Seminar 2: Relations among classes
24292-Object Oriented Programming

1 Introduction

The objective of this seminar is to learn how to design an application consisting of a number of classes, using the different types of relations that we saw in class. Each class has a number of attributes and methods that are to be set to complete the design. Remember that each completed design diagram is a connected graph.

The aim of this seminar is to continue the design of an application that manages football leagues. More specifically the task is to define a set of classes that can represent different types of football competitions and football teams in different countries. The idea is to exploit the types of relations seen in the theory classes. A partial solution to these exercises will be implemented in Java during Laboratory session 2.

2 Review

We will use the relations seen in class, and we will pay special importance to the fact that **some relations induce attributes** in the given classes. Remember also to include the **cardinality** of some relations. Give concrete examples, including the design and the cardinality (when necessary) of the relations seen in class: **Inheritance**, **Association**, **Composition** (strong), **Aggregation** (weak) and **Use/Dependency**.

3 Description of the football application

The football application should be able to represent and simulate football leagues. Each league has a name (e.g. "La Liga", "Liga F", etc.) Each league involves a set of football teams, each of which consists of a set of players. Teams and players are defined as in Seminar 1. Each league also has an associated country, and the participating teams have to be from the same country. Each league also has an associated gender (male, female, or mixed), and the participating teams have to have the correct gender designation. It should be possible to add teams to a league according to these rules.

Each league involves a number of matches. Each match is played between exactly two teams, and the application should record the score (goals scored of each team) and the goal scorers (list of players of each team who scored the goals). In a league, each team plays each other team twice (once home and once away). After adding all the teams to a given league, the application should be able to generate the associated matches, and then simulate the matches.

After a match is simulated, the statistics of the participating teams and players should be updated. It should be possible to print the result of all simulated matches, the league table, and a list of the top k goal scorers.

4 Design

Given the specifications in the previous sections, the seminar consists in defining all the classes and relations of the football application. For each class you will have to indicate attributes and methods. You will also have to indicate the cardinality and concrete type of relation.

To be successful, the design needs to include mechanisms for interaction between classes, since typically the information needed to perform an operation is stored in instances of different classes. A good design should attempt to conform to the following standards:

1. A large amount of decoupling, i.e. try to include as few relationships between classes as possible without breaking the necessary interaction.

2. A class should not call methods of other classes than those it is directly related to, which means that multiple methods might be needed in case information has to propagate through several classes.