Proyecto : Bomerman Retro Game

Elaborado por: Benjamin Ortiz Quispe y Steadman Murillo Parrales

Instituto Tecnológico de Costa Rica

IC-1802 Introduccion a la Programacion
Profesor: Diego Mora y Jeff Schmidt

Primer semestre de 2025

1. Introduccion

En esta documentación se ofrece una visión general del proyecto **Bomberman**, el cual tiene como objetivo el desarrollo de un videojuego tipo bomberman con niveles temáticos, enemigos progresivos, sistema de ítems y power-ups, así como pantallas funcionales como inicio, configuración, victoria y derrota. El proyecto destaca por incluir elementos personalizables, habilidades especiales por personaje y un sistema de puntajes y estadísticas. Su importancia radica en la aplicación práctica de programación orientada a objetos, manejo de archivos, diseño de interfaces y lógica de juego. La estructura del documento abarca desde la descripción de pantallas, clases implementadas y atributos, hasta los niveles, enemigos, mecánicas de juego y división del trabajo.

2. Descripcion del Problema

Se pretende que el juego ofrezca al usuario una experiencia completa y ordenada, iniciando con una pantalla principal en la cual se muestra el título y cuatro opciones claramente identificadas: Información, Configuración, Juego y Mejores Puntajes. Desde Configuración, el jugador podrá activar o desactivar la música de fondo, la cual se reproduce de manera aleatoria mediante un sencillo sistema de selección de pistas.

Al elegir la sección de selección de personaje, el usuario encontrará tres avatares distintos, cada uno dotado de una habilidad especial única y su correspondiente tiempo de recarga. Estas habilidades pueden ir desde la facultad de atravesar obstáculos y enemigos, hasta la posibilidad de colocar bombas en cualquier punto del escenario o intercambiar voluntariamente bombas por vida extra y viceversa. Con este diseño, se garantiza que cada partida promueva diferentes estilos de juego y tome en cuenta decisiones tácticas.

Una vez dentro de la partida, se despliegan niveles inspirados en la mecánica clásica de Bomberman, pero con ambientaciones temáticas que ofrecen desafíos variados. El primer nivel, de corte helado, modifica la velocidad del personaje –ya sea por un breve efecto de resbalón o por una ralentización temporal– para simular la sensación de avanzar sobre una superficie gélida. El segundo nivel introduce zonas de trampa, con fuego, minas o áreas venenosas que dañan al jugador al contacto. El tercer escenario sumerge al participante en la penumbra total, donde solo un halo de luz alrededor del personaje revela el entorno, aumentando la tensión. Finalmente, el cuarto nivel está concebido como un enfrentamiento contra un jefe único, un enemigo de patrones complejos que pone a prueba todo lo aprendido.

En cada escenario, el mapa se construye como un salón repleto de muros destruibles e indestructibles. Al demoler bloques, pueden aparecer enemigos, ítems o la llave necesaria para abrir la puerta de salida. La colocación de esta llave y de la puerta se realiza de forma estratégica, con el fin de intensificar la sensación de reto y urgencia.

El sistema de inventario limita al jugador a tres ranuras de ítems, gestionadas con una cola FIFO: al recoger un cuarto objeto, el primero en haber sido obtenido se desecha automáticamente. Entre los ítems disponibles están bombas adicionales, aumentos temporales de velocidad, patrones de explosión mejorados, un escudo que absorbe un golpe y muros protectores de duración limitada. Los power-ups –vida extra o daño incrementado– desaparecen al completar el nivel.

Para fomentar la competitividad, el juego acumula puntos según acciones específicas: 200 por cada enemigo derrotado, 2 por cada muro destruido, 100 por cada ítem recogido y un bono de 500 puntos si el jugador supera el nivel sin morir. Cada vez que se alcanzan 300 puntos, se otorga un punto de mejora que puede invertir en incrementar la cantidad de bombas simultáneas, la velocidad de movimiento o la probabilidad de encontrar ítems.

Los datos de los cinco mejores puntajes se almacenan en un fichero JSON o TXT y se presentan en la sección correspondiente, permitiendo al usuario consultar su posición histórica. Toda la lógica del juego se

organiza en clases bien definidas: Jugador, Enemigo, Jefe, Obstáculo, Bomba, Ítem y Power-Up, así como un módulo de renderizado que dibuja la cuadrícula y actualiza los sprites en pantalla de manera fluida, sin saltos bruscos por casilla.

