SISTEMA DE GESTIÓN DEPORTIVA

Practica de Programación Orientada a Objetos 2024-2025

Universidad Politécnica de Madrid

E.T.S. de Ingeniería en Sistemas Informáticos

Departamento de Sistemas Informáticos

PRACTICA - POO 24-25

En el contexto de los eventos deportivos que se celebran en el Campus Sur de la Universidad Politécnica de Madrid, se desea crear un sistema que permita registrar a los distintos participantes (jugadores) y realizar los emparejamientos (*matchmaking*) necesarios para los partidos y competiciones que se llevarán a cabo. Para ello, la Escuela Técnica Superior de Ingeniería en Sistemas Informáticos (ETSISI) cuenta con sus estudiantes de la asignatura de Programación Orientada a Objetos (POO).

La ETSISI necesita este software cuanto antes, por lo que les pide a los alumnos que realicen un producto mínimo viable con las siguientes operaciones o casos de uso que serán siempre ejecutadas por un administrador:

- Alta jugador: **crea** en el sistema **un nuevo jugador** del que solo se debe conocer su nombre. **No se pueden crear jugadores existentes.** Se asume que **los nombres son una sola palabra que identifica de forma única a cada jugador.**
- Baja jugador: **elimina** del sistema **un jugador existente.**
- Mostrar jugadores: **muestra la lista de nombres** de aquellos jugadores creados en el sistema.
- Establecer puntuación: **asociado a cada jugador**, **el sistema almacenará un valor decimal** que representa la puntuación de dicho jugador. Por defecto, la puntuación de los jugadores es 0.0, y nunca puede tomar un valor por debajo de -999999.0.
- *Rankear* jugadores: el programa muestra una **lista de nombres y puntuaciones ordenada de forma descendiente**.
- Mostrar emparejamientos: **muestra la lista de emparejamientos** existentes en el sistema
- Borrar emparejamientos: esta operación borra todos los emparejamientos del sistema
- Emparejamiento: crea un emparejamiento entre dos jugadores, sus nombres se especifican como entrada de este comando manualmente, **creando un emparejamiento entre ambos** que se almacenará en el sistema. No se podrá emparejar a un mismo jugador dos veces, en estos casos se omitirán los intentos de emparejamiento posteriores al primero.
- Emparejamiento aleatorio: **empareja todos los jugadores** del sistema de manera aleatoria. Si no se pueden emparejar todos los jugadores, esta operación no estará disponible. Por ejemplo, si hubiera 5 jugadores esta operación no se podría ejecutar ya que uno de ellos no podrá emparejarse.

Para ayudar a los estudiantes en esta tarea, la ETSI ha hecho una plantilla de cómo le gustaría ejecutar las distintas operaciones mediante comandos.

```
> create [player]

> remove [player]

> show

> rank

> score [player];[score]

> show_matchmake

> clear_matchmake

> matchmake [player1];[player2]
```

```
> random_matchmake
```

Teniendo en cuenta todo lo anterior, se pide:

- Implemente un código en Java que realice, y se ajuste, a todas las funcionalidades antes descritas teniendo en cuenta las restricciones explicitas y aquellas implícitas. No es necesario utilizar el paradigma de programación orientada a objetos. Sin embargo, tenga en cuenta que la ETSI nos pedirá cambios y mejoras en breve, por lo que se recomienda introducir este paradigma en la solución.
- La solución de la práctica debe ser un proyecto Maven correctamente realizado y que compile. **Es indispensable que la solución no posea errores en el IDE para que pueda ser corregida.**
- Toda aquella restricción no explicitada en el enunciado que se asuma debe indicarse.
- Al arrancarse la aplicación el software ya debe incluir los siguientes jugadores con sus puntuaciones:
    - Luisa, 4.5 puntos
    - Manuel, 2.7
    - Kurt, 4.0
    - Sofia, 3.8
    - Robert, 3.8

Entregables:

- Un documento PDF que contenga:
    - Portada: debe indicarse los integrantes del grupo y sus correos electrónicos
    - Un diagrama UML que refleje el código
- Proyecto Maven con el código de la solución
- Presentación (se recomienda uso de power point)
    - Explicación de las decisiones de implementación tomadas en la solución, se podrán incluir capturas del código.
    - Presentar el diagrama UML de la solución
    - Demo de la aplicación en funcionamiento que muestre todas las funcionalidades implementadas, así como casos en los que el input es incorrecto demostrando como el código maneja dichos casos correctamente

Los requisitos, casos de uso, de esta aplicación podrán ser modificados (tanto ampliados como reducidos) en