

Simulador de partido de NBA

Construir un programa de consola en Java que simule un partido de la NBA de 48 minutos (4 cuartos de 12 minutos) y vaya mostrando el progreso del marcador a lo largo del juego.

Clase: **Equipo**

- **Atributo (privado):** `String nombre`
- **Constructor:** `Equipo(String nombre)`
- **Métodos:**
 - `String getNombre()`.
 - `boolean sonIguales(Equipo equipo)`: devuelve si son iguales comparando por nombre.

Clase: **Marcador**

- **Atributos (privados):**
 - `Equipo local`
 - `Equipo visitante`
 - `int puntosLocal`
 - `int puntosVisitante`
- **Constructor:** `Marcador(Equipo local, Equipo visitante)`

- **Métodos:**

- `void sumar(Equipo equipo, int puntos)`: acumula puntos al local o al visitante dependiendo el equipo que llegue como parámetro.
- `int getPuntosLocal()`.
- `int getPuntosVisitante()`.
- `Equipo getLocal()`.
- `Equipo getVisitante()`.

Clase: RelojPartido

- **Atributos (privados):**

- `int cuarto`
- `int minutos`
- `int segundos`

- **Constructor:** `RelojPartido()`: Valores por defecto `cuarto: 1` y `minutos: 0` y `segundos: 0`.

- **Métodos:**

- `void avanzarSegundos(int)` : Avanza los segundos dependiendo del parámetro que llega al método. No deben pasarse los 59 segundos. Si se cumplen los 12 minutos, deben reiniciarse y avanzar al siguiente cuarto.
- `int getMinutoDelCuarto()`
- `int getSegundosDelCuarto()`
- `String getCuarto()` : Devuelve el cuarto en forma de cadena: "1er cuarto", "2do cuarto", dependiendo el cuarto actual.

- `boolean finalizoTiempoRegular()`: devuelve true si se terminan los cuartos.

Clase: **Partido**

- **Atributos (privados):**

- `Marcador marcador`
- `RelojPartido reloj`
- `Random random`
-

- **Atributos estáticos constantes:**

- `FALLO = 0,`
- `DOBLE = 2`
- `TRIPLE = 3`

- **Constructor:**

- `Partido(Equipo local, Equipo visitante, Random rnd)`

- **Métodos privados:**

- `int resultadoAleatorio()`: Devuelve **0**, **2** o **3** simulando la jugada (simula fallo/doble/triple).
- `String simularJugada(Equipo equipo)`: Simula una jugada y devuelve el texto correspondiente (reutilizar).
 - Si es **0**, no suma y devuelve "`fallo de <Equipo>`".
 - Si es **2** o **3**, **suma** al marcador y devuelve "`doble/triple de <Equipo>`".
- `void imprimirJuego(int minuto, int segundos, String jugada)`: Recibe por parámetro los minutos, los segundos y una cadena de texto con la jugada actual. Debe imprimirse en consola en el siguiente formato:

```
Min 0:00 (1er cuarto) - Warriors 0 - 0 Lakers (posición de Warriors)
Min 0:20 (1er cuarto) - Warriors 3 - 0 Lakers (triple de Warriors)
Min 0:40 (1er cuarto) - Warriors 3 - 0 Lakers (posición de Lakers)
Min 1:00 (1er cuarto) - Warriors 3 - 2 Lakers (doble de Lakers)
Min 1:20 (1er cuarto) - Warriors 3 - 2 Lakers (posición de Warriors)
Min 1:40 (1er cuarto) - Warriors 5 - 2 Lakers (doble de Warriors)
```

- `void simularTiempo()`: Pausa breve para visualizar el avance (ej, `Thread.sleep(1000)`).

- **Método principal:**

`void jugar()` : implementa el flujo:

- Decide al azar quién empiece con la **posesión** y lo **imprime**.
- En el **próximo avance de tiempo**, ejecuta la **jugada** de ese equipo y la imprime.
- En el **siguiente avance de tiempo** pasa la posición al rival y en el **siguiente** ejecuta su jugada.
- Hacer que avancen los segundos de a 20.
- Los cuartos van cambiando cada 12 minutos.
- Al llegar a los 12 minutos debe informar fin del cuarto e imprimirlo.
- Repite hasta agotar **los 12 minutos de los 4 cuartos**.
- Imprime el **resultado final**.

Método Main:

- **Pedir por consola al equipo local y visitante.**
- Crear objetos **Equipo** y **Partido**.
- Simular partido.