Documentación del Proyecto - Escapa del Laberinto

Introducción a la programación - ITCR

Estudiantes:

Camilo Jose Solis Gonzales – 2019048742

Johan Molina Vaglio – 2025094364

1. Análisis del Problema

1.1 El Problema que Resolvimos

Bueno, el problema principal era hacer un juego que funcione bien y no se trabe. Teníamos que crear un laberinto que cambie cada vez que juegues, que los malos te sigan de forma inteligente, y que todo se vea bonito sin que la compu se ponga lenta.

Usamos matemáticas para calcular distancias y para hacer que el laberinto se genere solo. También tuvimos que pensar en física básica para que los movimientos se vean naturales y las colisiones funcionen bien.

1.2 Cómo Lo Analizamos

El contexto era así:

- El laberinto tiene que ser diferente cada vez, pero siempre tener salida
- Los cazadores no pueden ser muy tontos ni muy inteligentes (que sea justo)
- El sistema de puntos tiene que premiar a los buenos jugadores

Las variables importantes:

- Qué tan grande es el laberinto (al final usamos 21x21)
- Qué tan rápido se mueven los personajes
- Cuánta energía tiene el jugador para correr
- Cuánto tiempo tardas en escapar

Lo de desarrollo sostenible:

- El juego no gasta mucha batería

- Cualquier compu lo puede correr, no necesita ser gamer
- Ayuda a pensar y planear, como ejercicio mental

1.3 Cómo Lo Solucionamos

Hicimos tres cosas principales:

1. **El laberinto que se hace solo:**
    - Usamos un algoritmo que va "cavando" caminos aleatorios
    - Le pusimos zonas especiales donde solo puede pasar el jugador o solo los cazadores
    - Nos aseguramos que siempre haya camino a la salida
2. **Los cazadores inteligentes:**
    - En modo escapar, te buscan y te siguen
    - En modo cazador, huyen de ti
    - Si se pierden, buscan otro camino
3. **Cosas para que el juego sea bueno:**
    - No puedes correr todo el tiempo, tienes energía limitada
    - Las minas tienen cooldown para que no las spamees
    - Los puntos premian si escapas rápido

1.4 Qué Salió Bien y Qué No

Lo bueno:

- ✅ El laberinto siempre tiene salida (probamos muchas veces)
- ✅ Los cazadores no se atoran en las paredes
- ✅ El sistema de puntos es justo
- ✅ Corre bien en computadoras normales
- ✅ Es fácil de entender cómo jugar

Lo malo:

- ⚠️ A veces los cazadores se demoran en encontrarte

- ⚠️ El laberinto a veces se ve parecido
- ⚠️ No hay muchos tipos diferentes de cazadores

Lo de sostenibilidad:

- ✅ No gasta mucha luz
- ✅ Lo puedes jugar muchas veces sin aburrirte
- ✅ Te hace pensar y planear tus movimientos

---

2. Herramientas que Usamos

2.1 Con Qué Hicimos el Juego

Lenguajes y programas:

- **Python**: Porque es fácil de aprender y usar
- **Pygame**: Para hacer los gráficos y que se vea como juego de verdad
- **PIL**: Para que las explosiones se vean bonitas.

Algoritmos que programamos:

- **Hacer laberintos**: Un algoritmo que va quitando paredes al azar
- **Buscar caminos**: Para que los cazadores sepan por dónde ir
- **Chocar cosas**: Para que no pases through las paredes
- **Explosiones**: Partículas que salen cuando explota una mina

Cómo organizamos el código:

- Cada cosa en su propia clase (jugador, cazador, mina, etc.)
- Diferentes pantallas para menú, juego, puntajes
- Todo separado para que sea fácil cambiar una cosa sin romper las otras

2.2 Diagrama de Clases (Cómo Está Organizado el Código)

JUEGO PRINCIPAL

├── Pantallas

│ ├── Menú (donde eliges modo de juego)

│ ├── Registro (donde pones tu nombre)

│ ├── Juego (donde juegas de verdad)

│ └── Puntajes (donde ves los mejores)

│

├── Personajes

│ ├── Tú (el jugador)

│ │ ├── Donde estás en el mapa

│ │ ├── Qué tan rápido te mueves

│ │ ├── Cuánta energía tienes

│ │ └── Cómo te mueves

│ │

│ ├── Cazadores (los malos)

