Proyecto Final Planteamiento

Integrantes 1: Daniel Cruz

Integrantes 2: Miguel Cacabelos.

Evento escogido: Ataque de la flota británica a Cartagena de Indias.

Profesor de teoría: Aníbal José Guerra.

**Resumen Histórico:**

El juego "Defensa de Cartagena de Indias" está inspirado en el histórico ataque de la flota británica a la ciudad de Cartagena de Indias en 1741. En este evento, el almirante español Blas de Lezo lideró la defensa de la ciudad contra las fuerzas navales británicas.

**Niveles del Juego:**

**Nivel 1: El Asedio Inicial**

En este nivel, el jugador asume el papel de Blas de Lezo, defendiendo la ciudad de Cartagena de los barcos británicos que se aproximan desde el norte. Los barcos enemigos descienden verticalmente hacia la zona inferior del juego, representando el avance de la flota enemiga hacia la ciudad. El jugador debe proteger las murallas de la ciudad utilizando un cañón que dispara verticalmente hacia arriba, emulando la estrategia defensiva utilizada por Blas de Lezo para repeler el primer asalto enemigo.

**Nivel 2: El Contraataque**

En este nivel, la batalla se intensifica, y los barcos británicos comienzan a disparar contra las defensas de la ciudad. Los barcos enemigos se mueven en zigzag descendente, representando el intento de los británicos de sortear las defensas españolas. El jugador debe proteger las murallas de la ciudad de los disparos enemigos mientras sigue utilizando el cañón para repeler los ataques y mantener a raya a los invasores.

**Nivel 3: Enfrentamiento con el Navío jefe**

En el nivel final, el jugador se enfrenta al navío insignia británico, representando el momento culminante del asedio. Este navío es controlado por el jefe enemigo más poderoso y representa la mayor amenaza para la ciudad. El jugador debe utilizar todas las habilidades adquiridas en los niveles anteriores para derrotar al navío jefe y asegurar la victoria para Cartagena de Indias.

**Explicación Mecánica de los Niveles:**

**Nivel 1: Defensa con Cañón**

En este nivel, el jugador controla un cañón que puede disparar proyectiles verticalmente hacia arriba. Debe apuntar con precisión y calcular la trayectoria de los proyectiles para impactar a los barcos enemigos que descienden hacia la ciudad.

**Nivel 2: Estrategia de Protección**

En este nivel, además de controlar el cañón, el jugador también debe manejar la defensa de las murallas de la ciudad contra los disparos enemigos. Se introduce la mecánica de movimientos en zigzag de los barcos enemigos, lo que requiere una mayor habilidad para anticipar y contrarrestar los ataques.

**Nivel 3: Batalla Final**

En este nivel, el jugador se enfrenta a un desafío final contra el navío jefe británico. Debe utilizar todas las habilidades adquiridas en los niveles anteriores para derrotar al enemigo más poderoso y asegurar la victoria para Cartagena de Indias.

**Poderes Especiales:**

**Furia de Lezo**: Este poder especial hace referencia al valiente almirante español Blas de Lezo. Cuando se activa, todos los barcos enemigos en pantalla son eliminados instantáneamente, permitiendo al jugador despejar rápidamente el campo de batalla y ganar ventaja estratégica.

**Reparación de la Muralla:** Este poder permite al jugador recuperar un corazón de vida, proporcionando una segunda oportunidad para continuar la defensa de Cartagena de Indias contra el asedio británico.

Con estos poderes especiales, el juego ofrece nuevas dimensiones estratégicas y tácticas, mientras rinde homenaje al legado histórico de Blas de Lezo y su valiente defensa de la ciudad de Cartagena de Indias.

**Entornos Físicas y Objetivos:**

**Nivel 1: El Asedio Inicial**

Vista: Vista superior en 2D, mostrando la ciudad de Cartagena y el mar circundante desde arriba.

Entorno: La ciudad de Cartagena se extiende a lo largo de la parte inferior de la pantalla, con las murallas defensivas claramente visibles. El mar ocupa la mayor parte de la pantalla, con barcos británicos descendiendo desde la parte superior hacia la ciudad.

Física de los Personajes: Blas de Lezo controla el cañón defensivo, con movimientos fluidos y precisos para apuntar y disparar proyectiles. Los proyectiles siguen una trayectoria balística realista hacia arriba.

