Trabajo Práctico Pacman 75.10
Segunda Iteración

Implementación del Pacman 75.10 - Segunda iteración

1 Implementación pedida

Se pide extender el modelo de dominio realizado en la iteración 1 para soportar los siguientes requerimientos

1.1 El laberinto y sus bolitas

El laberinto define: la distribución de lugares válidos de transito - en donde en cada una de estas posiciones hay bolitas (o bolones) - y los lugares posibles por donde se desplazan los fantasmas y el pac-man.
Tiene definido también la posición de inicio del pac-man y el lugar de donde salen los fantasmas (al comienzo y cuando pasan de estado muerto a estado cazador).

Las bolitas y los bolones son comidos únicamente por el pac-man.
Cuando un bolón es comido, todos los fantasmas que están en estado cazador son convertidos a presa (por un intervalo x de tiempo) y todos los que son presa permanecen presa (por el mismo intervalo x de tiempo)

Se pide modelar el laberinto del pac-man (con sus N bolones y sus portales)
El laberinto debe instanciarse a partir de un archivo, donde se indican:

- las conexiones entre los eslabones del laberinto.
- qué eslabones contienen bolitas o bolones.

El formato del archivo de definición de laberintos esta definido, descargar el campus.

No se pide modelar en esta iteración el manejo de partidas con niveles, ni tampoco de puntajes por bolita comida.

Dado que no hay interfaz gráfica en esta iteración, se desea tener un mecanismo de notificación ante cada iteración, intervalo o tick del estado del juego. Dicha notificación debe ser serializada en archivos xml con el estado de cada iteración del juego.
Para ello, ante cada iteración o tick del juego, se solicita leer de un archivo el movimiento elegido para el pacman, realizar los movimientos correspondientes y luego serializar a un xml el estado del juego para ese tick. El formato solicitado para la serialización tanto para el movimiento elegido del pacman, como al del estado del juego, ya estan definidos y se descargaran del campus.

1.2. El pac-man

El mismo recibirá las órdenes del usuario para moverse en las cuatro direcciones. El pac-man siempre tiene una velocidad de 2 x <Velocidad del fantasma en estado normal>. (Los fantasmas tienen la velocidad minima 1 en estado normal, 1,5 en estado molesto y 2 en estado furioso).

Cuando el pac-man come un fantasma cazador (o es cazado por un fantasma) muere.

1.3. Comportamiento autómata de los fantasmas.

Los fantasmas en estado cazador pueden desplazarse al menos en alguno de los siguientes 4 criterios de inteligencia:

● Zonzo: Se mueve aleatoriamente, solamente se lanza a comer al pac-man si este se encuentra a N1 (a cuatro casilleros seguidos por ej) lugares de distancia, si este se escapa a mas de N1 lugares continúa moviéndose aleatoriamente.

● Perezoso: Se mueve aleatoriamente, solamente se lanza a comer al pac-man si este se encuentra a N2 (ocho por ej) lugares de distancia.

● Buscador: Se mueve aleatoriamente, si el pac-man se encuentra a N3 lugares de distancia lo persigue. Si el pac-man se escapa a más de N3 (diez por ej) lugares de distancia el fantasma buscador va hacia la última posición donde lo vio (no se mueve de manera aleatoria) y si en el camino lo ve, se dirige hacia esa posición.

● Buscador temperamental: Es igual al buscador anterior, pero si el fantasma incrementa su nivel de ira incrementa en X la visión (inicialmente tiene una visión de N4 lugares(12). Si se reinicia el nivel de ira, entonces también se reinicia la visión).

El movimiento aleatorio mencionado anteriormente tiene que cumplir con los siguientes requerimientos:

● Una vez que entra a un canal del laberinto, sigue hasta llegar a un cruce o bifurcación, recién ahí vuelve a decidir hacia donde ir. Y no es opción regresar por donde vino, a menos que sea un canal sin salida.

● A la llegada de un cruce si no sabe a donde ir, dado su visión, entonces define aleatoriamente que camino tomar.

● Siempre se mueven, una vez entrado a un canal sigue hasta salir del mismo, al llegar a un cruce decide a donde ir, es decir nunca están quietos.

El fantasma en estado presa debe alejarse del pac-man, es decir que se debe moverse según se describió pero alejándose en los casos que antes se acercaba al pac-man.

1.4. Test unitarios

Se piden un set de test unitarios sobre el modelo de dominio descrito.

El set de test unitarios que se desarrolle debe representar un código valioso, debe motivar al equipo de trabajo a mantenerlo en el tiempo por la utilidad que brinda. Debe seguir las buenas prácticas de implementación de test unitarios.

2. Aclaraciones sobre la implementación

Queda fuera de la iteración los siguientes requerimientos:

- Interfaz gráfica del pac-man.

- Sistema de partidas: Jugadores y turnos, vidas y puntajes, nuevos laberintos. (El juego arranca en estado inicializado según el archivo que lo describe, y termina al morir el pac-man o al comer todas las bolitas)

- Lógica de cómo un fantasma muerto, regresa al inicio (como se desplazaría de muerto a su lugar de inicio) Directamente desaparece y aparece cuando revive del punto de inicio.

- Otras estrategias de los fantasmas cazadores de cómo atacar al pac-man (si eligen el camino mas corto o más largo, si lo encierran en grupo, etc)

- En caso de tomar una decisión de implementación que esté fuera del alcance descripto, esta no debe contradecir ningún requerimiento pedido y se debe enunciar en el informe la hipótesis que la justifique, las mismas deben validarse con el ayudante.

3. Criterios de corrección

- El diseño del dominio

- El diseño de código

- Los test unitarios

- Cumplir con toda la funcionalidad descrita, ambiente y buenas prácticas definidas.

- El informe completo. Carátula, índice, enunciado, justificaciones y decisiones para la resolución acompañada si es necesario de diagramas UML con descripciones orientadas a los contenidos de la materia, extractos de códigos de ejemplo, y conclusiones.

4. Herramientas a utilizar

- Maven >= 2
- JUnit 4.xx
- Repositorio Git

5. Sobre la entrega

Se debe etiquetar en el repositorio la entrega antes de la fecha definida.