FIUBA - 7507

Algoritmos y programación 3

Trabajo práctico 2: Algo-thief

1er cuatrimestre, 2014

(trabajo grupal)

Alumno:

Nombre	Padrón	Mail

Fecha de entrega final:

Tutor:

Nota Final:

Introducción

Objetivo del trabajo

Aplicar los conceptos enseñados en la materia a la resolución de un problema, trabajando en forma grupal y utilizando un lenguaje de tipado estático (Java)

Consigna general

Desarrollar la aplicación completa, incluyendo el modelo de clases e interface gráfica. La aplicación deberá ser acompañada por prueba unitarias e integrales y documentación de diseño. En la siguiente sección se describe la aplicación a desarrollar.

Descripción de la aplicación a desarrollar

Algo-thief es un juego simililar al Carmen Sandiego (http://en.wikipedia.org/wiki/Carmen_Sandiego - https://play.google.com/store/apps/details?id=ar.com.lslfra.carmen). El objetivo del jugador es perseguir y atrapar al ladrón antes del término de una semana

aproximadamente. Para ello se deberá emitir una orden de arresto con pista recolectadas entre los testigos que identifique unívocamente al autor del ilícito.

El policía deberá seguir al maleante a través de distintas capitales del mundo buscando las pistas que le permitan reconstruir la ruta de escape del mismo. En cada capital a su vez deberá visitar distintos lugares en busca de testigos que le provean de pistas para emitir la orden de arresto y que le permitan deducir la proxima capital a visitar.

Reglas

Grados de Policia

Novato: 0 arrestos.
Detective: 5 arrestos.
Investigador: 10 arrestos.
Sargento: 20 arrestos.

Pistas: las mismas se dividen en categorías fácil, media, o dificil. Cuanto mayor es el rango del policía más difíciles son las pistas que le son entregadas. Estas deben estar pre escritas y clasificadas en un archivo xml de donde seran leidas cada vez que se requiera entregar una.

A su vez el contenido de las mismas depende también del lugar en donde son provistas. Los mismos pueden ser:

Bolsa, Banco: entregan pistas relacionadas con la actividad económica del país de donde se dirige el ladrón. Por ej, la moneda, principal fuente de ingreso, etc.

Biblioteca: las pistas son relacionadas con hechos históricos, geografía, etc.

Puerto, Aeropuerto: entregan información descriptiva sobre la bandera, e idioma del lugar.

Tiempo: El jugador tendrá desde el Lunes a las 7hs hasta el Domingo a las 17hs para atrapar al ladrón.

- Entrar a un edificio (1hr la primera vez , 2 hs 2da vez, 3hs 3ra vez).
- Emitir orden de arresto (3 hs).
- Herida con un cuchillo (2 hs 1ra vez, 2hs las próximas veces).
- Herida por arma de fuego (4 hs cada vez).
- Dormir (8 hs por noche)

Tiempo de viajes entre países: los mismos dependen del presupuesto que

interpol le concede al policía. Así, policías de mayor rango, con mayor presupuesto, tardan menos en realizar viajes que aquellos con menor rango.

Novato: 900 km/h
Detective: 1100 km/h
Investigador: 1300 km/h
Sargento: 1500 km/h

La distancia entre los diferentes países puede obtenerse de google maps.

Ladrones: La descripción de los mismos debe estar tabulada en un archivo xml el cual se leerá para emitir la orden de arresto. Las categorias son:

Sexo: Femenino-Masculino
Hobby: Tenis-Música-Alpinismo-Paracaidismo-Natación-Croquet
Cabello: Castaño - Rubio - Rojo - Negro
Senia: Anillo - Tatuaje - Cicatriz - Joyas
Vehículo: Descapotable - Limusina - Deportivo - Moto

Objetos Robados: También serán clasificados en comunes, valiosos y muy valiosos. Cuanto mayor sea la cantidad de arrestos del jugador, casos con objetos robados más valiosos le seran encomendados.

A su vez, la estrategia de escape del ladron dependerá también de que tan valioso sea el objeto que haya robado según la siguiente tabla.

Comunes: 4 paises.
Valiosos: 5 paises.
Muy Valiosos: 7 paises.

La descripcion de los distintos ladrones y paises se puede obtener de: http://www.gamefaqs.com/sms/588168-where-in-the-world-is-carmen-sandiego/faqs/58370

Entregables

- Código fuente de la aplicación completa, incluyendo también: código de la pruebas, archivos de recursos
- Script para compilación y ejecución (ant)
- Informe, acorde a lo especificado en este documento

Forma de entrega

A coordinar con el docente asignado.

Fechas de entrega

Se deberá validar semanalmente con el docente asignado el avance del trabajo. El docente podrá solicitar ítems específicos a entregar en cada revisión semanal.

La entrega final deberá ser en la semana del 30 de junio, en la fecha del curso en que se está inscripto.

-

Informe

Supuestos

Decidimos que la cantidad de ciudades que se deben visitar para llegar a la ciudad donde se encuentra el ladrón sea 6 (seis).

