Prácticas de Programación Orientada a Objetos
Curso 2006/2007

Entrega 4

El **plazo de entrega** finaliza el día **18 de enero de 2007**

Introducción.

El trabajo de las tres primeras entregas de prácticas tuvo por objetivo el diseño de una biblioteca básica de clases para manejar los conceptos de geometría del juego. A lo largo de las tres entregas se trabajaron los conceptos principales de la Programación Orientada a Objetos: clases y objetos, herencia, polimorfismo, ligadura dinámica, genericidad, interfaces, etc. En esos enunciados se proponía tanto la especificación del trabajo como la orientación acerca de cómo resolverlo.

El enunciado de esta entrega de prácticas es sólo la especificación de los requisitos del videojuego. El propósito de esta práctica es que seáis capaces de identificar las clases e interfaces que caracterizan a la aplicación y las relaciones entre ellas, herencia y clientela. Asimismo, se pretende que apliquéis con criterio los conceptos que proporciona un lenguaje orientado a objetos como Java. Por último, se puede modificar la implementación de las clases de geometría ya desarrolladas con el fin de programar el videojuego siempre que los cambios estén bien documentados.

Especificación de requisitos del videojuego.

El videojuego gráfico *Cubos* consiste en un mundo que contiene líquido sobre el que flotan cubos sólidos que representan los soportes del juego. Los personajes del juego se mueven en ese mundo con distintos intereses. La naturaleza el mundo no permite que la masa de todos los elementos, cubos y personajes, supere un determinado valor crítico. En caso de que se alcance esa masa crítica, se llegaría al fin del mundo.

Entre los personajes del juego destaca el jugador. El objetivo del jugador es evitar que se llegue al fin del mundo. Con este propósito deberá aprovechar el efecto destructivo del líquido, que elimina a cualquier personaje por contacto. Sin embargo, en el mundo aparecen nuevos personajes constantemente. Por tanto, la tarea del jugador no tendrá fin. El reto del juego consiste en impedir el fin del mundo el mayor tiempo posible.

El jugador no se encuentra indefenso ante la hostilidad del mundo. Podrá empujar a cualquier elemento del juego, tanto personajes como cubos soporte. Lo que interesa al jugador es empujar a otros personajes fuera de los cubos soporte de manera que caigan al líquido y desaparezcan.

Sin embargo, la energía que podrá emplear el jugador es limitada. El jugador incrementará su energía tomando alimento del juego, pero perderá energía conforme vaya empleándola. Cuando el jugador empuja a otra entidad, sólo podrá desplazarla una unidad de desplazamiento. La energía que emplea en hacer ese esfuerzo se calcula según las leyes físicas

como el producto de la masa por el desplazamiento. Además, no podrá empujar a un elemento del juego si no tiene energía suficiente para hacer el trabajo asociado a la acción que desea realizar.

En general, el jugador podrá actuar sobre una entidad que se encuentre junto a él. El termino actuar es amplio y se puede entender como interactuar. Por ahora, en el juego sólo podrá actuar sobre dos entidades: comida y regalo. Si el jugador está junto a la comida, actuar significa comer la comida y obtener la energía asociada. Si por el contrario, el jugador actúa sobre un regalo, éste se transformará según las reglas que se indicarán más adelante.

En resumen, las capacidades del jugador son empujar y actuar sobre otras entidades. En general, la condición necesaria para aplicar estas capacidades es que el jugador esté junto a la entidad sobre la que quiera realizar la acción, es decir, que las regiones sean contiguas.

Un personaje en el juego ocupa una región y tiene una masa. Dado que la naturaleza del mundo es cúbica, lo lógico es que todo elemento del mundo así lo sea. Y así es con los personajes. La región que ocupan será un cubo sólido. En el mundo los cubos tienen la restricción de que sus caras deben ser paralelas a los planos que definen los ejes (práctica 1). Asimismo, los movimientos de los cubos están limitados a los seis sentidos establecidos por los ejes de coordenadas. Sin embargo, los personajes del juego sólo podrán decidir moverse en los sentidos marcados por los ejes XY, es decir, los movimientos serán hacia el este, oeste, norte y sur. Además, la acción de movimiento está limitada a una unidad de desplazamiento. Los movimientos hacia arriba y hacia abajo están limitados al paso entre los cubos soporte o la acción de la gravedad del mundo. Por tanto, la acción del salto no está permitida.