│ │ ├── Donde están

│ │ ├── Si están vivos o muertos

│ │ ├── Si te persiguen o huyen de ti

│ │ └── Cómo encuentran caminos

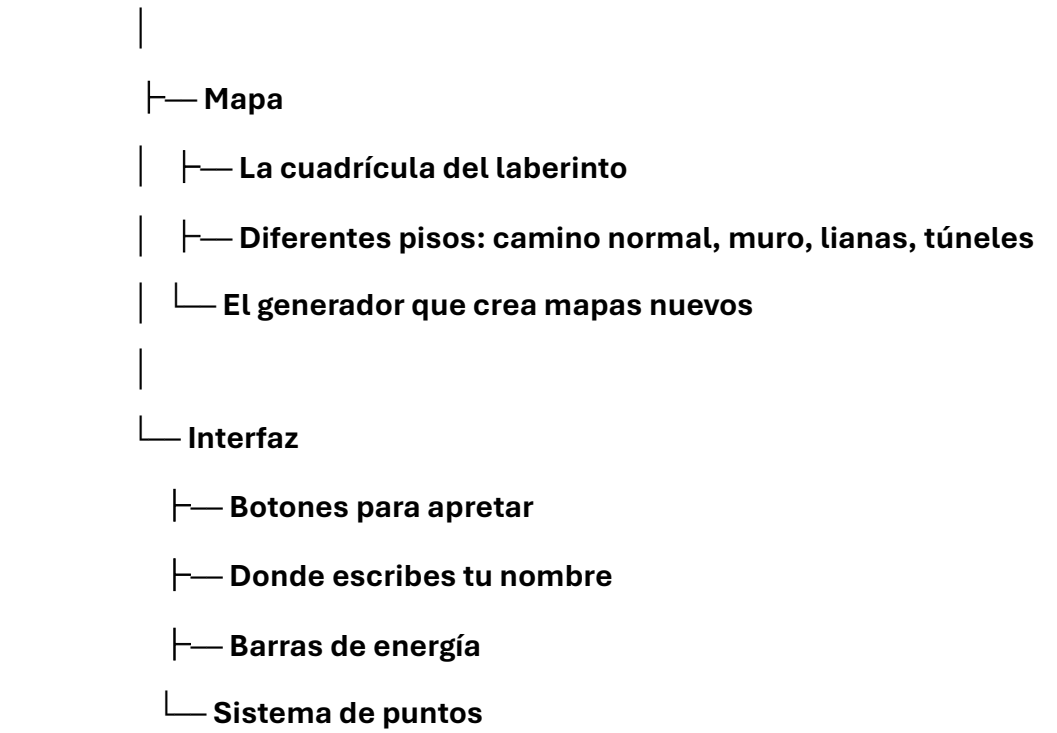
│ │

│ └── Minas (las trampas)

│ ├── Donde están puestas

│ ├── Si están activas o explotando

│ └── Cómo se ven cuando explotan

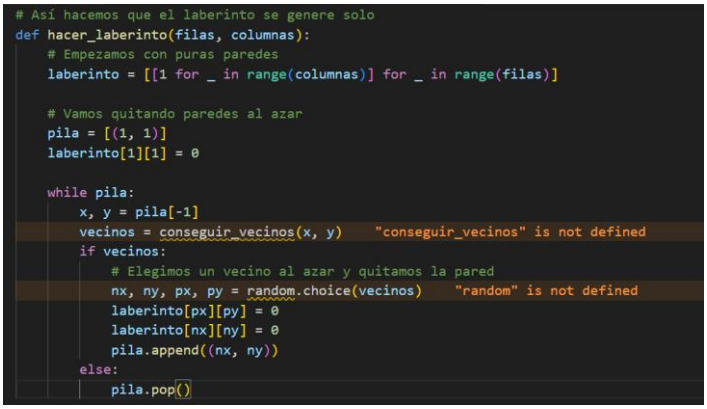


Cómo se conecta todo:

- El juego principal controla todas las pantallas
- Los personajes viven en el mapa y se mueven por él
- La interfaz muestra lo que está pasando en el juego
- El sistema de puntos guarda tus mejores marcas

2.3 Cómo Usamos Esas Herramientas

Para hacer el laberinto:

