

ISW1 - Nos volvimos a ilusionar!

Luego de haber conquistado el título en Qatar la fiebre mundialista no afloja y por lo tanto se nos pide diseñar una primera versión de un juego de fútbol para conmemorar el casi año de aquella histórica y electrizante final. Y en su honor, por ahora será sólo de **penales**.

El diseñador del juego nos explica que las reglas básicas del tiro de un penal son muy sencillas:

"... Cuando el árbitro pita el penal, el delantero patear al arco la pelota con cierta dirección (grilla de 3x3, siendo 0@0 izquierda-abajo, 1@1 centro-medio y 2@2 derecha-arriba, junto a todas las otras combinaciones válidas) y el arquero intenta atajarla lanzándose también en alguna dirección. Si las direcciones no coinciden (por ej. la pelota va a 2@0 y el arquero se quedó en el aburrido centro 1@1) es gol. Si en cambio las coordenadas son iguales, dependerá de si la fuerza del arquero es mayor o igual a la del delantero. Si lo es, el penal es atajado..."

Hasta aquí, todo maravilloso pensamos, esto se termina más rápido que pasar los tests del Ej. de Stack con ifs... Pero enseguida nos agregan:

"... Igual, ahora que lo pienso mejor, quizás ese caso básico no ocurra casi nunca, porque lo cierto es que dependiendo de la forma en que patear el delantero, la pelota que se use, y cómo se lance el arquero, las cosas pueden complicarse un poquito, así como para darle gracia..."

A nosotros no nos hizo tanta, pero seguimos escuchando la aclaración:

*"... Por ej. en principio los delanteros podrían patear los penales de 2 maneras, **"a colocar"** o **"a matar"**. Si remata "a colocar" el jugador sólo tendrá disponible el 50% de su fuerza base, mientras que el que tira "a matar", dispondrá de toda su fuerza. Sin embargo, tirar "a colocar" garantiza que la pelota siempre vaya donde el delantero quiere, mientras que el remate "a matar" puede que a veces no llegue a la trayectoria pretendida, pues no todos somos Messi... ah claro, y también depende de la pelota que se use, no es lo mismo patear con la **Jabulani** del mundial de Sudáfrica 2010 que era rapidísima y le suma fuerza al delantero si la usa para "matar" que la **playera** que usamos la última vez que fuimos a Costa Azul que siempre salía volando y apenas si llegaba al arco..."*

-*"Pará.. vas muy rápido... dame un Francia para pensarlo un poco..."*, planteamos.

Sin embargo el diseñador continuó inmutable como una fecha, ignorando nuestra perplejidad ante tanta verborragia interminable:

"... El resultado evidentemente también dependerá de cómo se lance el arquero. Si decide elegir una dirección antes de que el delantero pateé, y elige bien, tendrá por ej. beneficios en su fuerza, ya que llegará cómodo. Si en cambio espera al remate para lanzarse, y justo le tiraron a matar con la Jabulani, seguro no tiene chance a reaccionar y será gol. Además estos mismos arqueros que esperan por el remate en vez de elegir una dirección antes, son medio tramposos y se adelantan de la línea del arco para lograr beneficios extra y compensar que se lanzan tarde. Cuando les tiran "a matar" sale bien porque las pelotas salen disparadas rápido, pero "a colocar" y encima con la playera, que tarda en llegar al arco, queda expuesto, el VAR lo detecta y es punto para el delantero..."

*"Igual no te preocupes, **por ahora** sólo hace falta que me modes una ronda de penales de un único tiro y esos dos tipos de pelotas. Encima ya te escribí los tests con todos los casos explicados de forma detallada... y hasta un bosquejo del método principal... Ponete a mirar todo... lo que sí, lo necesito corriendo para antes de las 21:50hs, que hay que mostrárselo a los inversionistas... Ah y en Smalltalk, nada de Unity..."*

Trabajo a Realizar

Deben abrir el archivo de tests que les dejó el diseñador y construir un modelo que los pase (o al menos todos los que puedan!). Dicho modelo debe organizar el conocimiento utilizando clasificación y seguir las heurísticas de diseño vistas hasta ahora en el curso. Se evaluará qué tan bien representa la realidad, su declaratividad, si tiene o no código repetido, el reemplazo de ifs por polimorfismo donde corresponde, la asignación de responsabilidades, los nombres de sus variables, mensajes y categorías, la creación de objetos válidos y completos desde su creación, la no violación del encapsulamiento de los objetos, etc...

