TP2: Class Emblem

Detalles del trabajo

Objetivos

- Analizar el problema planteado correctamente.
- Diseñar un modelo general de solución utilizando correctamente los conceptos de **programación orientada a objetos.**
- Demostrar el conocimiento de principios y patrones de diseño y de las buenas prácticas de programación.

Indicaciones del trabajo

Grupos de trabajo

El trabajo se desarrollará de forma grupal, en grupos de 4 alumnos. El trabajo práctico debe realizarse utilizando Git y Github como herramienta de colaboración.
Todos los alumnos integrantes del grupo **deben** participar activamente del desarrollo del trabajo de forma equitativa. Mientras la participación sea equitativa, queda a criterio de cada grupo la forma de trabajo y organización.

Tecnología

El trabajo debe realizarse en **Java**, utilizando cualquier distribución de **Java 21**. El proyecto debe ser un proyecto Maven. Debe poder ejecutarse utilizando un comando de Maven.

Tiempo de trabajo y entregas parciales

Se contarán con 5 semanas completas para llevar a cabo el desarrollo del TP.
El trabajo cuenta con una entrega parcial no obligatoria en la semana 2 del trabajo, para hacer junto al corrector asignado una evaluación prematura del diseño planteado, y la separación de incumbencia entre las distintas partes del programa.
A final de la fecha establecida, debe realizarse la entrega final del trabajo.

Modalidad de entrega

Para realizar la entrega se debe:

- Github:
    - El repositorio tiene que ser privado y debe estar su corrector invitado al mismo.
    - Para realizar la entrega se debe crear un branch llamado tp-2, con el código correspondiente a la entrega. No se debe modificar más luego de la fecha de entrega.

    - Se debe contar con un README.md que explique cómo correr el programa “desde cero”, cómo instalar las dependencias y otras explicaciones pertinentes. El código debe poder correrse con un comando que compile el código y lo ejecute, **no es válido subir un ejecutable y que el comando simplemente lo corra**.

- Mail: Mandar un mail a algoritmos3.fiuba@gmail.com indicando número de grupo y sus integrantes (nombre, apellido y padrón), aclarando cuál es la branch y el repositorio de la entrega. El mail debe llevar adjunto el informe en formato PDF y asunto debe TP2 - <nombre del grupo>.

Fecha de entrega

La entrega definitiva, que debe alinearse a lo especificado en la Modalidad de entrega, debe realizarse con anterioridad a las 23:59hs del 13/11.

Dominio

Introducción

El destino de un reino en guerra está en tus manos… y solo tus decisiones estratégicas podrán salvarlo. Inspirado en el universo de Fire Emblem, este trabajo práctico propone el desarrollo de un sistema de combate táctico por turnos, donde cada movimiento puede significar la victoria… o la pérdida definitiva de tu ejército.

Tu misión es clara: programar un ***sistema de batalla*** *por turnos, donde héroes y enemigos se enfrentan. Cada personaje es único, con sus propias estadísticas, habilidades y armas. No basta con programar:* ***vas a tener que pensar como un estratega.*** *¿Qué unidad mover primero? ¿Conviene atacar o esperar? ¿A quién curar para mantener a tu ejército en pie?*

Cada línea que escribas será una orden en el campo de batalla. Cada decisión de diseño, una jugada táctica que puede salvar a tus aliados o condenarlos al olvido. Y al final, cuando el humo se disipe… solo quedará un bando en pie.

La batalla ha comenzado. ***¿Estás listo para liderar tu ejército?***

Descripción del juego general

El juego consiste en un combate táctico por turnos entre dos ejércitos: el Reino Druida y el Reino Nigromántico.

El enfrentamiento ocurre en un mapa cuadriculado, donde cada casilla representa un terreno (llanura, bosque, montaña, fuerte, etc.) que influye en el movimiento y combate.

El objetivo del jugador es ganar el combate, que se logra derrotando al “lord” del reino rival. También puede acabar antes una partida si un jugador decide rendirse.

Inicio de partida

Al comienzo del programa, el usuario deberá seleccionar un mapa desde el cuál se cargará el tablero con su respectiva disposición y relieves.

Luego, deberá seleccionar un ejército y arsenal de armas e instrumentos a utilizar por ambos jugadores durante la partida.

Una vez cargada la partida, cada jugador deberá ubicar a su **lord** en una posición estratégica del mapa, a partir de la cual podrá comenzar a desplegar sus otras unidades en turnos siguientes.

