Programación III

Trabajo práctico 3

Autores:

Elías Espinillo 40.663.777

Federico Farias 36.495.959

Profesores:

Javier Marenco

Patricia Bagnes

Contenidos

1 Introducción

El objetivo del trabajo práctico es implementar un algoritmo goloso para aplicarlo en una app para asignar árbitros a partidos de un campeonato el cual se llama La hora, referí

- Se tiene 2n equipos en el torneo de modo que cada fecha se juegan n partidos.
- Cada uno de estos partidos tendrán n árbitros los cuales solo pueden dirigir 1 de estos por fecha.

2 Diseño y especificación

Se decide la creación de 7 packages diferentes, cada uno para poder organizar mejor la implementación y el trabajo en grupo. Archivos, controlador, main, objetos, solver, tests y vistas.

En primera instancia, se fue creando las clases de los objetos que formarían parte del campeonato. Las clases Arbitro, Equipo, Partido, Fecha, Fixture y Campeonato en su totalidad. También creamos una instancia especial para poder usarlo en el Solver.

Una vez implementado el diseño que teníamos en mente comenzamos a crear el archivo Json en el cual estarían los datos necesarios y estructura del campeonato. Creamos una plantilla que se encuentra en el proyecto para poder realizar el campeonato de una forma organizada y que sea lo más equitativo entre locales y visitantes de cada partido. Luego de esto y de lo aprendido en clase hicimos la clase LecturaJson la cual se encarga de cargar y leer los datos que se encuentran en el Json.

Luego de implementar lo básico testeamos la lectura del archivo y verificamos que la estructura cargara bien los objetos para poder utilizarlos correctamente en el programa. Una vez que los testeos unitarios pasaban correctamente según nuestros parámetros comenzamos a realizar el algoritmo en el cual los árbitros serian asignados.
Los árbitros forman parte de una lista en el cual se crean con un entero como su código y con un String que servirá más adelante para el usuario pueda elegir un apellido de una lista predefinida, sin embargo el usuario aunque pueda asignarle a cada código de arbitro su apellido nunca podrá saber que arbitro se asignara a cada partido.

Para el Solver utilizando la instancia para realizar la consigna del trabajo practico creamos una matriz en donde tenemos *cantidad de árbitros X cantidad de equipos.* A su vez junto con un Random iteramos desde un índice aleatorio de la lista de árbitros y seleccionamos los árbitros a lo largo de un fixture auxiliar los cuales agregamos para poder removerlos con seguridad y evitar la repetición.
Una vez terminado "parseamos" la instancia a un campeonato nuevo junto con el fixture auxiliar que nos retornó el método asignar y así obtendremos el campeonato con los árbitros asignados y equilibrados de una manera aceptable a lo largo del mismo.

Por ultimo tenemos los paquetes de controlador y vistas. En el primero la clase Controlador inicializa al campeonato y lo usaremos como un nexo junto al paquete de vistas y que tengan total independencia entre el código de negocio y la interfaz de la app.
Para la interfaz decidimos crear un diseño que sea fácil de leer al usuario y que al mismo tiempo tenga una apariencia cercana al deporte en el cual se utilizará la aplicación.
Esta posee un botón en el cual asigna los apellidos de los árbitros a cada código del mismo para luego asignarlos al fixture del campeonato. También tiene un botón el cual reinicia cualquier resultado de la asignación y que pueda realizarse nuevamente.

- **Paquete archivos:**
  - Contiene el archivo Json que se utilizara para leer los datos del campeonato y también las imágenes que se usaran en la interfaz.

  - **Clase LecturaJson:**

    Operaciones:
    - public static Campeonato leerFixture(String archivo):

      Se encarga de leer un archivo Json.
    - public static Campeonato nuevoCampeonato():

      Retorna el campeonato cargado para utilizarlo en el programa.

- **Paquete controlador:**

  - **Clase LecturaJson:**
    - campeonato: Campeonato.
    - posibilidades: String [ ]

    Operaciones:

    - public Controlador():

      Constructor del controlador.
    - Getters().
    - public void resetearNombre():

      Método para reiniciar los árbitros de la app.

    - public void fixtureConAsignacionCompleta():

      Inicia el método del Solver que equilibra y asigna los árbitros.

