# UNIVERSIDAD DE ANTIOQUIA

1 8 0 3

## Facultad de Ingeniería

# Proyecto Final. Juego

## CARRERA DE LA CORRIENTE

Michael Gómez Rojas
INFORMATICA II | UNIVERSIDAD DE ANTIOQUIA | 2024

A. Introducción

Bienvenidos a un viaje a través del tiempo y la ciencia, donde la historia se encuentra con la tecnología en un momento crucial: la Exposición Mundial de Chicago de 1893. En este escenario vibrante, nos sumergimos en la vida y el trabajo de Nikola Tesla, el genio detrás de la corriente alterna (AC), una innovación que transformó el mundo. En "Carrera de la Corriente", exploraremos este emocionante capítulo de la historia mientras nos enfrentamos a desafíos educativos y entretenidos.

B. Objetivo

En "Carrera de la Corriente", nuestro enfoque es doble: ofrecer un desafío atractivo para los estudiantes de programación avanzada y al mismo tiempo integrar de manera dinámica conceptos de física. A través de este proyecto, buscamos crear una experiencia de aprendizaje inmersiva donde los estudiantes puedan aplicar habilidades de programación en C++ mientras exploran y experimentan con los principios de la física cinemática en un contexto práctico y divertido. El objetivo final es que los estudiantes consoliden su conocimiento en programación y fortalezcan su comprensión de la física, todo ello mientras disfruta de una experiencia de juego única y emocionante.

C. Descripción del juego

"Carrera de la Corriente" es un juego educativo ambientado en la Exposición Mundial de Chicago de 1893. El objetivo del jugador es ayudar a Nikola Tesla a instalar la corriente alterna (AC) en diferentes puntos de la exposición, superando diversos obstáculos utilizando principios básicos de física cinemática.

D. Mecánicas del Juego

- Movimiento y Velocidad:

El jugador controla a Tesla, quien puede moverse a la izquierda y a la derecha y saltar. La velocidad de Tesla es controlable, y la aceleración puede ser modificada por diferentes terrenos (resbaladizo, pegajoso, etc.).

- Lanzamiento de Objetos:

Tesla debe lanzar herramientas o dispositivos eléctricos a ciertos puntos para activar la corriente alterna. Los jugadores deben calcular la velocidad inicial y el ángulo del lanzamiento para acertar en el objetivo, aplicando las fórmulas de movimiento parabólico.

- Caída Libre y Gravedad:

En algunas partes del juego, Tesla debe bajar de plataformas altas. Los jugadores deben gestionar la caída libre y evitar obstáculos en el descenso. La velocidad de caída aumenta uniformemente debido a la gravedad, y hay áreas donde Tesla puede encontrar amortiguadores para reducir la velocidad de caída.