Un personaje del juego podrá moverse entre cubos soporte, siempre que éstos sean contiguos. Cuando un movimiento saca a una entidad de un cubo soporte pueden suceder dos casos. El personaje trepará o se dejará caer para situarse en el nuevo cubo, dependiendo de la altura del cubo al que se acceda. Si el cubo es más alto, trepará a la posición correspondiente del nuevo cubo, pero si ya está ocupada esa posición, el movimiento no estará permitido. En caso de que el cubo al que se accede sea más bajo, se dejará caer. Si en este caso existe una entidad en la posición que debe ocupar, esta entidad desaparecerá (otro modo de hacer desaparecer entidades del juego). En cambio, si se realiza un movimiento de salida de un cubo soporte y no hay ningún cubo contiguo, la entidad caerá al líquido y desaparecerá. Es importante hacer notar que la caída sobre otro cubo o sobre el líquido no es realizada por el personaje, sino que al salir del soporte y quedarse en el aire es el mundo quien aplica la acción de la gravedad que lo conduce a un nuevo soporte o al líquido. La acción de la gravedad se describe más adelante.

El jugador no es la única entidad del juego. Existen varios tipos de entidades o personajes. Las entidades más peligrosas son las móviles. Tienen las capacidades de movimiento que se han descrito anteriormente. Además, comparten con el jugador su capacidad para empujar a otras entidades. Uno de estos tipos de entidades se llama topo. Los topos se caracterizan por ser ciegos, es decir, se mueven erráticamente empujando continuamente todo lo que encuentran. Sin embargo, tienen un sexto sentido para evitar caerse al líquido. Otro tipo de personaje móvil son las estatuas. En general están quietas, pero si junto a ellas se sitúa una entidad móvil, la empujan y se colocan en la posición de la entidad. Entre las entidades móviles destaca un grupo que tienen capacidades de visión. Es decir, son capaces de analizar una parte del espacio tridimensional del mundo y actuar en función de ese análisis. De este grupo los más peligrosos son los broncas. Su objetivo es sacar al jugador del cubo en que se

encuentre. Es decir, lo buscarán, se aproximarán hacia él, y en cuanto estén a su lado lo empujarán. Otros personajes móviles con visión no menos peligrosos son los kamikazes. Su propósito es hundir el cubo en el que se encuentre el jugador. Para lograr ese objetivo identifican el cubo sobre el que está el jugador y se dirigirán hacia él con el fin de hundirlo y morir todos.

Los algoritmos de movimiento de algunos personajes como los broncas o los kamikazes pueden llegar a ser muy sofisticados. Sin embargo, no es el propósito de este videojuego programar personajes inteligentes. Es decir, los algoritmos que se programen deben ser eficaces, pero no es necesario que sean eficientes y menos aún óptimos.

El jugador no es una entidad móvil como las demás. El resto de entidades móviles del juego podemos considerarlas autónomas, ya que tienen un objetivo claro en el mundo y actúan consecuentemente. Es más, las entidades móviles autónomas también se caracterizan por no perder energía. Sin embargo, el personaje del jugador es controlado. Es decir, si el usuario del juego no le indica qué tiene que hacer, no hace nada. Por tanto, el usuario ha de controlar al jugador con las cuatro direcciones de desplazamiento (norte, sur, este y oeste), el empujón y la actuación. Además, la acción de empujar se realiza en un sentido. Se toma como criterio tomar el sentido del último movimiento (o al menos la intención de moverse, hay movimientos que no llegan a efectuarse).

