
Points Scored: 8, Points Allowed: 5

Game #1
Temperature: 90
Away Team: Team 2, 4
Home Team: Team 4, 7

Game #2
Temperature: 90
Away Team: Team 1, 8
Home Team: Team 3, 5

Game #3
Temperature: 35
Away Team: Team 1, 1
Home Team: Team 2, 4

Game #4
Temperature: 35
Away Team: Team 3, 1
Home Team: Team 4, 1

Hottest Temp: 90
Average Temp:62.5
```

Hint
It's possible that your program may get stuck in an infinite loop during this exercise. This isn't uncommon during software development. The final program should have a way of terminating itself. But until you've implemented that feature, your IDE provides a **Stop** button.