Recomendaciones:

1. **Pase los tests** lo más pronto posible con un modelo simple operacional utilizando ifs. Para ello puede seguir el bosquejo que le dejó el diseñador del juego en el método `#arbitroPitaConPenales:...` de la clase `Penales` e ir completando dicho método test a test para pasarlos.
2. Luego reemplace los ifs por polimorfismo en los casos que considere necesario. No olvide tener presente responsabilidades y esencia de los objetos resultantes.

3. Quite el código repetido que le haya quedado, y mejore la declaratividad de sus métodos. Recuerde que cuenta con un ambiente que posee **refactors automatizados** muy útiles para este paso, como el *rename* o el *extract method*.
4. ¡Todo suma! Incluso si no termina, entregue el modelo hasta donde llegó para poder recibir feedback de sus dificultades. Recuerde que el parcial se aprueba con 4.
5. Grabar la imagen cada cierto tiempo.
6. Configurar la opción de “autosave” en la categoría de clases de la solución. Es recomendable hacerlo con versioning cada 60 segundos.

Aclaraciones:

- **No se pueden modificar los tests** del diseñador. Luego, no hace falta que le quite código repetido. El trabajo a realizar está en el modelo, no en los tests.
- Mantenga el castellano que se eligió para escribir los tests.

Ayudas:

- Los tests hacen uso de **Points** para describir coordenadas en la dirección tanto del tiro como del arquero. Un Point se puede crear haciendo: *coordenada* := 0@3. Luego se puede enviar el mensaje **x** a la coordenada para obtener la coordenada horizontal (0) o el mensaje **y** para la vertical (3). No es necesario modelar que las coordenadas sean números enteros porque los tests no lo contemplan.
- Considere utilizar el mensaje **between** para ahorrarse condiciones. Por ej. (5 between: 3 and: 12) retorna true, y (-1@1 between 0@0 and: 2@2) retorna false.

Entrega:

1. Entregar por mail el fileout de la categoría de clase **ISW1-2023-2C-Parcial-1** que debe incluir toda la solución (modelo y tests). El archivo de fileout se debe llamar: **ISW1-2023-2C-Parcial-1.st**
2. Entregar también por mail el archivo que se llama **CuisUniversity-nnnn.user.changes**
3. Probar que el archivo generado en 1) se cargue correctamente en una imagen “limpia” (o sea, sin la solución que crearon) y que todo funcione correctamente. Esto es fundamental para que no haya problemas de que falten clases/métodos/objetos en la entrega.

4. Realizar la entrega enviando mail a la lista de Docentes: ingsoft1-doc@dc.uba.ar y con el Subject: **LU nnn-aa - Solución 1er Parcial 2c2023**. Sólo en el caso de problemas, la lista docentes@isw2.com.ar también estará disponible para recibir mails.
5. Subir a sus repos grupales los archivos **CuisUniversity-nnnn.image** y **CuisUniversity-nnnn.changes**. Debe **zippearlos** previamente para reducir su tamaño o podría dejar sin espacio disponible a sus compañeros.
6. Deberán subirlos al main branch de sus respectivos repos (tenga en cuenta hacer pull antes de ser necesario), y al subdirectorio **/Parcial1/LUnnn-aa/**

IMPORTANTE:

No retirarse del labo sin tener el *ok* de los docentes de haber recibido el mail con su solución. **No abandone su lugar** hasta no haber recibido un mail de los docentes habiendo confirmado su recepción.