Trabajo práctico 1:
"Pacalgo2"

Normativa

Límite de entrega: Domingo 11 de Abril, 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 26 de marzo de 2021 (ver CHANGELOG-tp1.md)

Enunciado

Este trabajo práctico plantea modelar una lógica simplificada del Pac-Man de 1980. `https://www.youtube.com/watch?v=dScq4P5gn4A`. Además, por el contexto de la pandemia y por razones de copyright haremos modificaciones pertinentes al juego y lo llamaremos Paclgo2.

El enunciado consiste de dos partes que incrementalmente modelan el juego. Las partes se deben resolver sucesivamente y entregarse ambas. Recomendamos resolver la primera parte, luego copiar la solución y modificarla para resolver la segunda.

Parte 1

- El proposito del juego es llevar al jugador de un lugar a otro sin que se asuste en el camino. Si el personaje se asusta queda inmovil y pierde.
- El jugador se puede mover en las cuatro direcciones (arriba, abajo, izquierda, derecha) de a un casillero por vez.
- En nuestra versión del juego, las partidas comienzan con un mapa ya definido. Un mapa puede considerarse una grilla finita de dos dimensiones. Cada casillero puede ser una pared o un espacio vacío. Dos casilleros especiales se asignan como punto de inicio y de llegada para el jugador. Además se deben asignar casilleros donde se encontraran los fantasmas. Los fantasmas no se pueden mover. Los fantasmas y paredes no pueden solaparse. El jugador puede moverse solamente a través de espacios vacíos.
- La cantidad de fantasmas y sus posiciones se conoce desde el principio como parte del mapa.
- El jugador gana cuando llegó al punto de llegada sin haberse asustado y solo se puede mover si aun no gano o perdió la partida.
- El jugador se asusta si está a menos de 3 casilleros de distancia (manhattan[1]) de un fantasma.

Parte 2

Se agregan las siguientes reglas al juego:

- Para una partida que terminó y fue ganada (el jugador llegó al destino), se debe poder calcular un puntaje definido como la cantidad de movimientos realizados por el jugador. Este puntaje será utilizado en un futuro (tp2) para armar un ranking de partidas resueltas en menos movimientos.
- El mapa ahora puede contar con barras de chocolate definidas al inicio de la partida.
- El jugador puede agarrar un chocolate si pasa por un casillero con uno. De ser así, el chocolate desaparece del mapa.
- Si se pasa por un chocolate el jugador tendrá 10 movimientos de inmunidad al susto. Esto implica que si todavía está comiendo el chocolate (dentro de los 10 movimientos), puede acercarse e incluso atravesar un fantasma sin asustarse.

Entrega

Para la entrega deben hacer `commit` y `push` de un único documento digital en formato `pdf` en el repositorio **grupal** en el directorio `tpg1/`. 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.

[1] Geometría del taxista

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.