**FIUBA - 7507**

**Algoritmos y programación 3**

*Trabajo práctico 2: BombitaRodriguez*

1er cuatrimestre, 2012 (trabajo grupal)

|  |  |  |
| --- | --- | --- |
| Nombre | Padrón | email |
| Andres Sanabria | 93403 | andysanabria\_2e@hotmail.com |
| Martin Carrera | 93521 | Martinc.92@hotmail.com |
| Ezequiel Reyes | 92119 | ezemr\_cuervo@hotmail.com |
| Hugo Chavar |  | hechavar@gmail.com |
|  |  |  |

Página en blanco (intencionalmente)

Introducción 3

Objetivo del trabajo 3

|  |  |  |
| --- | --- | --- |
| Consigna | 3 |  |
| Entregables |  | 5 |
| Entrega | 5 |  |
| Fechas de entrega | 5 |  |
| Pruebas | 5 |  |
| Documentación 6 | | |
| Supuestos | 6 |  |
| Elección del modelo entre los disponibles del TP1 | 6 |  |
| Modelo de dominio | 6 |  |
| Diagramas de clases | 6 |  |
| Detalles de implementación | 6 |  |
| Excepciones | 6 |  |
| Diagramas de secuencia | 6 |  |
| Diagrama de estados | 6 |  |
| Diagrama de paquetes | 7 |  |
| Código fuente |  | 7 |
| Checklist de corrección |  | 8 |

**Introducción**

***Objetivo del trabajo***

Aplicar los conceptos enseñados en la materia a la resolución de un problema, trabajando en grupo utilizando Java o C#.

***Consigna***

Desarrollar la aplicación completa, incluyendo el modelo de clases, la interfaz con el usuario, con sus pruebas correspondientes, de un juego cuyas reglas se muestran a continuación, y se pedirá pruebas unitarias y de integración completas, y elementos de documentación que se detallan en este enunciado.

**Introducción**

La empresa *Unidos o dominados Inc.*, creadora de grandes éxitos en el campo de juegos de video, está encarando un proyecto para realizar una remake de un famosísimo juego de los años

'80 (bomberman – [a ver que es esto?!)](http://www.minijuegostop.com.mx/items/accion/0/15391_bomberman-clasico/). Para levar adelante ésta tarea ha elegido a su grupo de

trabajo para realizar el diseño y desarrollo el producto en cuestión.

El juego consiste en un arcade donde Bombita Rodriguez (en adelante Bombita) tiene que destruir, en sucesivos niveles a aquellos que están en contra de la Nación.

**Reseña del juego y su modalidad:**

Bombita deberá destruir a sus enemigos para así liberar ciudadanos del Imperialismo. Una vez liberada una ciudad pasará a la siguiente. Para ver como pasar de nivel ver sección **Paso a nivel siguiente.**

Para llevar a cabo la liberación contará con bombas de detonación retardada que podrán ser de varios tipos, según se especifique en la **Tabla 1**. Estas bombas en función de su tipo tiene una onda expansiva y un poder de destrucción dependiendo del tipo.

El ambiente en el cuál Bombita se desempeñará será un mapa de dos dimensiones. Este mapa puede tener tando espacios vacios (por los cuáles todos los personajes pueden trasladarse) y espacios con obstáculos, obviamente por estos nadie puede pasar en principio.

A modo de ejemplo, ésta sería una configuración de mapa posible:

Los casilleros que están vacíos o en blanco pueden ser utilizados por los personajes para desplazarse. Aquellos etiquetados con una **OX** están ocupados con obstáculos, que pueden tener diferentes características. Tanto los obstáculos como sus características se detallan más adelante en la **Tabla 2.**

Aquellos casilleros ocupados pueden ser desalojados utilizando bombas. Por ejemplo: Si Bombita planta bomba en el casillero (5,2) luego de unos segundos la bomba detonará y el obstáculo puede: destruirse completamente, dañarse o no dañarse; en función de sus características (ver **Tabla 2**).

