Inés Vergés (u213900)
Nil Tomàs (u214575)

INFORME LAB02

Introducció

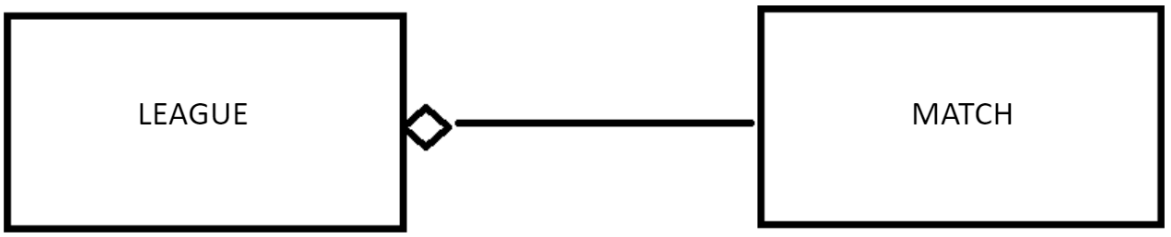
En aquesta segona pràctica de l'assignatura, se'ns ha demanat ampliar el programa creat al LAB01, que recordem, consistia en crear 3 classes: "Country", "Player" i "Team" i definir uns certs mètodes que ens permetien crear equips i afegir-hi jugadors. Així doncs, en aquest LAB02, hem creat noves classes amb l'objectiu de crear una lliga amb els seus equips i simular els partits dels equips que la conformen, reutilitzant les classes i mètodes creats al LAB01. Per això, hem creat 3 noves classes:

- La primera és "League" i conté tota la informació de la lliga: nom, gèneres, país, equips que la conformen i els partits jugats. Els mètodes emprats en aquest classe són:
    - El constructor, on s'inicien les variables de nom, país i gènere referent a la lliga i les llistes dels equips i partits.
    - Els getters: "getName()", "getCountry()" i "getGender()" que retornen el nom, el país i el gènere respectivament de la lliga.
    - "addTeam(...)" on afegim un equip a la lliga.
    - "generateMatches(...)" i "simulatematches(...)" on generem les combinacions de partits de cada equip i els simulem i "printmatches()" on imprimim els partits.
- La segona classe és "Match". Conté com atributs dos equips, el local i el visitant, a més dels seus respectius gols. També té dues llistes (LikedLists) dels golejadors locals i visitants que han anotat gol. Els mètodes emprats en aquesta classe són:
    - El constructor on s'inicien les dues llistes, els dos equips que conformen el partit amb el seu número de gols (escollit de manera aleatòria) i el número de jugadors que té cada equip.
    - Els getters: gethomeTeam() i getawayTeam(), que retorna l'equip local i visitant respectivament, gethomeGoals() i getawayGoals(), que retornen respectivament el número de gols de l'equip local i visitant.
    - simulateMatch(), on es determina quins jugadors han marcat en cas d'haver-hi algun i printMatch(), que imprimeix totes les estadístiques del partit, com els golejadors de cada equip i el resultat final del partit.
- La tercera i última classe és "FootballApplication" que fa la funció de "Main". En aquesta iniciem els diferents països (5 en total), la lliga, els equips (3 en total) i els jugadors (15 en total). Tot seguit afegim els jugadors als respectius equips (5 per a cada equip), i generem i simulem els partits de cada equip. Per cada equip imprimirem el número de victòries, derrotes i empats, els gols a favor i en contra. Finalment, per cada jugador també imprimirem les seves estadístiques.

Inés Vergés (u213900)
Nil Tomàs (u214575)

Descripció de les solucions emprades

A l'hora d'elaborar el codi en conjunt, s'ha seguit el següent ordre: Primerament s'ha creat la classe "Match" i s'han definit els seus mètodes i atributs, després la classe "League" també amb els seus respectius mètodes i atributs. Hem cregut que aquest ordre era l'òptim ja que per a desenvolupar els mètodes de "League", ens feien falta els de "Match". En altres paraules, la classe Match comparteix una relació d'agregació amb la classe League.



Per generar el mètode simulateMatch() s'ha hagut de fer dos fors, un que assigni els gols de l'equip visitant als seus jugadors i un que assigni els gols de l'equip local. En comptes de fer dos fors, primer es contemplava fer un únic for, amb els jugadors de l'equip local i de l'equip visitant que havien marcat dins el mateix for. No obstant, ens vam adonar que, tal i com estava estructurat no era possible fer un únic for. La variable g actualment està definida com el nombre de gols màxims que pot marcar un equip. Però si haguéssim fet un únic for, no hagués tingut aquesta definició, sino que hagués sigut el número màxim de gols que pot marcar entre els dos equips, i no cada equip.

Per tant, teníem dues opcions: o redefinir la variable g com el número màxim de gols que es pot marcar entre els dos equips (i llavors fer un únic for), o fer dos fors i que g prenés la definició actual, és a dir, el número màxim de gols que pot marcar cada equip. Hem decidit executar aquesta última opció, perquè per la primera opció, no hi ha manera de que els dos equips tinguessin exactament el mateix nombre màxim de gols possibles en un partit. Posem un exemple: imaginem que g = 16. Llavors l'equip visitant pot haver marcat màxim 15 gols i el local 1, o l'equip visitant 6 i el local 10, etc.

Conclusió

Per veure si la funció simulateMatch() funcionava correctament, s'ha fet el següent test:

```
//Player scorerB = homeScorers.getLast();
```

```
            //System.out.println("Hello "+scorerB.getName());
```

S'ha definit un nou jugador (tant de l'equip visitant com el local) com l'últim que ha marcat gol i s'ha imprès per pantalla el seu nom i "Hello". Per tant, la decisió de fer dos fors i no un ha sigut acertada.

També hem volgut crear més jugadors i equips per tenir més joc i possibilitats a l'hora d'imprimir-ho tot.

Una dificultat que ha sorgit a l'hora de programar ha sigut averiguar com obtenir l'índex d'un jugador que ha marcat.

```
indexplayerA = random.nextInt(a); //jugador away que ha marcat
awayScorers.add(awayTeam.getPlayers().get(indexplayerA)); //Player Away
```