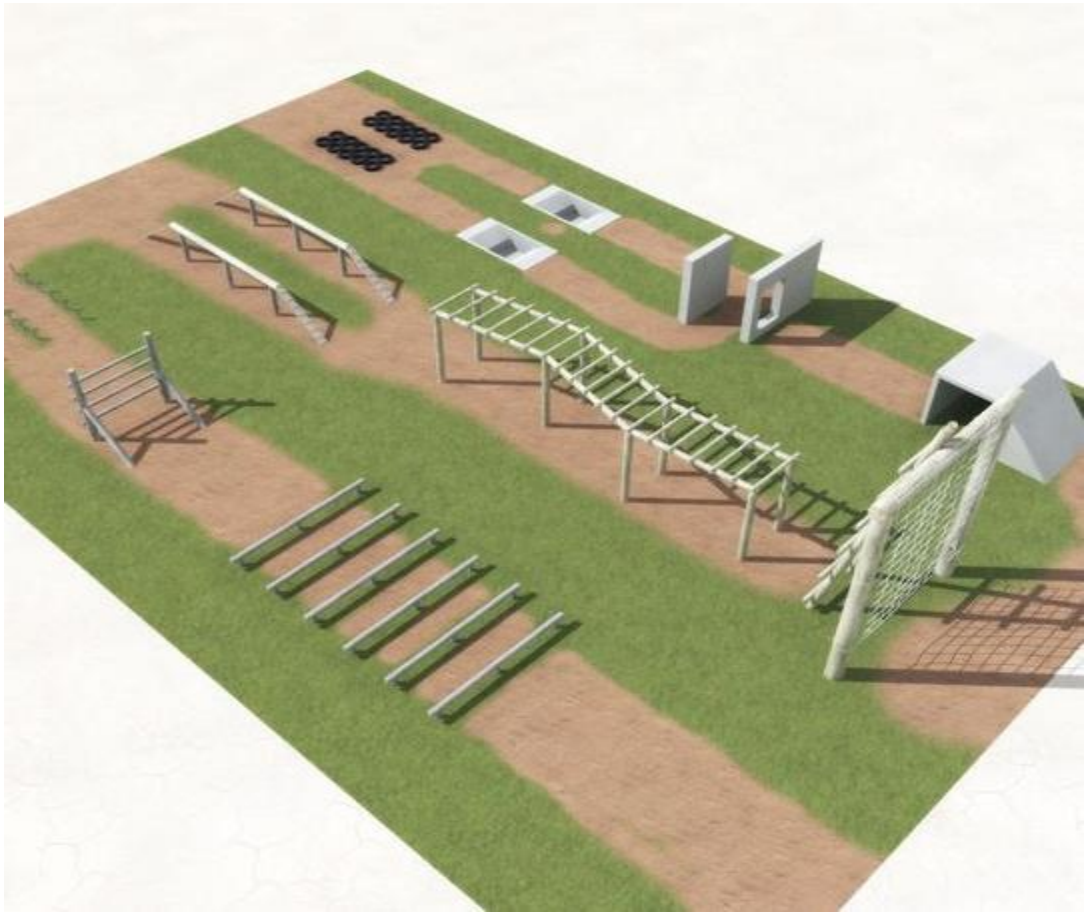

**Figura 1.** Ilustración pista de obstáculos.

E. Objetivos y Niveles:

- **Nivel 1:** Preparación del Terreno

El jugador aprende a moverse y a lanzar objetos.

Obstáculos sencillos y tutoriales sobre cómo calcular la velocidad y el ángulo de lanzamiento.

- **Nivel 2:** Instalación de Dispositivos

Más obstáculos y plataformas en movimiento.

Introducción de terrenos con diferentes fricciones, enseñando sobre aceleración y resistencia.

- **Nivel 3:** Distribución de la Corriente

Desafíos que incluyen caída libre con obstáculos.

Se presentan puzzles que requieren un entendimiento más profundo de la cinemática para resolver.

F. Física Cinemática Incorporada:

- **Movimiento Rectilíneo Uniforme y Uniformemente Acelerado:** Movimiento de Tesla y cálculo de distancias recorridas bajo diferentes condiciones de aceleración.
- **Movimiento Parabólico:** Lanzamiento de herramientas o dispositivos, donde los jugadores deben aplicar conceptos de ángulo, velocidad inicial y aceleración debido a la gravedad para acertar en los objetivos.
- **Caída Libre:** Descenso de plataformas, controlando la velocidad de caída y utilizando amortiguadores para evitar daños.

G. Aspectos Educativos:

El juego introduce y refuerza conceptos de física cinemática de manera práctica y divertida.

Los jugadores aprenden sobre la importancia de la corriente alterna y su impacto en la historia, especialmente en el contexto de la Exposición Mundial de Chicago de 1893.

Se presentan retos que requieren el uso de fórmulas físicas reales para ser superados, promoviendo el aprendizaje activo y la resolución de problemas.

H. Ambientación y Estilo Visual

Escenarios detallados de la Exposición Mundial, con elementos históricos y tecnológicos.

Música de fondo inspirada en la época para crear una atmósfera inmersiva Y reflejar la época de finales del siglo XIX.

"Carrera de la Corriente" combina historia, educación y entretenimiento, ofreciendo a los jugadores una experiencia única y enriquecedora mientras aprenden sobre física y el desarrollo de la electricidad.