Bombita luchará contra varios enemigos, cada uno de estos tienen características/comportamiento especiales, las cuales se describen en la **Tabla 3.** La manera con la que cuenta nuestro héroe podrá destruir a sus enemigos es a traves de la utilización de bombas.

Un poco más acerca de Bombita:

Inicialmente Bombita ingresa la ciudad portando un tipo de bomba (preferentemente el más inofensivo) una velocidad de acción o desplazamiento y una vida, la cual perderá en caso de ser alcanzado por cualquier proyectil, bomba u objeto contundente. Bombita podrá modificar su comportamiento y el de sus bombas tomando del mapa objetos especificados en la **Tabla 4**. Estos objetos son el resultado de la destrucción de los obstáculos.

**Paso a nivel siguiente:** Bombita liberará una ciudad una vez que haya eliminado a todos sus enemigos y haya encontrado la salida de la Ciudad en cuestión. La salida de una ciudad es secreta y está ocultada por un objeto del mapa, que al ser destruido dejará ver ésta salida.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Características** |  |  |
|  | **Destrucción** | **Retardo** | **Onda expansiva** |
| Molotov | 5 unidades | 1 seg | 3 casilleros a la redonda |
| ToleTole | Infinitas unidades | 5 seg | 6 casilleros a la redonda |

**Tabla 1** – Armamento (Tipos de bombas)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Características** |  |  |
|  | **Durabilidad** | **Observaciones** |  |
| Bloque acero | Solo se puede destruir con bomba **Toletole** | No puede ocultar salida de ciudad |
| Bloque Cemento | 10 unidades | Puede ocultar salida de ciudad |
| Bloque ladrillos | 5 unidades | Puede ocultar salida de ciudad |

**Tabla 2 –** Obstáculos

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Características** |  |  |
|  | **Habilidades** | **Resistencia** |  |
| Cecilio | Al igual que Bombita planta bombas con el fin de destruir a nuestro héroe. | 5 Unidades |
| Los López  Reggaé | Siempre subidos a un tanque que les permite una mayor  velocidad de desplazamiento. En lugar de plantar bombas lanzan proyectiles que detonarán a una distancia y tendrán una onda  expansiva a determinar. | 10 Unidades |
| Los López Reggaé Alado | Al igual que Cecilio puede plantar bombas pero además puede atravesar cualquier obstáculo por encima sin que este esté destruido. | 5 Unidades |

**Tabla 3 –** Enemigos de Bombita

|  |  |  |  |
| --- | --- | --- | --- |
| **Artículo** | **Características** |  |  |
|  | **Efecto** | **Duración** |  |
| Habano/Chala | Acelera el ritmo cardíaco de Bombita y le permite correr | Hasta que muere |
| Bomba  Toletole | Ahora bombita plantará bombas toletole únicamente. No se agotan | Hasta que muere |
| Timer | Reduce el tiempo de retardo 15% | Hasta que muere |

**Tabla 4 –** Artículos (Modifican comportamiento de las bombas y de bombita)

**Entregables**

Se deberá desarrollar la aplicación completa, incluyendo la interfaz gráfica. Deberá poder grabarse los puntajes altos (los n mas altos, configurable) y grabar el estado del juego para retomarlo nuevamente en otro momento.

Deberá entregarse:

● todas las clases con sus métodos , organizados en paquetes/namespaces según criterio del alumno.

● conjunto de pruebas unitarias que muestren el uso del modelo y su correcto funcionamiento.

● documentación completa del código fuente

● documentación completa del diseño de clases (ver siguientes secciones del enunciado)

● la aplicación deberá poder correrse desde consola, para lo cual deberá proveerse el archivo de Ant o NAnt correspondiente.

***Entrega***

Este documento se deberá completar con las secciones correspondientes. Deberá acordarse con su ayudante la forma de entrega de los elementos que se evaluarán.

***Fechas de entrega***

**Entrega 1** (semana del 7 de junio de 2012): se deberá entregar la primer versión de la documentación con todos correspondientes y el modelo de datos con sus pruebas unitarias.