3. Analisis de Resultados

Pantallas:
Se lograron implementar todas las pantallas originalmente planteadas y se añadieron algunas adicionales que enriquecen la experiencia general del juego. Entre las pantallas desarrolladas se encuentran:

- Menú principal
- Pantalla de opciones (configuración)
- Mejores puntajes
- Créditos
- Game Over
- Pantalla de victoria (Win)
- Pantalla de juego
- Pantalla de guardado (Save Game)
- Pantalla de compra de power-ups

Cada una de estas cumple una función específica dentro del flujo del juego y permite al usuario navegar entre distintas secciones de forma clara y funcional. (con estados técnicamente funciona)

Interfaz de juego (GUI):
En el desarrollo de la interfaz principal durante la partida, algunas ideas originales fueron adaptadas. Por ejemplo, el sistema de inventario basado en FIFO (first in, first out) no fue implementado como se había planteado. En su lugar, se optó por mostrar directamente tres ítems fijos en pantalla, los cuales indican su cantidad respectiva y pueden ser activados con teclas específicas (1, 2, 3). Esta decisión facilitó la implementación y ofreció una experiencia más sencilla al jugador.

Además, se integró una pantalla de compra de power-ups, donde el jugador puede gastar créditos obtenidos a partir de puntos acumulados durante la partida. También se incorporó una mecánica donde, al alcanzar ciertos umbrales de puntaje, se otorgan power-ups adicionales de forma automática.

Niveles y enemigos:
La estructura de los niveles cumplió con la propuesta original. Se desarrollaron distintos escenarios con temáticas variadas: nivel helado, nivel de trampas y nivel de oscuridad. Cada uno introduce mecánicas únicas que modifican la experiencia del jugador, como el deslizamiento en el hielo o la visión limitada en zonas oscuras.

En términos de enemigos, se implementaron tres tipos con comportamientos distintos, lo que genera variedad y desafío en los niveles. Además, se completó el desarrollo del jefe final (boss), que representa una dificultad elevada y requiere habilidad (y suerte) para ser derrotado, cumpliendo con el propósito de cerrar el juego con una batalla más exigente.

Power-ups e ítems:
Los power-ups están disponibles tanto como elementos de recompensa automática por puntaje como a través del sistema de compra. Esto le da una capa adicional de estrategia al juego, ya que el jugador debe decidir cómo invertir sus créditos y qué mejoras priorizar. Los ítems como bombas, velocidad y escudo están presentes y funcionales, y su activación se mantiene ligada al HUD.

Clases y arquitectura interna:
El proyecto logró estructurarse correctamente a nivel de programación orientada a objetos. Se definieron las clases esenciales como Jugador, Enemigo, Boss, Obstáculo, Ítem, PowerUp y Bomba, cada una con sus respectivos atributos y comportamientos. La modularidad del código permite mantener el orden y facilita futuras expansiones del proyecto.

Persistencia de datos:
Se logró implementar el guardado y carga de puntajes altos mediante archivos JSON, lo cual cumple con el objetivo de mantener registros del rendimiento del jugador entre sesiones. La pantalla de "Hi Scores" permite visualizar esta información de manera accesible.

- Musica : Se escogió a ZUN como el principal musico de este juego.

4. Dificultades Encontradas

Orden del proyecto :

Por la semejante magnitud que tiene el proyecto, llevar el orden de los SPRITES, clases, funciones es algo bastante tedioso y frustrante , ya que si no se lleva un orden correcto, entonces todo el proyecto se desordena , y por la magnitud es casi imposible hacer algo. La solución propuesta fue organizar todo en un montón de carpetas, y cada una de ellas conteniendo subcarpetas. En el tema del código, la sección de mas arriba era para definir clases, mientras que la de ajabo d esta eera puras funciones para el juego, y abajo deesta ya como tal procede el juego. Primero los movimientos y las cosas "backend" , después la parte grafica.

Creación e integración de sprites animados
Para que el juego no solo funcionara, sino que también se viera bien, se tomó la decisión de que tanto los personajes como los enemigos tuvieran animaciones. Esto implicó no solo buscar sprites ya hechos (algunos de pago), sino también crear varios desde cero, lo cual llevó bastante tiempo. Además, animarlos e integrarlos correctamente al sistema de movimiento y colisiones fue un reto técnico adicional.
Control de colisiones y movimiento fluido
Una parte compleja fue hacer que el jugador se moviera suavemente por pixeles, en lugar de saltar por casillas como si fuera una matriz. Esto requirió rehacer varias funciones de movimiento y ajustar constantemente las hitboxes para que no se generaran errores visuales ni problemas de colisión con muros, enemigos o ítems.
Gestión de niveles con diferentes mecánicas
Cada nivel tenía mecánicas únicas: hielo, trampas, oscuridad, etc. Implementar estos efectos sin romper la lógica principal del juego fue todo un reto. Hubo que desarrollar condiciones especiales para cada nivel y modificar temporalmente cómo se comportaba el jugador en cada caso. Esto también generó errores al inicio porque ciertas funciones del nivel anterior seguían activas si no se manejaban bien los estados.
Carga y guardado de puntajes
Implementar el guardado de puntajes usando JSON fue más complicado de lo esperado. A veces el archivo

se corrompía, o no actualizaba correctamente si el usuario cerraba el juego sin pasar por el flujo correcto. Para solucionarlo, se añadieron funciones de manejo de errores y validaciones para asegurar que los datos fueran consistentes.

