Assignment 2: Soccer Ants, part II

Due: The fifth week of class, with due date and time as per the course information sheet.

Hints, announcements and starter code: On the course Web site.

Purpose

To make you work with to ADTs, interfaces, assertions, exceptions and Javadoc.

Overview

Our program models a team of soccer-playing ants, those lovable creatures from Assignment 1. We’ve put aside the genetic algorithm-related parts from that assignment, so that you can concentrate on developing a Java class for our team.

Once again, please note that you don’t need any prior knowledge of ants or soccer for this assignment. We’ll tell you all you need to know.

A soccer team consists of 11 players (we do not care about substitutes): one goalkeeper and some combination of defenders, midfielders and strikers, arranged in a formation. The formation is represented as a triple of limits: the first for the number of defenders, the second for the number of midfielders, the third for the number of strikers. For example, a 4-4-2 formation consists of at most four defenders, at most four midfielders and at most two strikers, while a 3-4-3 formation consists of at most three defenders, at most four midfielders and at most three strikers. Note that the numbers in the triple must add up to 10.

We provide a more formal description of our team through the SoccerAntTeam ADT:

Data:

- A set of SoccerAnts. There can be at most 11 SoccerAnts in this set at any one time.

- A team formation, represented as a triple of limits: the first for the number of defenders, the second for the number of midfielders, the third for the number of strikers. These numbers must add up to 10.

Operations:

- **add(a)**: Adds SoccerAnt a to the team. The addition must be consistent with the team’s formation. For example, if the team has a 5-3-2 formation and it already has already 5 defenders, another defender cannot be added.

- **removeWorst(p)**: Removes the worst player at a particular position p, which could be one of goalkeeper, defender, midfielder or striker. Note that if the position is goalkeeper, then the goalkeeper is removed. Furthermore, there must be at least one player at position p for this method to work properly.
• **setFormation(f):** Set the team’s formation to f. As a result, the team’s set of ants must become consistent with the new formation. For example, if the team’s formation changes from 4-4-2 to 4-2-4, then the team must have only 2 midfielders after the change (so the 2 worst midfielders should be removed if there are originally 4 midfielders, or the worst midfielder if there are originally 3 midfielders). Empty spots do not have to be filled and should not be; for example, if the team’s formation changes from 4-4-2 to 4-2-4, extra strikers should not be added so that there are 4 of them.

• **getPlayers():** Returns the set of all SoccerAnts on the team; this set does not have to be sorted.

**Your Task: Implement SoccerAntTeam**

The SoccerAntTeam ADT described above is represented in the Java interface SoccerAntTeam, which is provided to you along with some other classes you will need for this assignment. Your task is to implement the SoccerAntTeam interface in a Java class, which you must call MySoccerAntTeam. The implementation of the class is totally up to you; read the Web page for this assignment for hints on possible implementations.

Make sure to comment your code well; this includes providing a representation invariant for your class and detailed method specifications that must be written using Javadoc (see the Javadoc section in the Course Handbook for more details).

**What to Submit**

All your .java files (including MySoccerAntTeam.java, which contains the class MySoccerAntTeam, your implementation of the SoccerAntTeam interface) MUST be electronically submitted to your A2 subdirectory or you get a zero. Read the handout Rules for Submission carefully before submitting. You should thoroughly test your code, but you don’t have to hand in any testing. No paper submission is necessary.

**Hints**

Start by reading all of the starter code carefully.

Before you start writing your class, do the following exercise: Add detailed method specifications to each of the methods in the SoccerAntTeam interface. Not only can you reuse the comments directly inside your class, but this will give you a much better understanding of what you have to do.