**Entrega final** (A determinar día de junio de 2012): se deberá entregar la aplicación completa junto con la documentación revisada y corregida según los comentarios realizados por el ayudante asignado.

***Pruebas***

Todas las clases deberán contar con sus pruebas unitarias **COMPLETAS**. La aplicación deberá contar con pruebas de integración **COMPLETAS**.

**Documentación**

***Supuestos***

1. Todo Explosivo podrá dañar a cualquier personaje que se encuentre en la zona de alcance del mismo al momento de su explosión
2. Todo Proyectil será lanzado en la dirección hacia donde apunte el personaje. Se escogió un al alcance y un Poder de expansión arbitrario.
3. Cuando un personaje planta una bomba lo hará en la posición donde se encuentra al momento de plantarla.
4. No se podrá plantar una bomba en un casillero que ya contenga una, pero si podrá explotar en él un proyectil.
5. Dos personajes podrán transitar la misma casilla sin causarse daño alguno.
6. Los casilleros con bombas podrán ser transitados por los personaje

***Modelo de dominio***

En una primera etapa, entre los integrantes del grupo de trabajo se realizaron varios diagramas de clases, los cuales se fueron comparando y se fue discutiendo las ventajas y desventajas de cada uno. A partir de ello se obtuvo un modelo estimado de cómo se podría llegar a solucionar el problema y con ese idea comenzamos la etapa de la programación. Sin embargo, durante el desarrollo de la implementación fueron surgiendo múltiples inconvenientes que nos obligaron a ir modificando este modelo inicial. Se debieron agregar clases, modificar relaciones entre ellas y eliminar otras cuya existencia terminó resultando sin sentido. Con la ayuda de algunos patrones de diseño se fueron solucionando otros inconvenientes y a partir de las pruebas realizadas se fueron encontrando errores y resolviéndolos. Sin embargo, en esta etapa del trabajo hubo ciertos aspectos del problema que todavía no hemos podido solucionar.

Ciertas Clases resultaron de gran importancia como ser las clases Juego y la clase Mapa. Cada juego deberá tener durante el desarrollo de un nivel un mapa cargado, el cual deberá relacionarse con cada uno de los objetos allí presentes. En ese contexto, podremos observar que la modificación del mapa podrá ser solicitada desde diversos lugares.

***Diagramas de clases***

Ver Archivo .Asta

***Detalles de implementación***

1)Es importante destacar a la clase Juego. Esta clase presentaba la necesidad de

tener una única instancia pues toda vez que necesitara llamar a una instancia de

esta clase debíamos referirnos al mismo Juego. Con la implementación del patrón

Singleton pudimos solucionar este inconveniente.

2)Otra de las problemáticas que surgió fue la de tener un objeto que

dependiendo de su estado se comportara de una u otra forma. Esto sucedió con

las casillas y lo pudimos solucionar aplicando el patrón State, el cual nos permite

guardando el estado del casillero variar su comportamiento.

***Excepciones***

1)CasillaYaIngresada:

EstadoNulo: Al querer agregar una casilla en el mapa cuya referencia sea null me lanzará un excepción.

NoExisteCasilla

NoPuedeAgregarMovil

PosicionNula

PosicionFueraDeRangoEnMapa

***Diagramas de secuencia***

[Mostrar las secuencias interesantes que hayan implementado. Pueden agregar texto para explicar si algo no queda claro]

***Diagrama de estados***

[Mostrar los estados generales de al aplicación]

***Diagrama de paquetes***

[Mostrar los paquetes en los cuales se organiza el código fuente]

**Código fuente**

[Código fuente impreso de las clases del dominio (no pruebas). Consultar con su ayudante si el código se incluirá o no aquí]

**Checklist de corrección**

Esta sección es para uso exclusivo de la cátedra, 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?

**Diagramas de estado**

● ¿Está completo?

● ¿Está bien utilizada la notación?

**Código**

**Generalidades**

● ¿Respeta estándares de codificación?

● ¿Está correctamente documentado?