Turnos

El juego es alternado por turnos. En cada turno, el jugador podrá optar por un número de acciones posibles, basadas en la posición de sus unidades en el mapa:

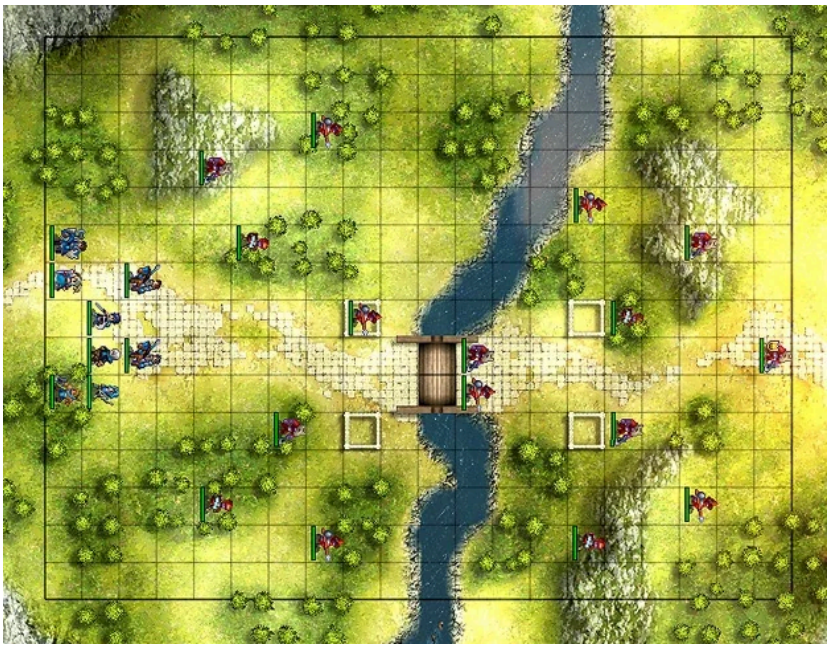
- **Desplazar unidad**
  - Cada unidad puede moverse dentro de su rango de movimiento una sola vez por turno (siempre y cuando no haya atacado antes).
  - Las unidades pueden moverse a través de casillas vacías únicamente, en terreno transitable.
- **Preparar emboscada**
  - Colocar unidades en modo "oculto" en terreno con cobertura
  - El enemigo desconoce de la existencia de esta unidad, hasta que ataque o sea revelada (porque decide moverse/atacar).
- **Atacar o Curar** *(dependiendo del equipamiento)*
  - El jugador *puede* atacar o curar con sus unidades con las siguientes restricciones:
    - Cada unidad puede atacar o curar una sola vez por turno
    - Unidades cuerpo a cuerpo deben estar adyacentes al objetivo
    - Unidades a distancia atacan dentro de su rango
    - Si una unidad no realizara dicha acción durante un turno propio, recibe un boost de **DEF** durante el próximo el turno del contrincante
- **Finalizar el turno**
  - Se reducen los contadores de recarga de habilidades/efectos
  - El turno pasa al oponente

Campo de juego/Tablero

Representado como una **rejilla de casillas rectangulares** en cualquier disposición.

Cada casilla tiene un **tipo de terreno** que influye en el juego y en las estadísticas de la unidad que la ocupa:

- ***Llanura*****:** sin efectos.
- ***Bosque*****:** aumenta el ataque y la magia.
- ***Pantano*****:** dificulta la movilidad, reduciendo la movilidad de las unidades que transiten por allí a 1.
- ***Fuerte/Castillo*****:** una unidad que termina su turno en esta casilla, recupera HP y aumenta la defensa durante el turno del rival.
- ***Área contaminada*****:** una unidad que termina su turno en esta casilla disminuye su HP, si llega a 0 obviamente debería ser eliminada.
- ***Agua, Enredadera o Acantilado*****:** intransitable.



Representación gráfica del tablero

Unidades

Cada unidad representa un personaje con las siguientes características:

- *Nombre*: identificación única.
- *Estadísticas*:
  - **HP:** vida total de la unidad.
  - **ATK:** ataque físico.
  - **DEF:** defensa física.
  - **MGC:** poder mágico, ofensivo o de soporte.
  - **MOV:** cantidad de casillas que puede desplazarse por turno.
- *Equipamiento*: influye en el tipo de daño y alcance del ataque

Cuando una unidad llega a 0 HP, es eliminada del juego y no puede volver a invocarse.

Existirá una unidad con **estadísticas mejoradas** bajo la identificación de **lord**, que puede realizar las mismas acciones que una unidad común, con la diferencia de que al perder la vida, el jugador pierde automáticamente el combate.

Desplazamiento

Las unidades, desde una determinada celda, puede moverse en cualquiera de las 8 direcciones que tiene a su alrededor, mientras sean transitables.

Una unidad se puede desplazar hacia cualquier celda a la cual pueda acceder mediante un camino en el rango de movilidad (**MOV**) que posea, y que el camino esté libre. Durante un camino, no se pueden atravesar terrenos intransitables (como cuerpos de agua, enredaderas, acantilados, o celdas ocupadas por otras unidades), y se puede ver afectado por condiciones de terreno (como el pantano).