En el juego no todas las entidades son móviles. Hay entidades estáticas, que pueden ser empujadas o arrojadas al líquido, pero que no tienen movimiento ni acción por sí mismas. La entidad estática más útil del juego es la comida. La comida contiene una cantidad de energía que sólo podrá tomar el jugador. Otra entidad estática es el regalo. El jugador, cuando actúa sobre un regalo, éste desaparece y aparece en su posición, con las mismas dimensiones, comida o alguna de las entidades desaparecidas en el juego. La probabilidad de que aparezca directamente comida será del 50%. En el caso de que las entidades desaparecidas, si no hay ninguna, aparecerá una estatua.

Los cubos soportes determinan la configuración del mundo. Estos cubos flotan sobre el líquido. La flotación de un cubo depende de su densidad. Si la densidad del cubo es mayor que la del líquido se hundirá. En el caso de que no lo sea, se mantendrá a flote. Por simplicidad, vamos a suponer que la flotación es todo o nada, es decir, que los cubos no se van hundiendo conforme aumente su densidad. Si la densidad es menor que la del líquido, el cubo deberá sobresalir completamente. En cambio, si supera el límite, se hundirá por completo y desaparecerá el cubo soporte y todas las entidades que soporta.

En el mundo aparecen nuevas entidades periódicamente. La frecuencia de aparición se puede establecer en un valor constante, por ejemplo, cada 5 segundos. Por simplificar la creación de nuevas entidades, se propone que aparezcan en el juego copias de entidades que hayan desaparecido, y en el caso de que no las hubiera, que aparezca comida. La posición donde deben aparecer se determinará como un valor aleatorio dentro de las dimensiones del mundo. Por último, podemos suponer que si la posición de inserción de la nueva entidad no es legal (por ejemplo, hay colisión con otro elemento), la entidad se descarta.

El mundo puede considerarse como un cubo contenedor que contiene al resto de elementos del juego. No obstante, extiende las características básicas de los cubos contenedores al establecer una serie de leyes que gobiernan el mundo:

- Una vez dentro, ningún elemento podrá salir de las dimensiones del mundo.
- No puede haber colisiones entre los elementos del mundo. Es decir, no debe haber intersección entre las regiones de los elementos. Se entiende por elemento tanto las entidades como los cubos soporte.
- No se podrá introducir un elemento en el mundo si la posición donde desea introducirse ya está ocupada total o parcialmente.
- El mundo tiene dos partes, aire y líquido. Un elemento que esté en el aire caerá por la acción de la gravedad que será controlada por el mundo. Si el elemento es una entidad y cae encima de otra entidad, la entidad sobre la que cae desaparece del mundo. Si cae sobre un soporte, se queda en el soporte, y si cae al líquido, desaparece. En cambio, si el elemento es un cubo soporte y en su caída encuentra a otro cubo soporte, desaparecerá el que está debajo. Si cae sobre el líquido, flotará en función de su densidad.
- El mundo periódicamente revisa los elementos que quedan en el aire para aplicarles la gravedad. La aplicación de la gravedad a un elemento no es progresiva, es decir, se le aplica de una vez toda la acción de la gravedad hasta llegar a una posición estable o desaparecer. En el caso de las entidades, se entiende por posición establece quedar situadas sobre un cubo soporte. Para los cubos soportes, una posición estable es flotar sobre el líquido.
- Cualquier entidad que caiga al líquido muere.
- El mundo *animará* periódicamente a todas las entidades móviles aplicándoles un turno de ejecución para realicen la funcionalidad que les caracteriza. El periodo de aviso debe ser un valor constante que habrá que ajustar de forma sensata para conseguir el dinamismo del juego. Nótese que en cada turno de ejecución una entidad móvil sólo podrá realizar una de sus capacidades, es decir, un movimiento, un empujón o la actuación, si es el jugador. Y que además, la acción de movimiento desplaza la entidad sólo una unidad.

Requisitos de implementación.

