CB100 - Algoritmos y Estructuras de Datos

Cátedra Ing. Gustavo Schmidt - 2er cuatrimestre 2025

Trabajo Práctico 2: Rolgar II

Objetivo

Ampliar el juego desarrollado en el TP1 hacia una versión multijugador en un tablero tridimensional.

En esta versión, participan hasta N personajes en simultáneo (cada uno es un jugador), se introducen cartas de poderes especiales y la posibilidad de generar alianzas temporales entre jugadores.

Enunciado

Definir el **tablero** de tamaño X x Y x Z (ingresado por teclado o fijo al inicio en varios niveles de dificultad), mostrarlo a partir de una interfaz gráfica que se definirá más adelante, donde se vean los casilleros vacíos y un Personaje en el centro.

No todo el tablero es transitable, puede tener casilleros Roca (no transitables), espacios vacíos, rampa, agua. El tablero debe estar implementado con listas.

Ejemplo para 3x3x2 (P = Personaje, . = vacío):

Nivel Z=0

```
. . .
. P .
. . .
```

Nivel Z=1

```
. . .
. E .
. . .
```

Se debe una **Personaje** con atributos:

1) nombre
2) vida (nivel de energía)
3) posX, posY, posZ (posición en el tablero)
4) poderes (coleccion de cartas obtenidas)
5) Vision (cantidad de casilleros que puede ver)
6) Fuerza (nivel de energia que descuenta al atacar)
7) Salud (porcentaje de recuperacion de energia al pasar el turno)

El personaje comienza en el centro del tablero.

Los movimientos

Una vez empezado el juego, el usuario puede mover al personaje con las teclas:

W = atras (y - 1)

S = adelante (y + 1)

A = izquierda (x - 1)

D = derecha (x + 1)

Definir las letras para los demas movimientos: diagonal atras derecha, diagonal atras izquierda, diagonal adelante derecha, diagonal adelante izquierda.

Enemigos

Colocar aleatoriamente M enemigos en el tablero . Al crear un enemigo también se cuenta con nombre, energía y la posición.

El jugador debe buscar los enemigos y contrincantes hasta estar en la misma celda, al encontrarlo el alumno debe definir una forma de pelea para encontrar un ganador.
Al pasar los turnos, el Personaje recupera un % de vida. Al imprimir el tablero, el Personaje solo ve lo que le permite la vision de las casillas alrededor. El jugador gana cuando su personaje eliminó a los demas en el tablero, o pierde cuando se queda sin energía.

En el menú se deben ver las instrucciones del juego.

Cartas de Poder

- Existen **8 tipos de cartas de poder**. Ademas deben agregar 2 con proderes definidos por el grupo.

- Ejemplos:

    1. Aumento de vida
    2. Ataque doble
    3. Escudo (reduce daño)
    4. Teletransportación a otra celda
    5. Robo de carta a otro jugador
    6. Curación de un aliado
    7. Invisibilidad por 1 turno
    8. Doble movimiento

- Las cartas aparecen aleatoriamente en casillas vacías del tablero.

- Cuando un personaje entra en esa celda, la recoge y se guarda en su inventario. El inventario tiene como máximo 10 cartas.

Alianzas

- Los jugadores pueden **formar alianzas temporales** con otros.
- En una alianza, los jugadores:
    - Pueden compartir información de sus posiciones.
    - Pueden ayudarse en combate (ejemplo: atacar en conjunto).
    - Pueden intercambiar cartas de poder.
- Las alianzas son **dinámicas** y se pueden romper en cualquier turno.

Reglas de Visión

- Igual que en TP1, un jugador solo ve un área limitada alrededor de su posición.
- Ahora, la visibilidad abarca las **26 casillas vecinas en 3D** (adyacentes en los 3 ejes).

Condiciones de Victoria

- Un jugador gana si elimina a todos los enemigos y queda como el **último jugador en pie**.

- También se puede definir una condición de victoria compartida si una alianza sobrevive sin rivales.
- Hay que definir las reglas de que al eliminar otro jugador que pasa con sus pertenencias.

Interfaz de usuario

Toda la interfaz de usuario debe estar basada en texto. El menú debe ser muy sencillo y solo se imprime el tiempo demorado por pantalla.

No es necesario que se limpie la pantalla, simplemente escribir el menú cada vez que se necesite. Toda la parte grafica del juego se maneja con bitmaps.

Cuestionario

Responder el siguiente Cuestionario:

1) ¿Qué es un svn?
2) ¿Que es una “Ruta absoluta” o una “Ruta relativa”?
3) ¿Qué es git?

Normas de entrega

Trabajo práctico grupal: 7 persona. El grupo debera definir un nombre

Reglas generales: respetar el Apéndice A.

Se deberá subir un único archivo comprimido al campus, en un link que se habilitará para esta entrega. Este archivo deberá tener un nombre formado de la siguiente manera:

NombreDeGrupo-TP2.zip

Deberá contener los archivos fuentes (no los binarios), los datos personales (padrón, nombre y mail), el informe del trabajo realizado, las respuestas al cuestionario, el manual del usuario y el manual del programador (Todo en el mismo PDF).

La fecha de entrega vence el día viernes 08/11/25 a las 23.59hs por el campus.

Se evaluará: funcionalidad, eficiencia, algoritmos utilizados, buenas prácticas de programación, modularización, documentación, gestión de memoria y estructuras de datos.

Apéndice A

1) Usar las siguientes convenciones para nombrar identificadores.
    a) Clases: Los nombres de clases y structs siempre deben comenzar con la primera letra en mayúscula en cada palabra, deben ser simples y descriptivos. Se concatenan todas las palabras. Ejemplo: Coche, Vehiculo, CentralTelefonica.
    b) Métodos: Deben comenzar con letra minúscula, y si está compuesta por 2 o más palabras, la primera letra de la segunda palabra debe comenzar con mayúscula. De preferencia que sean verbos. Ejemplo: arrancarCoche(), sumar().
    c) Variables y objetos: las variables siguen la misma convención que los métodos. Por Ejemplo: alumno, padronElectoral.
    d) Constantes: Las variables constantes o finales, las cuales no cambian su valor durante todo el programa se deben escribir en mayúsculas, concatenadas por "_". Ejemplo: ANCHO, VACIO, COLOR_BASE.
2) Si el trabajo práctico requiere archivos para procesar, entregar los archivos de prueba en la entrega del TP. Utilizar siempre rutas relativas y no absolutas.
3) Entregar el informe explicando el TP realizado, manual de usuario y manual del programador.

4) Comentar el código. Todos los tipos, métodos y funciones deberían tener sus comentarios.
5) Modularizar el código. No entregar 1 o 2 archivos, separar cada clase.
6) No utilizar variables globales.
7) Si cualquier estructura de control tiene 1 línea, utilizar {} siempre, por ejemplo:

```
for(int i = 0; i < 10; i++) {
      System.out.println( i );
}
```