Si un jugador se equivoca y viaja a una ciudad que no es la que corresponde, le damos la opción de volver a la ciudad donde estaba.

El jugador puede emitir varias órdenes de arresto por caso, aunque cada orden le consume tiempo, por lo cual no es muy conveniente.

El arresto del ladrón se produce automáticamente al visitar un edificio de la última ciudad del recorrido (el jugador, sin embargo, no sabe en cuál).

Las pistas de los sospechosos se irán dando de a una en cada ciudad, es decir, en la primer ciudad en el edificio económico por ejemplo se va a dar la pista pelo, en la segunda ciudad en el edificio cultural, la del auto etc. Es decir, se dará en un solo edificio por ciudad, por lo cual hay que analizar si conviene gastar mas tiempo revisando, o si se prefiere encontrar las pistas de casualidad.

Modelo de dominio

Cuando se inicia un juego nuevo, el programa se encarga de leer todos los archivos xml y pasarlos a listas para su posterior utilización. También se le asigna el nombre al policía. Las partidas se encuentran dentro del juego que se acabo de ejecutar.

Durante todas las partidas se utilizara al mismo policía que se designo en el juego y no se volverán a leer los archivos xml, si no que se utilizaran las listas creadas para crear un caso. En cada partida se eligira un camino por el que debe ir el policía y un sospechoso.

Para hacer el juego mas emocionante decidimos , después de leer el archivo con todas las ciudades y sus características, seleccionar algunas al azar y esas guardarlas como el recorrido del ladrón. EL policía deberá ir viajando por todas estas ciudades, sin tener idea cuales son, hasta llegar a la ultima donde estará el ladrón.

En cuanto a viajar entre ciudades, se decidió que cada ciudad debería tener una serie de opciones de las ciudades a las que podía ir. Las ciudades que estuvieran en el camino del ladrón tendrían 4 opciones de ciudades diferentes (contando su anterior y su siguiente). A diferencia del resto de las ciudades (cuando un jugador se equivoca y va una ciudad que no esta en el recorrido) en las que aparecerán mas opción para despistarlos.

En cada ciudad, el policía tendrá la opción de buscar pistas, acceder a la

computadora (donde podrá ingresar las características del sospechoso que se le fue descubriendo o emitir una orden de arresto contra un sospechoso) o viajar. Cuando se llegue al final del recorrido el policía tendrá la posibilidad de atrapar al ladrón (si no se quedo sin tiempo antes). Si no se emitió una orden de arresto o si la orden fue emitida contra una persona inocente, el ladrón se escapara. Por el contrario, si logra atraparlo se le sumara una victoria a su historial, y podrá acceder una vez que haya conseguido las suficientes.

Las pistas que haya el Policía en cada edificio pueden ser de diferente tipo:

Pista del lugar: Es la pista mas común de hallar. Esta pista solo contendrá información del próximo destino del ladrón.

Pista del sospechoso: Vendrá acompañada de una pista del lugar. Se puede encontrar solo una de estas por ciudad y te permitirá obtener información con la cual se va a poder emitir una orden de arresto.

Herida: Este mensaje no es una pista en si, sino que le informa al policía que en ese edificio lo hirieron. Las heridas pueden ocurrir en cualquier momento y no se puede prevenir

Ladrón esta cerca: Esta pista se encuentra solo en la ultima ciudad y se le informa al Policía que algo muy raro esta pasando en la ciudad.

Perdido: Esta pista se encuentra cuando el policía se confunde y va a una ciudad que no era la correcta. En cada edificio de cada ciudad que no corresponda al recorrido que hizo el ladrón, se le informara que no han visto a nadie sospechoso.

Diagramas de clases

[se adjunta archivo .asta]

Detalles de implementación

Pistas:

Tiene dos atributos, String pista del lugar(da la información de la ciudad siguiente o si no vieron al ladrón) y la String pista del ladrón (tiene las características del ladrón). Por defecto la pista del lugar dice que no vio al ladrón y la pista del lugar esta vacía.

Las pistas se utilizan para comunicarle al policía diferentes situaciones como se menciono anteriormente.

Novato, Sargento, Investigador, Detective:

Todos estos tipos de policía poseen la interfaz rango. Cada tipo de rango tiene

un atributo int velocidad con un valor distinto.

Mientras mayor sea el rango, menor sera el tiempo que le lleve viajar, pero también sera mayor la dificultad de las pistas que se le dará.

Para que esto ultimo ocurra, utilizamos el patrón de *double dispatching*, el rango del policía llama al método de edificio que devuelve la pista, y este método recibe a su vez el rango. En base al rango que recibe, el edificio devuelve la pista correspondiente.

Dentro de rango, también agregamos un método que determine si se tiene la cantidad de casos necesarios para que el policía ascienda al rango siguiente. Este método devuelve el rango siguiente que debe adquirir el policía si se tiene la cantidad de casos resueltos necesarios.