1. Interfaz gráfica.

La interfaz del videojuego será gráfica. Los profesores de la asignatura ofrecerán una biblioteca de clases documentadas en javadoc que proporcionan la interfaz gráfica del juego. En los seminarios no se explicará nada acerca de estas clases. Uno de los objetivos de la práctica es valorar la capacidad para interpretar el código hecho por terceros e integrarlo dentro de la aplicación.

2. Configuración.

La interfaz gráfica ofrece una opción para seleccionar el mundo. Cada vez que se cree una nueva partida se utilizará el último mundo seleccionado. Un mundo se define en un fichero de texto. A continuación se incluye un ejemplo con comentarios donde se ilustra el formato. En general, las palabras clave son insensibles a mayúsculas/minúsculas.

```
# Esto es una línea de comentarios. Podrá haber tantos comentarios como sea necesario.

# Dimensión del cubo contendor
# Se introduce la esquina inferior y el tamaño del lado

Dimensiones (3, 5, 2) 50

# Altura del líquido y densidad

Liquido 20 1

# Los cubos soporte se sitúan en el plano XY con una posición absoluta
# Un cubo soporte se caracteriza por el punto inferior, el tamaño y masa

Soporte (8, 8) 5 3
Soporte (18, 20) 9 2
Soporte (39, 27) 8 10
…

# Entidades del juego
# Para todas las entidades se asume un tamaño y masa por defecto
# Se sitúan en el plano XY con coordenadas absolutas

# El jugador tiene un nivel de energía inicial
Entidad Jugador (12, 7) 100

Entidad Broncas (7, 4)

# La comida tiene un aporte energético
Entidad Comida (3, 7) 10
Entidad Regalo (5, 7)
…
```

Del fichero de configuración se desprenden varias implicaciones. La primera es que se asume un tamaño y masa por defecto para todas las entidades (personajes). Por simplificar la representación podemos suponer que tienen tamaño 1 (cubos 1x1x1) y una masa de 1. La segunda consecuencia es que los elementos en el juego se sitúan con coordenadas absolutas en el plano XY. Además, se establece la restricción de que la declaración de las características del cubo contenedor debe aparecer al principio del fichero y que han de declararse todos los soportes antes de introducir las entidades. La última restricción es que debe declararse un solo jugador.

La segunda consecuencia extraída del formato del fichero (coordenadas absolutas sobre el plano XY) facilita la configuración del juego pero dificulta la situación de los elementos en el

mundo. Este modo de configuración asume que los elementos se dejan "caer" sobre el mundo. Esta característica también se aplica a la aparición espontánea de personajes descrita previamente. Tal como se ha comentado anteriormente, el mundo se caracteriza por el hecho de que cualquier entidad que esté en el aire caerá hasta encontrar un soporte o el líquido, en cuyo caso moriría. A los soportes les sucede lo mismo. Caen hasta encontrar otro soporte (que destruirían) o encontrar el líquido. En este último caso, se comprobaría la flotación. Y en caso de flotar, seguirían en el mundo.

En el proceso de construcción del mundo los elementos se van dejando caer en el orden en el que son declarados. Se situarán pegados al "techo" del mundo y se dejarán caer. Por este motivo no se indica la coordenada Z en la configuración. Por tanto, situaremos la esquina superior del elemento a la misma altura que la esquina superior del cubo contenedor antes de introducirlo en el mundo.

Asimismo, existe la restricción de que el vértice inferior del elemento a insertar debe estar por encima del nivel del líquido. Además, la política de control del contenedor debe evitar que al situar inicialmente el elemento haga intersección con cualquier otro elemento del juego. En general, se aplicarán las leyes del mundo.