Enemigos con comportamientos personalizados

Crear enemigos con inteligencia básica y comportamientos distintos fue otra fuente de dificultad. Cada enemigo tiene un patrón diferente (seguir en línea recta, perseguir al jugador, disparar), y lograr que todos se comportaran de forma estable y sin errores de colisión o desincronización gráfica requirió bastante prueba y error.

Sincronización entre backend y visual

A veces, las acciones que ocurrían "detrás" (como lógica de movimiento, daño o aparición de ítems) no coincidían perfectamente con lo que se mostraba en pantalla. Esto generaba momentos donde el jugador recibía daño sin verlo venir, o los ítems aparecían en lugares raros. Hubo que ajustar tiempos de actualización y coordenadas cuidadosamente para evitar estos desfaces.

5. Bitacora de Actividades y división de trabajo :

Benjamin		Steadman	
Interfaz/Opciones: Añadió elementos de UI como el botón de scoreboard y etiqueta de volumen de música en el menú de opciones, así como pantallas de game over.	**22 horas**	**Diseño de niveles:** Completó los mapas intermedios y finales (niveles1, 2, 3 y nivel "oscuro"), además del layout del nivel del jefe	**28 horas**
Mecánicas de juego: Desarrolló la lógica de niveles (manejo de llaves y cerraduras para avanzar de nivel) y la mecánica de compras de potenciadores (botones de compra, efectos de mejora)	**39 horas**	**Lógica del boss:** Definió el comportamiento del jefe final, implementando sus ataques (varios patrones de ataque) y ajustando colisiones con el jugador	**22 horas**
Jugador y mejoras: Ajustó atributos del jugador (vida, escudo, recuperación) y gestionó estadísticas (puntos de mejora, visualización de salud máxima)	**21 Horas**	**Elementos de entorno:** Añadió trampas y obstáculos en niveles (por ejemplo, en el nivel 2) y detalles del nivel del jefe	**10 horas**
Sprites : Preparacion y creación (otancion) de todos los sprites del juego. Además de implementaciondes de algunos de estos en player	**17 horas**	**Colisiones y ajustes:** Corrigió bugs de hitboxes y colisiones en jugabilidady afinó atributos de enemigos (por ejemplo, ajustó la fuerza de explosión y la salud del enemigo 3, 2, 1	**42 horas**
Audio y efectos: Integró efectos sonoros (explosión de bomba, curación, compra de power-up) y equilibró parámetros de juego (por ejemplo, fuerza de explosión)	**13 horas**	**Diseño de niveles:** Completó los mapas intermedios y finales (niveles 1, 2, 3 y nivel "oscuro"), además del layout del nivel del jefe	21 horas

Conclusiones

A lo largo del desarrollo de este proyecto, se lograron consolidar múltiples aspectos fundamentales del diseño y programación de un videojuego. Se alcanzaron objetivos clave como la creación de niveles jugables, implementación de enemigos con comportamientos definidos, desarrollo de un jefe final con mecánicas propias, integración de sistemas de puntaje y guardado, así como una interfaz de usuario funcional e intuitiva.

Uno de los mayores logros fue la colaboración efectiva entre los miembros del equipo. Por un lado, se contó con un enfoque sólido en el diseño y la lógica del juego, liderado por "benjaortizq", quien desarrolló elementos centrales como el sistema de guardado, las pantallas de victoria, la lógica de power-ups y el diseño de interfaz gráfica. Por otro lado, "esteman221" se encargó del diseño de niveles, la lógica del jefe final y el ajuste de colisiones, elementos cruciales para la jugabilidad y balance del juego.

Entre las lecciones aprendidas, destaca la importancia de una buena organización del proyecto desde el inicio. La magnitud del juego implicó retos importantes en cuanto a la gestión de recursos, sprites, scripts y funcionalidades, lo cual evidenció la necesidad de mantener una estructura de carpetas ordenada y una documentación constante del avance. Además, se aprendió a identificar errores rápidamente mediante pruebas continuas y a solucionarlos colaborativamente.

Como áreas de mejora, se identificó que una mejor planificación previa de roles y tareas podría haber optimizado los tiempos de entrega. Asimismo, el uso de herramientas de control de versiones más avanzado, como ramas para cada funcionalidad, habría permitido una integración más limpia y menos propensa a conflictos.