Enemigos: Barcos británicos que descienden verticalmente desde la parte superior de la pantalla hacia la ciudad. Tienen diferentes tamaños y velocidades, lo que requiere al jugador ajustar su estrategia de defensa.

Objetivo: Proteger las murallas de la ciudad disparando a los barcos enemigos antes de que lleguen a la parte inferior de la pantalla.

**Nivel 2: El Contraataque**

Vista: Vista superior en 2D, similar al nivel anterior, pero con un mayor énfasis en la interacción entre los proyectiles y los disparos enemigos.

Entorno: La ciudad de Cartagena sigue siendo visible en la parte inferior, pero ahora hay más actividad en la superficie del mar, con disparos cruzando de un lado a otro. Los barcos británicos se mueven en patrones de zigzag hacia abajo, representando un desafío adicional.

Física de los Personajes: Blas de Lezo sigue controlando el cañón defensivo, pero ahora debe esquivar los disparos enemigos mientras sigue disparando proyectiles hacia arriba.

Enemigos: Además de los barcos británicos, se introducen disparos enemigos que representan el contraataque británico. Estos disparos pueden dañar las murallas de la ciudad y restar vida al jugador.

Objetivo: Proteger las murallas de la ciudad tanto de los barcos enemigos como de los disparos enemigos, manteniendo la ciudad a salvo del asedio.

**Nivel 3: Enfrentamiento con el Navío jefe**

Vista: Vista superior en 2D, similar a los niveles anteriores, pero con un enfoque más centrado en el enfrentamiento directo con el navío jefe.

Entorno: La ciudad de Cartagena sigue siendo visible en la parte inferior, pero ahora el navío jefe británico ocupa una posición prominente en el centro de la pantalla. El entorno marino circundante está lleno de acción, con olas agitadas y movimientos intensos de ambos bandos.

Física de los Personajes: Blas de Lezo controla el cañón defensivo con movimientos aún más ágiles y precisos, necesarios para enfrentarse al poderoso navío jefe.

Enemigos: El navío jefe británico es el enemigo principal en este nivel, con una salud significativamente mayor y armamento más poderoso que los barcos comunes. También puede desplegar barcos adicionales para complicar la batalla.

Objetivo: Derrotar al navío jefe británico y asegurar la victoria para Cartagena de Indias, resistiendo su embate final y protegiendo las murallas de la ciudad contra su poderío naval.

**PLANTEAMIENTO POO**

**Clase ReglasJuego**

La clase ReglasJuego se creó para gestionar el flujo general del juego, controlando la transición entre niveles, iniciando y finalizando el juego. Actúa como el controlador principal que coordina las distintas partes del juego.

**Atributos**:

Nivel\* niveles[]: Array de niveles del juego.//PROPENSO A CAMBIOS

int nivelActual: Índice del nivel actual.

**Métodos**:

void iniciarJuego():

Inicia el juego estableciendo el primer nivel.

void cambiarNivel(int nuevoNivel):

Cambia al nivel especificado por nuevoNivel.

Parámetros:

int nuevoNivel: Número del nuevo nivel al que se desea cambiar.

Retorno: Ninguno.

void finalizarJuego():

**Clase Nivel**

La clase Nivel representa un nivel específico del juego. Su responsabilidad incluye la creación y control de los enemigos, la gestión de la interacción del jugador con el entorno, y la actualización del estado del nivel.

**Atributos**:

int numeroNivel: Número identificador del nivel.

Jugador\* jugador: Referencia al objeto Jugador.

Enemigo\* enemigos[]: Array de enemigos en el nivel.

Fisica\* fisica: Referencia al objeto de física.

Métodos:

void iniciarNivel():

Inicializa el nivel, creando enemigos y configurando al jugador.

void actualizar():

Actualiza el estado del nivel en cada frame del juego.

void finalizarNivel():

Limpia y cierra el nivel actual, liberando recursos.

**Clase Jugador**

La clase Jugador controla al personaje principal del juego, Blas de Lezo, gestionando sus acciones como disparar y mover, así como su vida y poderes especiales. También se encarga de la actualización de su animación.

**Atributos**:

Cañón\* cañón: Referencia al objeto Cañón controlado por el jugador.

int vida: Puntos de vida del jugador.