Al inicio de un turno, una unidad puede moverse y luego atacar. Pero una vez haya realizado un ataque, ya no podrá desplazarse por el mapa.

Unidades ocultas

Al comienzo de la partida, se pueden colocar unidades en modo oculto. Estas unidades son sólo visibles para el jugador que las posee, y permanecen en este modo hasta que el jugador decide moverla o atacar con ella (dando conocimiento de su existencia). A partir de ese momento, sale de modo oculto y no puede volver a este estado.

Para el contrincante, una unidad oculta se verá como un terreno de enredadera más en el mapa, por lo que será una celda intransitable para sus unidades.

Equipamiento y ataques

El jugador dispone de un arsenal de armas o instrumentos, que determinan el estilo de combate y alcance de ataque de su unidad:

- **Armas cuerpo a cuerpo:** `espadas, hachas o lanzas`. Tienen siempre un RNG de valor 1.
- **Armas de ataque a distancia:** `arco`. El arco es un *arma a distancia*. No *tiene ventaja ni desventaja frente a espadas, hachas o lanzas*.
- **Grimorios:** son antiguos libros que contienen hechizos y conjuros para canalizar magia ofensiva. Se basan en el atributo de **MGC** de la unidad atacante y la atacada para determinar el total de daño que realizará.
- **Báculos:** no dañan al objetivo directamente, sino que se usan para curar a otras unidades o aplicar efectos:
    - **Báculo de curación:** restaura moderadamente el HP de un aliado.
    - **Báculo de sanación:** restaura todo el HP de un aliado.
    - **Báculo de fortaleza:** aumenta temporalmente las defensas de un aliado.

Cada equipamiento tiene un valor de **ATK** o **MGC** que se le suma a la unidad al poseerlo. También, cuentan con un alcance o rango de ataque (**RNG)**, que indicarán a qué distancia máxima debe estar un enemigo para poder ser atacado con dicho. Un **RNG** de 1 indica que solo se pueden atacar enemigos adyacentes.

Cada equipamiento tiene **usos limitados**: tras cierto número de ataques o curaciones, se rompe. Cuando esto sucede, la unidad podrá solo atacar a puño limpio.

Una unidad a puño limpio solo puede atacar a enemigos adyacentes, y su **ATK** pasa a ser su estadística base (sin el boost del equipamiento).

Batalla

Cuando una unidad ataca a otra, se verifica:

- Que la unidad atacada esté dentro del **rango de ataque** del equipamiento de la atacante.
- Que el equipamiento de la unidad atacante tenga usos restantes

Si ambas condiciones se cumplen, se inicia el combate.

Pueden ocurrir distintos escenarios de ataque dado su equipamiento:

- **Ataque cuerpo a cuerpo:** se da cuando el atacante posee un arma física de corta distancia (espada, lanza, hacha o puño limpio), a un enemigo directamente aledaño.
- **Ataque a distancia:** se da cuando un atacante se encuentra lejos de un enemigo y realiza un ataque utilizando un arco o un grimorio.
- **Báculos:** no participan en combate directo. Tienen un RNG de aplicación a todos sus aliados, sin importar su lugar en el campo. Cuando una unidad con báculo ataca, es como si lo hiciese a puño limpio.

La resolución de los ataques se realiza de la siguiente manera:

1. **Verificación de alcance y usos:**
    - Se confirma que el objetivo esté dentro del rango (**RNG**) del equipamiento y que éste posea usos disponibles.
2. **Cálculo del daño:**
   Según el tipo de equipamiento, se determina el daño base:
    - **Ataques físicos:** *ATK (atacante) – DEF (atacado).*
    - **Ataques mágicos:** *MGC (atacante) – MGC (atacado).*
3. **Aplicación del daño y efectos secundarios:**
    - Se resta el daño final del HP del objetivo
4. **Verificación de supervivencia:**
    - Si el HP de la unidad atacada llega a 0, se elimina del campo de batalla.
5. **Reducción de usos:**
    - El equipamiento utilizado pierde un uso. Si los usos llegan a 0, el equipamiento se rompe (descarta).

Detalles de juego

- El juego indica qué bando comienza primero al azar.
- En todo momento debe poder consultarse el estado de la partida: HP de cada unidad (y estadísticas), posición en el tablero, armas equipadas, número de usos restantes, etc.

Requerimientos

> *Importante: En caso de tomar una decisión respecto a una definición ambigua del dominio o los requerimientos, volcarlo en la correspondiente sección de hipótesis del informe.*

Modelo

En un proyecto Java, se debe volcar la representación del modelo del juego, con todas sus características enunciadas en el Dominio, contemplando tanto la lógica básica como sus *corner-cases*.