- **Paquete objetos:**

  - **Clase Arbitro:**
    - codigo: Integer.
    - nombre: String.

    Operaciones:

    - public Arbitro(int codigo, String nombre):

      Constructor del árbitro.
    - Getters() y setters().
    - hashCode() y equals().
    - toString().

- **Clase Equipo:**
  - nombre: String.

  Operaciones:

  - public Equipo(String nombre):

    Constructor de equipos.
  - Getters().
  - hashCode() y equals().
  - toString().

- **Clase Partido:**
  - local: Equipo.
  - visitante: Equipo.
  - arbitro: Arbitro.

  Operaciones:

  - public Partido(Equipo local, Equipo visitante):

    Constructor de partidos.
  - Getters() y setters().
  - toString().

- **Clase Fecha:**
  - partidos: ArrayList de Partido.
  - numeroDeFecha: Integer.

Operaciones:

- public Fecha(ArrayList<Partido> partidos, int numeroDeFecha):

  Constructor de fechas.
- Getter de partidos mediante clonación.
- toString().

- **Clase Fixture:**
  - fechasDelTorneo: ArrayList de Fecha.

  Operaciones:
  - public Fixture(ArrayList<Fecha> fechasDelTorneo):

    Constructor de fixture.
  - Setter de fechas().
  - Getter de fechas mediante clonación.
  - toString().

- **Clase Campeonato:**
  - arbitros: ArrayList de Arbitro.
  - equipos: ArrayList de Equipo.
  - fixture: Fixture.

  Operaciones:
  - public Campeonato(Fixture fixture, ArrayList<Arbitro> arbitros,

    ArrayList<Equipo> equipos):

    Constructor de campeonato.
  - Getters() y setters().
  - Getter de árbitros y equipos mediante clonación.

- toString().

- **Clase InstanciaParaSolver:**
  - campeonato: Campeonato.
  - arbitrosPorEquipos: Matriz de Integers.

  Operaciones:

  - public InstanciaParaSolver(Fixture fixture, ArrayList<Arbitro> arbitros, ArrayList<Equiopo> equipos):
    Constructor de instancia.
  - Getters() y setters().
  - public void elegirArbitro(Partido partido, Arbitro arbitro):
    Verifica que los equipos existan y acomoda la matriz con los índices de cada elemento dentro de la lista de arbitros y en los partidos del campeonato.
  - private void localesYVisitantes(Partido partido, Arbitro arbitro):
    Método que se utiliza en elegirArbitro() para encontrar el índice de los equipos según sean locales o visitantes a través de los partidos y se va sumando las veces que un arbitro corresponde al local o al visitante.
  - private int vecesArbitroPorEquipo(Equipo equipo, Arbitro arbitro):
    Muestra la cantidad de veces que un árbitro corresponde a un equipo.

- **Paquete solver:**

  - **Clase Solver:**
    - insCampeonato: InstanciaParaSolver.
    - fixture: Fixture.

Operaciones:

- public static void inicializarSolver(InstanciaParaSolver c, Fixture f):
  Inicializa la instancia para el Solver.
- public static Fixture asignar(InstanciaParaSolver c, Fixture f):
  Asigna en el fixture los árbitros elegidos desde la instancia ya inicializada.
- public static Fixture fixtureEquilibrado(Campeonato c):
  Retorna el campeonato pasado como parámetro con la asignación de los árbitros.

- **Paquete vista:**
  - La clase ImagenSuavizada la cual implementa Icon para que tanto el fondo como el icono que se utiliza en los árbitros a elegir queden escaladas y tengan una calidad aceptable para la interfaz y mejora visual para el usuario.
  - La clase VentanaPrinicpal en donde se crean e inicializan los elementos de la interfaz utilizando el controlador para los ActionListener de botones, el texto dinámico el cual tiene un scrollPane para que sea más cómodo para poder leer la totalidad de las fechas y la asignación de los árbitros los cuales se encuentran en una la lista de opciones de apellidos para estos.

- **Paquete main y paquete tests:**
  - Aquí se encuentran la clase LaHoraReferiApp donde se le da inicio al programa y en el otro paquete los tests unitarios que vimos necesarios para el programa.

3 Diseño visual de la aplicación