1. Atributo de Análisis de Problema

a) Identificación del problema complejo de ingeniería.

La elaboración de este proyecto supone el diseño de un sistema interactivo que simula un entorno dinámico (juego) donde un jugador y múltiples agentes autónomos (enemigos) se intentan capturar (según la modalidad de juego) sobre un mapa generado aleatoriamente.

Esto implica el modelado del movimiento en la cuadrícula, decisiones autónomas de los "personajes" (perseguir, huir, escapar), el modelado de las restricciones físicas del entorno (muros, lianas, túneles) y su manera de relacionarse con el resto de los elementos del entorno, sincronización de eventos, diseño de la gestión de energía y las trampas, generación de mapas aleatorios en cada partida, animación y sonido (diseño en general del juego).

b) Análisis del contexto y variables (incluyendo sostenibilidad)

El juego simula un entorno tipo laberinto donde interactúan varios elementos con reglas distintas para relacionarse entre sí. Las variables principales por considerar serían:

Terreno: determina la movilidad posible de cada "personaje".

Velocidad y comportamiento de enemigos: incide sobre la dificultad de la partida.

Energía del jugador: parámetro que limita movilidad acelerada (correr).

Trampas: son el elemento que permite al jugador defenderse de los enemigos.

Mapa aleatorio: evita patrones repetitivos y dinamismo.

Sostenibilidad

Se integra sostenibilidad conceptual al proyecto al:

- Optimizar cálculos: minimizar ciclos innecesarios para reducir consumo de CPU:

  L**a CPU es crítica**, porque:

  Cada enemigo recalcula movimiento, cada ciclo del juego ejecuta lógica, Tkinter usa un loop constante, hay verificaciones de colisión y hay IA básica pero repetitiva.

- Reutilizar objetos e información: respawn de enemigos en vez de crear objetos infinitos.
- Diseñar código mantenible y modular: facilita la escalabilidad.

c) Plan de solución (con principios de ingeniería)

La solución se construyó usando POO. El mapa genera matrices válidas y garantiza un camino con salida. Las clases de terreno establecen restricciones físicas. El jugador gestionará entrada, energía y trampas; los enemigos emplean IA basada en distancia Manhattan para decidir su movimiento; y el controlador del juego sincroniza ciclos, colisiones, respawn y condiciones de finalización. Además, se implementó un sistema de puntajes por modo y una interfaz en Tkinter que actúa como puente de comunicación con el usuario. Se aplican principios de ingeniería como modularidad (mapa.py,

terreno.py, juego.py etc), encapsulamiento, abstracción y separación de responsabilidades (jugador, enemigo, GUI).

d) Evaluación de pros y contras

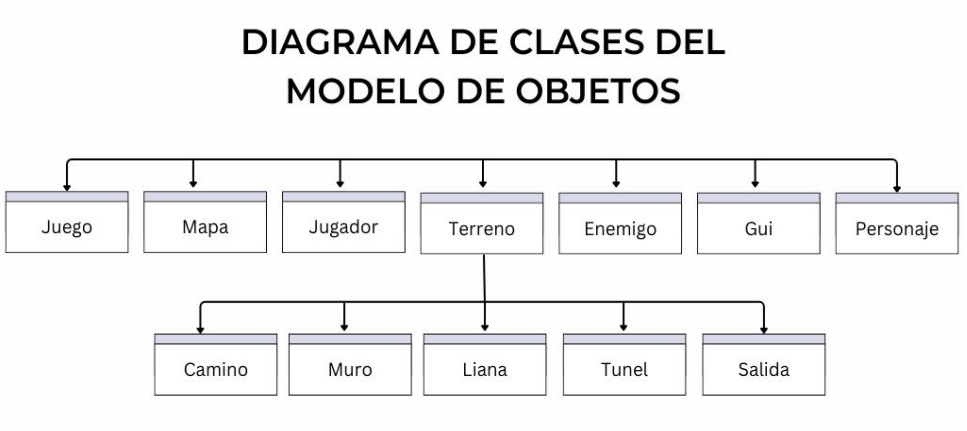
Entre los beneficios destacan la facilidad para extender o modificar partes del sistema gracias a la POO, y la eficiencia de la IA simplificada. Como desventajas, encontramos que los mapas aumentan la complejidad. Tkinter además, limita las animaciones posibles y el ciclo del juego requiere control estricto para evitar retrasos ("lag").

2. Atributo de Herramientas de Ingeniería

a) Técnicas, herramientas y métodos usados

Se utilizaron clases para modelar entidades del sistema, matrices para representar el entorno, distancia Manhattan para comportamiento enemigo, Tkinter para la interfaz, archivos TXT para persistencia de datos, generación procedural para mapas aleatorios y el sistema de eventos de Tkinter para coordinar acciones en tiempo real.

b) Diagrama de clases del modelo de objetos



https://www.canva.com/design/DAG6C3I5Uxs/Z8VbTqO9M-AzefQ2BBJI7w/view?utm_content=DAG6C3I5Uxs&utm_campaign=designshare&utm_medium=link2&utm_source=uniquelinks&utlId=h11754b02b8

c) Cómo aplicaste esas herramientas

La matriz del mapa permite identificar el tipo de casilla en tiempo constante; Tkinter gestiona eventos y pantallas finales; el archivo TXT almacena puntajes entre ejecuciones; y la POO facilita ampliar el sistema sin modificar la base existente.

d) Cómo las adaptaste durante el proyecto

Se reforzó la separación entre lógica y presentación para cumplir con la modularidad. Se ajustó la IA enemiga para evitar errores de movimiento, se implementó un sistema de, se adaptó el generador de mapas para garantizar caminos válidos y se desarrolló un sistema de puntajes robusto usando únicamente listas.