Todas estas restricciones ponen de manifiesto que el fichero de configuración del mundo puede ser erróneo. Puede tener errores sintácticos (palabra clave desconocida, falta un valor, etc.), en el orden de las declaraciones (por ejemplo, se declara un cubo soporte antes que las dimensiones del mundo) o en la semántica de las declaraciones (la situación inicial de un elemento está fuera del cubo contenedor, al situar un elemento se destruye otro, etc.). Los dos primeros errores los consideraremos como críticos: no se puede crear un mundo con errores sintácticos o de declaración. Sin embargo, la aparición de errores semánticos será informada al usuario con una ventana de diálogo y la descripción de los errores se guardará en un fichero de texto. Por último, es interesante comprobar la validez del fichero de configuración en el momento en el que es seleccionado. Esto implica que habrá que intentar crear un mundo y comprobar si existen problemas de configuración.

3. Robustez de la aplicación.

La aplicación debe ser robusta. Esto significa que las situaciones de error que puedan producirse deben ser tratadas adecuadamente. Para ello utilizaremos el mecanismo de excepciones de Java. Nótese que algunas clases de la biblioteca de Java lanzan errores no comprobados (por ejemplo, no se puede acceder a una posición ilegal de una cadena) y excepciones comprobadas (en la entrada/salida este tipo de excepciones es habitual).

En el código de la aplicación debemos dar un tratamiento adecuado a las excepciones de las clases de Java que se utilicen. Asimismo, el código programado debe contemplar las situaciones de error. Será necesario definir excepciones propias y utilizarlas convenientemente. Además, se seguirán documentando las precondiciones de los métodos y se controlarán con excepciones de *Runtime*.

El tema de excepciones será tratado fundamentalmente en el horario de teoría de la asignatura, aunque en los seminarios de prácticas se comentarán los aspectos más destacados. El tratamiento de errores es importante. En los criterios de evaluación se podrá valorar negativamente este apartado si el tratamiento que se da a los errores es demasiado permisivo o

incorrecto. Es decir, una programación inadecuada de la robustez de la aplicación podrá perjudicar la nota obtenida por méritos en otros apartados de la práctica.

4. Temporizadores.

El uso de temporizadores es esencial para el desarrollo del juego. Por un lado, el mundo debe hacer actuar periódicamente a las entidades móviles del juego para cumplir con su funcionalidad. Por otro lado, es necesario controlar el tiempo que el jugador consigue sobrevivir en el mundo. Se recomienda el uso de los temporizadores de de Java para programar toda esta funcionalidad. En la clase *javax.swing.Timer* se puede encontrar más información sobre el uso de temporizadores.

Entrega.

Todas las clases deberán estar documentadas adecuadamente en **javadoc** y deberá generarse la **documentación** también para esta entrega. El **responsable** del grupo (el primer alumno del grupo) deberá dejar la práctica en su zona de contenidos de la asignatura. La falta de entrega en el plazo indicado implica la no corrección de la práctica y por tanto significa no presentar las prácticas en esta convocatoria. Para las convocatorias de febrero/septiembre las prácticas serán distintas.

Esta entrega también deberá entregarse en **papel**. El formato de la entrega debe ser el siguiente:

- Una portada con el nombre y dirección de correo electrónico de los miembros del grupo de prácticas, la titulación y el profesor
- El código fuente de todas las clases implementadas hasta el momento impreso preferiblemente por las **dos caras** del folio.
- El código impreso no debe encuadernarse, simplemente meterlo en una funda de plástico.
- El trabajo debéis dejarlo en Conserjería.

Asimismo, hay que generar el fichero "**cubos.jar**" con la aplicación y un fichero por lotes para lanzarla.

Criterios de evaluación.

1. Documentación Javadoc *significativa* en todas las declaraciones y generación de la documentación (5 %).
2. Correcta aplicación de los conceptos de la Programación Orientada a Objetos: clases y objetos, herencia, genericidad, interfaces, etc. (70%)
3. Robustez de la aplicación (10%)
4. Entrada/Salida y procesamiento de cadenas (5%).
5. Otros criterios: uso de colecciones, legibilidad, etc. (10%).

Nótese que en esta entrega no se incluye un criterio sobre **corrección y completitud** de la funcionalidad. Es condición imprescindible para aprobar la práctica implemente correctamente todos los requisitos.