Jugabilidad

La partida se juega por consola, en una sola computadora; debe permitir a los jugadores realizar todas las acciones descritas anteriormente.

Inicio de partida

Al iniciar, se carga el mapa a utilizar (puede tener dimensiones variables, mientras sea rectangular) con su configuración de terrenos, a partir de un archivo (cada grupo define formato y estructura).

También, se asignan los ejércitos (unidades y equipamientos) desde un archivo de configuración (cada grupo define formato y estructura).

Claridad de la partida

Antes de cada turno, se muestra un resumen del estado de la batalla: estadísticas de unidades, posiciones en el tablero, etc.

Al querer desplazar una unidad, debe ser claro a cuáles celdas puede moverse.

Se debe poder elegir claramente la unidad enemiga que se desea atacar.

Flujo de la partida

Si un jugador selecciona una opción incorrecta, debe ser capaz de “cancelar” y volver a su menú principal de opciones.

Al finalizar cada turno, se alterna el control al otro jugador.

Al detectarse la condición de victoria o derrota, se debe mostrar el mensaje correspondiente.

Anexo: Estado de la partida

Debe poder consultarse en cualquier momento información como:

- Disposición del mapa y ubicación de los elementos:
    - Se deben mostrar en distintos colores los distintos tipos de terreno, y también las unidades de cada banda deben poder diferenciarse claramente.
- Listado de unidades restantes, de ambos bandos.
- HP, armas o instrumentos de cada unidad.
- Número de turno actual, y nombre del jugador al que le corresponde el turno actual.

Anexo: Separación de responsabilidades

Es indispensable para lograr bajo acoplamiento y flexibilidad a la hora de desarrollar el juego que los grupos sean capaces de identificar y establecer barreras de responsabilidades entre el modelo del juego y lo que implica la interacción con el usuario.

Se recomienda basarse en un diseño a lo **MVC** o **MVP**.

Interfaz de usuario

> *Importante: Esta siguiente sección contempla una jugabilidad vía terminal.*
> *Si se quisiera hacer una interfaz gráfica para el juego, debe ser charlado y coordinado previamente con el corrector asignado.*

Los mensajes de información y de error quedan para ser definidos por los alumnos. Se recomienda utilizar una librería como JLine para manejar el input del usuario y mostrar listas de opciones para limitar la cantidad de errores posibles. Como por ejemplo:

1 . A
2 . B
3 . C
Ingrese número de la opción deseada : 1
Seleccionado : A

Recomendación: Tener un módulo que ayude con este tipo de opciones que reciba una lista de opciones y retorna la seleccionada para tener la impresión por consola en un solo lugar.

Anexo: Menús de opciones

Si bien no se obliga a utilizar un esquema de menú específico a seguir, desde el curso sugerimos trabajar con un esquema de submenús para facilitar la usabilidad del juego y también simplificar el armado de acciones a nivel código.

El modelo de submenús plantea que haya un menú principal de opciones de usuario, y que se habilite un nuevo submenú con opciones más específicas para la opción principal elegida.

Por ejemplo, si en un primer momento se elige la opción “Mover unidad”, luego se mostrará un submenú donde cada opción serán aquellas unidades que el jugador tenga desplegada en el mapa y con posibilidad de mover y también otro para determinar en qué casilla posicionarlo. Al elegir una determinada unidad. Cada submenú además debería contar con la posibilidad de regresar al estado/submenú anterior.

Informe

Se espera la realización de un informe que contenga:

- Cómo se compila y corre el programa.
- Qué funcionalidades fueron implementadas.
- Cómo se juega al juego (formas de realizar las acciones del juego, si es que no está claro en la jugabilidad misma).
- Un diagrama de clases UML **completo**, dando las explicaciones que consideren pertinentes respecto a la toma de decisiones en el diseño. Puede ser dividido en múltiples diagramas si lo consideran necesario.
- Al menos un diagrama de secuencia, de algún flujo que consideren interesante. Explicar por qué eligieron mostrar ese flujo.
- Una sección de las hipótesis tomadas durante la realización del trabajo.
- Conclusiones.

Criterios de aprobación

Para que el trabajo práctico se considere aprobado, deberá cumplir con los siguientes requisitos mínimos:

- Cumplir con los objetivos enunciados en los detalles del trabajo, y también con los lineamientos y buenas prácticas del POO.
- Debe haber una correcta división de responsabilidades entre el modelo del juego y la parte interactiva con los jugadores. Se debe utilizar el patrón de arquitectura que separe al modelo de la interacción con el usuario (ej: MVC).

Si bien estos son los requisitos mínimos para la aprobación, un TP se considerará *completo* si cumple con todos los requerimientos descritos en las secciones anteriores. Un TP incompleto, puede no ser considerado para la promoción.