PoderEspecial\* poderes[]: Array de poderes especiales disponibles para el jugador.

Fisica\* fisica: Referencia al objeto de física que maneja el movimiento del jugador.

string animacionActual: Estado actual de la animación del jugador.

**Métodos**:

void disparar():

Controla el disparo del cañón, creando un nuevo proyectil..

void activarPoderEspecial(int indicePoder):

Activa un poder especial del jugador basado en el índice.

int indicePoder: Índice del poder especial a activar.

void recibirDaño(int cantidad):

Reduce la vida del jugador en la cantidad especificada.

Parámetros:

int cantidad: Cantidad de daño a recibir.

void mover(int direccion):

Mueve al jugador en la dirección especificada, actualizando su posición.

Parámetros:

int direccion: Dirección en la que se mueve el jugador.

void actualizarAnimacion():

Actualiza la animación del jugador según su estado actual.

**Clase Enemigo**

La clase Enemigo representa a los barcos enemigos en el juego. Controla su movimiento, disparo, y manejo de daños, además de actualizar su animación constantemente.

**Atributos**:

int vida: Puntos de vida del enemigo.

Fisica\* fisica: Referencia al objeto de física que maneja el movimiento del enemigo.

Disparo\* disparo: Referencia al objeto de disparo del enemigo.

string animacionActual: Estado actual de la animación del enemigo.

**Métodos**:

void mover():

Mueve al enemigo según su patrón de movimiento..

void disparar():

Controla el disparo del enemigo, creando un nuevo proyectil.

void recibirDaño(int cantidad):

Reduce la vida del enemigo en la cantidad especificada.

Parámetros:

int cantidad: Cantidad de daño a recibir..

void actualizarAnimacion():

Actualiza la animación del enemigo según su estado actual.

**Clase Cañón**

La clase Cañón se encarga de controlar el disparo del cañón del jugador, gestionando la munición y la velocidad de disparo de los proyectiles.

**Atributos**:

int munición: Cantidad de munición disponible para el cañón.

int velocidadDisparo: Velocidad a la que se disparan los proyectiles.

**Métodos**:

void disparar():

Controla el disparo del cañón, creando y lanzando un nuevo proyectil.

**Clase PoderEspecial**

La clase PoderEspecial define y gestiona los poderes especiales del jugador, permitiendo su activación y desactivación durante el juego.

**Atributos**:

string nombre: Nombre del poder especial.

int duración: Duración del poder especial en tiempo de juego.

**Métodos**:

void activar():

Activa el poder especial, aplicando sus efectos.

void desactivar():

Desactiva el poder especial, eliminando sus efectos.

**Clase Fisica**

La clase Fisica maneja la física de los movimientos y disparos en el juego. Proporciona métodos para aplicar diferentes tipos de movimientos a los objetos.

**Atributos**:

string tipoMovimiento: Tipo de movimiento aplicado (e.g., "vertical", "zigzag").

int velocidad: Velocidad del objeto en movimiento.

int aceleración: Aceleración aplicada al objeto.

int ángulo: Ángulo de movimiento del objeto (en caso de movimiento parabólico).

int posiciónX: Posición X actual del objeto.

int posiciónY: Posición Y actual del objeto.

**Métodos**:

void aplicarFisica():

Aplica la física correspondiente al objeto según su tipo de movimiento.

void moverVertical():

Mueve el objeto verticalmente, actualizando su posición Y.

void moverZigzag():

Mueve el objeto en zigzag, actualizando su posición X e Y.

void moverParabolico():

Mueve el objeto parabólicamente, actualizando su posición X e Y basándose en un ángulo y aceleración.

void moverMRU():

Mueve el objeto con Movimiento Rectilíneo Uniforme (MRU), actualizando su posición X e Y.

**Clase Disparo**

La clase Disparo define el comportamiento de los disparos en el juego, incluyendo su velocidad, daño y la física que los gobierna.

**Atributos**:

int velocidad: Velocidad del disparo.

Int daño: Cantidad de daño que el disparo inflige.

Física\* Física: Referencia al objeto de física que maneja el movimiento del disparo.

Métodos:

void disparar():

Controla el comportamiento del disparo, aplicando la física correspondiente y actualizando su posición.

**DIAGRAMA UML (SUJETO A CAMBIOS)**