I. Diagrama de clases

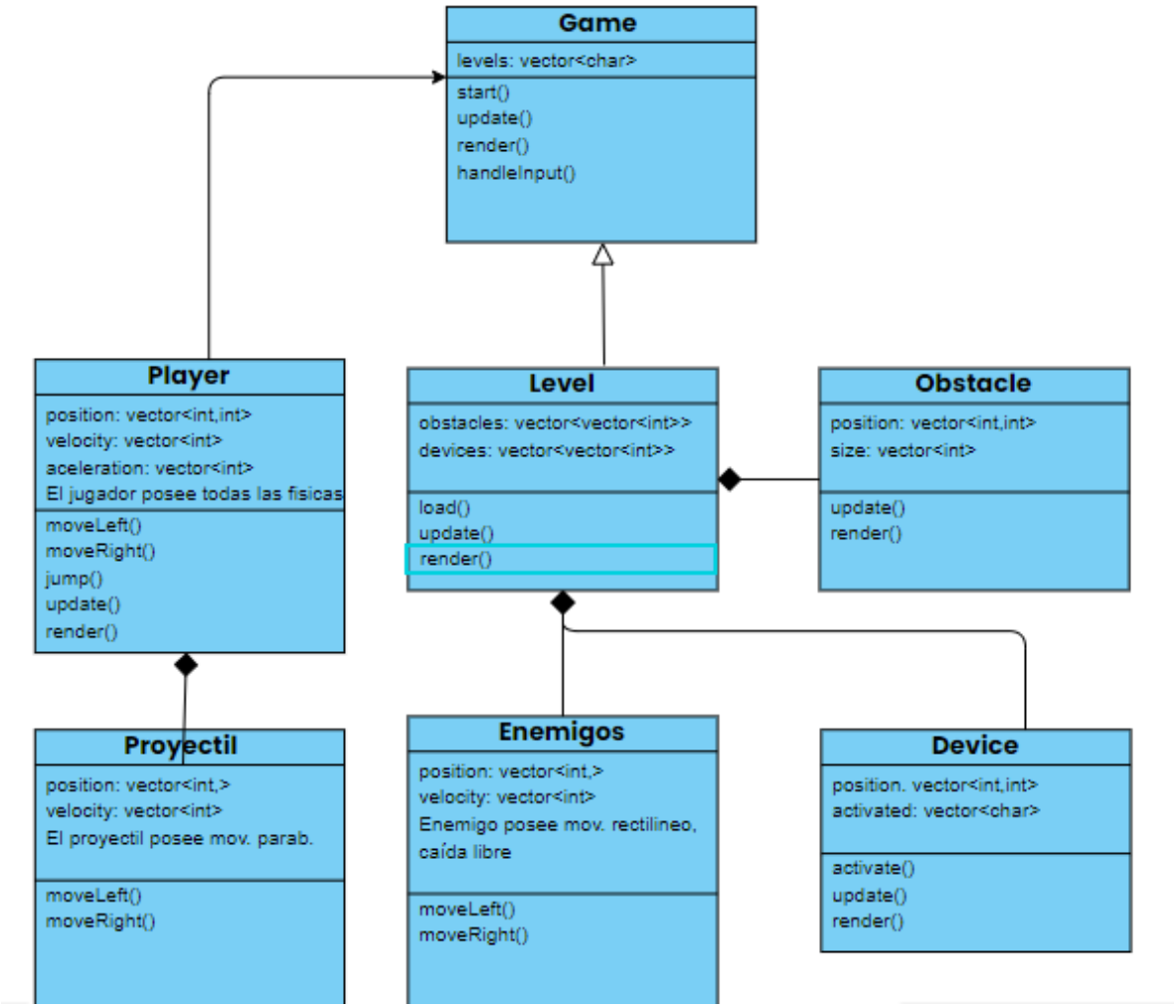


**Figura 2.** Diagrama de clases.

- **Game**

1. Atributos:

levels: Un arreglo (o una estructura de datos similar) que contiene todos los niveles del juego.

2. Métodos

start(): Inicia el juego, cargando el primer nivel y realizando cualquier inicialización necesaria.

update(): Actualiza la lógica del juego en cada fotograma, como la detección de colisiones y la interacción entre los objetos.

render(): Renderiza los gráficos del juego en cada fotograma.

handleInput(): Maneja la entrada del usuario, como las pulsaciones de teclas o los clics del ratón.

- **Level**

1. Atributos:

obstacles: Un arreglo u otra estructura de datos que contiene los obstáculos presentes en el nivel.

devices: Un arreglo u otra estructura de datos que contiene los dispositivos eléctricos que el jugador debe activar.

2. Métodos:

load(): Carga el nivel desde algún archivo o define los obstáculos y dispositivos programáticamente.

update(): Actualiza la lógica del nivel en cada fotograma.

render(): Renderiza los gráficos del nivel en cada fotograma.