Ladron:

Tiene varios atributos Strings que se refieren a sus características (nombre pelo,sexo,auto,marca personal,hobby) y dos atributos boolean tieneOrdenDeArresto y EstaArrestado.

Las características que tiene el ladrón no son infinitas, sino que se creo un enum para cada tipo con algunas opciones.

Edificio:

Tiene 3 atributos Pista que se diferencian en su dificultad.

Aqui se realiza el *double dispatching*, cuando se pide una pista como se menciono anteriormente.

Ciudad:

Tiene 3 atributos de edificios, una String que es el nombre, un ArrayList de Ciudades conectadas, dos boolean EstaElLadron y AcaHierenAlPolicia.

La ciudad es la que se encarga de unir a los diferentes tipos de edificios, las diferentes ciudad, y al ladrón y al policía

También decidimos que tanto para arrestar al ladrón como para que el policía sea herido podíamos crear algún tipo de lógica dentro de ciudad que nos permitiera saber a nosotros dónde iban a ocurrir estas cosas (para luego poder testearlo), pero que a su vez el jugador no pueda descubrir fácilmente cómo es que funciona. Para esto, nos basamos en la cantidad de caracteres que tiene el nombre que ingresa el jugador y creamos un método que determina en qué edificio deben darse estas acciones.

Computadora:

Posee las características que el policía selecciono de los sospechosos, una lista de todos los sospechosos, una lista de los sospechosos luego de filtrar las características ingresadas, y un boolean que se fija si se emitió la orden de arresto.

La computadora permite emitiri la orden de arresto. A medida que el policía va ingresando caracteristicas la computadora va guardando estos atributos. Cuando se emite la orden de arresto la computadora filtra a los sospechosos por las características que les fueron dando y si solo quedo uno puede finalmente emitir la orden de arresto. Los atributos se pueden pisar; es decir, si primero elijo que el ladrón tiene pelo negro y me arrepiento y digo que tiene pelo rubio, el filtrador buscaría solo los rubios.

Coordenada:

Tiene dos atributos int que representa la ubicación en el mundo. Las poseen las ciudades, y se utilizan para calcular las distancias entre estas.

Elector de ladrones:

Posee una lista de ladrones y la dirección donde se encuentra el xml de Ladrones

Su utilidad es leer el archivo xml para crear una lista de lso Ladrones ,seleccionar el un Ladrón y setear sus pistas en las ciudades que va a recorrer el mismo.

Generador de Casos:

Posee una lista de todas las ciudades que hay y una lista de listas de pistas con todas las pistas del juego. Observación: las listas comparten los indices para acceder mas fácilmente a las pistas .

Tiene mucha similitud con el elector de ladrones. Su tarea es leer el xml para crear las listas y seleccionar las ciudades por las que va ir el ladrón. En este ultimo caso se setean las pistas de las ciudades que vamos a recorrer para que nos de la pista de la siguiente, las ciudades continuas para que en ningún momento no se pueda volver al lugar anterior y se esconde al ladrón.

Juego:

Es quien se encarga de unir todas las partes: crea un nuevo juego, crea las partidas, lee los archivos, elije a un ladrón crea a un policía(todo la pare de instancia y seteo) como también, hace que el policía viaje, pueda emitir una orden de arresto, pueda obtener pistas y filtrar sospechosos (toda la parte lúdica).

En resumen, dentro de Juego van a estar todos los métodos que se van a usar para poder hacer funcionar el programa. Esto hace que el controlador solo necesite tener a esta clase y no a todo el resto de las clases .

Policia:

El policía maneja el descuento de horas que se produce por viajar, por consultar pistas en los distintos edificios o por emitir una orden de arresto. El tiempo descontado por emitir órdenes de arresto y por visitar edificios se resolvió por constantes.

Al policía también le pueden ocurrir cosas sin que el lo desee (como es normal en la vida) como recibir herida, resolver un caso o ser ascendido.

Excepciones

La única excepción que creímos conveniente crear es ExcepcionTiempoAgotado, ya que de otra forma se complejizaba la validación de si el jugador podía o no realizar la operación deseada.

Diagramas de secuencia

[se adjunta archivo .asta]

-

Checklist de corrección

Esta sección es para uso exclusivo de los docentes, por favor no modificar.

Carpeta

Generalidades

- ¿Son correctos los supuestos y extensiones?
- ¿Es prolija la presentación? (hojas del mismo tamaño, numeradas y con tipografía uniforme)

Modelo

- ¿Está completo?¿Contempla la totalidad del problema?
- ¿Respeta encapsulamiento?
- ¿Hace un buen uso de excepciones?
- ¿Utiliza polimorfismo en las situaciones esperadas?

Diagramas

Diagrama de clases

- ¿Está completo?
- ¿Está bien utilizada la notación?

Diagramas de secuencia

- ¿Está completo?
- ¿Es consistente con el diagrama de clases?
- ¿Está bien utilizada la notación?

Diagrama de estados

- ¿Está completo?
- ¿Está bien utilizada la notación?

Código

Generalidades

- ¿Respeta estándares de codificación?
- ¿Está correctamente documentado?