Trabajo práctico 1:
"Fichin Pacalgo2"

Normativa

Límite de entrega: Domingo 2 de Mayo, 23:59hs. Subir el pdf al repo grupal
Normas de entrega: Ver "Información sobre la cursada" en el sitio Web de la materia. (`urlnodefinida`)
Versión: 1.0 del 13 de abril de 2021 (ver CHANGELOG-tp1.md)

Enunciado

En este trabajo práctico vamos a extender el modelado del trabajo práctico anterior. Con las partidas de Paclgo2 modeladas, vamos a modelar el sistema de ranking (o high-scores) en el contexto de un juego completo donde distintos jugadores juegan distintas partidas y van quedan grabados sus mejores puntajes.

Considerando un 'fichín' con el Paclgo2 como juego, en el mismo se debe:

- Iniciar partidas nuevas, donde se debe registrar el nombre de la persona que está jugando.
- Jugar la partida hasta que se pierda o se gane.
- En todo momento, se debe poder conocer el ranking. O sea, para cada persona que haya jugado y ganado una partida, la menor cantidad de movimientos con que lo logró.
- De ganarse la partida, se debe actualizar el puntaje del jugador actual si el mismo es mejor (menor cantidad de movimientos) que su mejor puntaje anotado anterior. Si es la primera partida que gana, se anota ese puntaje.
- Además, mientras haya una partida en curso de alguien que ya esté en el ranking, se debe poder conocer su puntaje y el siguiente mejor puntaje de otro jugador en el ranking. Además se debe poder conocer el nombre de quién logró ese mejor puntaje. El objetivo es poder mostrarle a quien está jugando cuál es la maxima cantidad de movimientos que puede hacer para que ganar implique mejorar su posición en el ranking. Queremos mostrar el nombre del contrincante para que se sienta personal.

Debido que este enunciado se apoya en el enunciado anterior, es esperable que aprovechen el modelado ya realizado. El informe de este práctico deberá incorporar todo el modelado que utilicen del trabajo previo, incluyendo modificaciones que sean pertinentes ya sea para adaptarlo a este enunciado o necesarias por correcciones del la entrega anterior.

Entrega

Para la entrega deben hacer `commit` y `push` de un único documento digital en formato `pdf` en el repositorio **grupal** en el directorio `tpg2/`. El documento debe incluir la especificación completa del enunciado presentado usando el lenguaje de especificación con TADs de la materia. Se recomienda el uso de los paquetes de LaTeX de la cátedra para lograr una mejor visualización del informe.

Rubricas

Agregamos a continuación rúbricas que exponen qué se espera de la entrega. Las mismas presentan una serie criterios con los que se evaluarán las producciones entregadas. En términos generales, se considera que entregas con soluciones que solo logren los criterios parcialmente deberán ser reentregados con correcciones en estos aspectos en particular.

Por ser criterios generales, pueden no cubrir todos los detalles relacionados con este enunciado. No obstante buscamos que sean lo más completas posibles. Esperamos que las mismas les sirvan de orientación para la resolución del TP.

	Logra Totalmente	Logra	Logra Parcialmente	No Logra
Abstraccion	Los TADs capturan todos los elementos relevantes del enunciado, y NO capturan elementos irrelevantes.	Los TAD capturan todos los elementos relevantes del enunciado, pero capturan elementos irelevantes.	Los TAD NO capturan todos los elementos relevantes del enunciado.	Los TAD NO capturan todos los elementos relevantes del enunciado y modelan cosas no relacionadas con él.
Abstraccion funcional (o modularización)	Los TADs tienen una responsabilidad adecuada (ni mucha ni poca) y no hay funciones muy extensas. Cuando es adecuado, hay más de un TAD de forma de separar el problema.	Los TADs tienen una responsabilidad adecuada pero hay funciones muy extensas que podrían separarse en funciones auxiliares.	Hay TAD(s) que acumulan más responsabilidad de la necesaria, pero la formulación es correcta y no dificulta la comprensión del modelado.	Hacen todo en un TAD y no es correcto o no se puede entiende sin leer toda la axiomatizacion (es ilegible, no es una especificacion para un humano).
El Comportamiento Automático (CA) es adecuado	Para todos los CA NO es necesario aplicar un generador para que sea efectivo.	Falta algun CA pero es justificable con una interpretacion distinta del enunciado sin simplificar enormemente el mismo.	Falta algún CA que simplifica el enunciado.	No hay CA (y debería haberlo).
Sobreespecificacion	No hay especificacion de aspectos no definidos (por ejemplo, poner una lista donde va un conjunto) ni restricciones que sobresimplifican en problema (por ejemplo, asumir cierto dominio en los parametros).	Hay restricciones innesesarias sobre el dominio pero no sobresimplifica el enunciado (por ejemplo Ponen Nat pero podria ser Int).	Sobreespecifica (por ejemplo: poner lista cuando cuando no hay orden en los el los elementos, tengo una lista de jugadores).	Se sobresimplifica el enunciado.
Declaratividad	Los nombres de las funciones principales (generadores y observadores) ayudan a entender el dominio modelado.	Los nombres de las funciones principales ayudan a entender el modelado pero los axiomas no se condicen con el nombre (ejemplo: existeFantasma() devuelve una lista en lugar de un booleano).	Se abrevia o nombran los metodos de manera que solo se entiende leyendo toda la axiomatizacion.	Se abrevian los nombres y ademas la axiomatizacion no utiliza ninguna funcion auxiliar que ayude a entender por donde van (es ilegible, no es una especificacion para un humano).
Observadores Minimales	Los observadores son mininales y representan todas las instancias posibles validas.	Los observadores NO son mininales, representan todas las instancias posibles validas, pero NO rompe congruencia.	Los observadores NO son mininales, NO representan todas las instancias posibles validas, pero NO rompe congruencia.	Los observadores NO son mininales, NO representan todas las instancias posibles validas, y rompe congruencia.
Correctitud en general	No mezclan generadores entre TAD ni dejan combinaciones de obs/gen sin resolver. Los axiomas terminan y tipan bien.	Algún comportamiento no está perfectamente reflejado en la axiomatización (bugs), hay errores de tipo leves o brazos de condicionales sin resolver. Ninguno de estos errores afectan a la comprensión de la espeficiación.	Hay funciones sin axiomatizar. Hay errores en la axiomatización que dificultan la comprensión de la solución.	Hay funciones sin axiomatizar o errores en los mismos que dejan partes del modelado sin resolver.
Restricciones en funciones	Se aclaran restricciones en funciones que no son necesarias para que la función pueda resolverse pero que acotan correctamente el dominio del problema.	Se detectan las funciones parciales y se restringe su dominio correctamente.	Hay funciones que deberian ser parciales pero no lo son pero su comportamiento extendido no genera comportamientos inválidos.	Hay funciones que deberian ser parciales pero no lo son, permitiendo comportamientos que no pueden resolverse o violan el enunciado.