- **Player**

1. Atributos:

position: Una estructura de datos (como un vector o un par de coordenadas) que representa la posición actual del jugador en el nivel.

velocity: Una estructura de datos que representa la velocidad actual del jugador.

acceleration: Un valor numérico que representa la aceleración del jugador.

2. Métodos:

moveLeft(): Mueve al jugador hacia la izquierda.

moveRight(): Mueve al jugador hacia la derecha.

jump(): Hace que el jugador salte.

update(): Actualiza la posición y la velocidad del jugador en cada fotograma.

render(): Renderiza el sprite o la representación visual del jugador en cada fotograma.

- **Obstacle**

1. Atributos:

position: Una estructura de datos que representa la posición del obstáculo en el nivel.

size: Una estructura de datos que representa el tamaño del obstáculo.

2. Métodos:

update(): Actualiza la lógica del obstáculo en cada fotograma.

render(): Renderiza el sprite o la representación visual del obstáculo en cada fotograma.

- **Device**

1. Atributos:

position: Una estructura de datos que representa la posición del dispositivo en el nivel.

activated: Una variable booleana que indica si el dispositivo ha sido activado por el jugador.

2. Métodos:

activate(): Activa el dispositivo, cambiando su estado a activado.

update(): Actualiza la lógica del dispositivo en cada fotograma.

render(): Renderiza el sprite o la representación visual del dispositivo en cada fotograma.

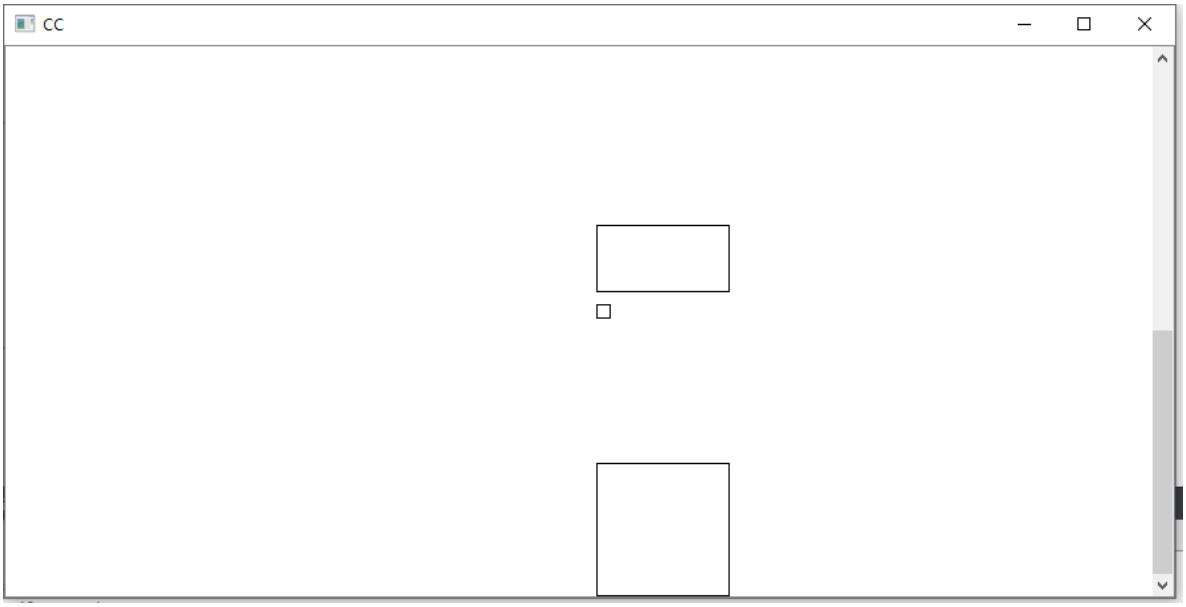


**Figura 3.** Ejecutable, primer boceto.

## J. Conclusiones

## K. Bibliografía

1. Schildt, H. (2017). "C++: The Complete Reference." McGraw-Hill Education.
2. Stroustrup, B. (2013). "Programming: Principles and Practice Using C++." Addison-Wesley.
3. Fu, X., & Liu, J. (2015). "Discrete-Event Simulation: A First Course." Chapman and Hall/CRC.