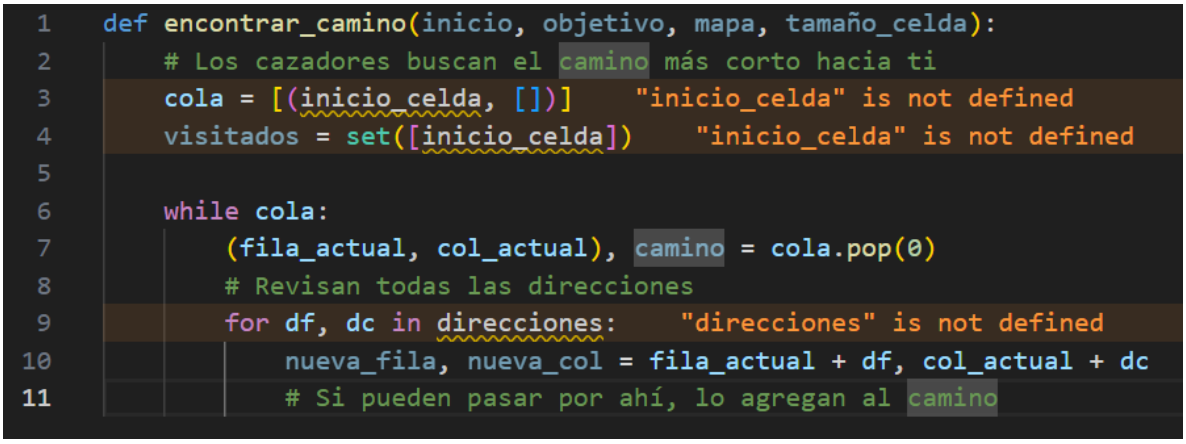


```
# Así hacemos que el laberinto se genere solo
def hacer_laberinto(filas, columnas):
    # Empezamos con puras paredes
    laberinto = [[1 for _ in range(columnas)] for _ in range(filas)]

    # Vamos quitando paredes al azar
    pila = [(1, 1)]
    laberinto[1][1] = 0

    while pila:
        x, y = pila[-1]
        vecinos = conseguir_vecinos(x, y)    "conseguir_vecinos" is not defined
        if vecinos:
            # Elegimos un vecino al azar y quitamos la pared
            nx, ny, px, py = random.choice(vecinos)    "random" is not defined
            laberinto[px][py] = 0
            laberinto[nx][ny] = 0
            pila.append((nx, ny))
        else:
            pila.pop()
```

#NOTA: Se ven errores de variables no definidas porque solo tomamos la parte del código que íbamos a pegar en el documento y lo pusimos en un archivo independiente para hacer mas cómoda la captura de pantalla.

Cabe aclarar que la función no esta completa en la imagen, es solo la parte que hace el laberinto.

Para que los cazadores te encuentren:



```
1   def encontrar_camino(inicio, objetivo, mapa, tamaño_celda):
2       # Los cazadores buscan el camino más corto hacia ti
3       cola = [(inicio_celda, [])]    "inicio_celda" is not defined
4       visitados = set([inicio_celda])    "inicio_celda" is not defined
5
6       while cola:
7           (fila_actual, col_actual), camino = cola.pop(0)
8           # Revisan todas las direcciones
9           for df, dc in direcciones:    "direcciones" is not defined
10              nueva_fila, nueva_col = fila_actual + df, col_actual + dc
11              # Si pueden pasar por ahí, lo agregan al camino
```

2.4 Cosas que Cambiamos Mientras Hacíamos el Juego

Problemas que tuvimos y cómo los arreglamos:

1. **Los cazadores se trababan:**
    - Antes: Recalculaban camino en cada frame (muy lento)
    - Ahora: Solo recalculan cada cierto tiempo o si te pierden
2. **El juego se ponía lento:**
    - Antes: Cargaba todas las imágenes al inicio
    - Ahora: Las carga cuando las necesita y las guarda en memoria
3. **El juego era muy fácil o muy difícil:**
    - Probamos diferentes velocidades para los cazadores
    - Ajustamos cuánta energía gastas al correr
    - Cambiamos cuántas minas puedes poner a la vez

Cosas que simplificamos:

- Usamos búsqueda simple en lugar de algoritmos complicados
- La física es básica, no realista
- Los terrenos son de sí o no (no hay medio terreno)

Cómo sabíamos qué arreglar:

- Poniendo mensajes en la consola para ver qué estaba pasando
- Probando el juego una y otra vez
- Pidiendo a amigos que jugaran y nos dijeran qué no entendían

Info Extra Técnica

Qué Necesita tu Computadora

- **Python 3.8 o más nuevo**
- **Pygame 2.0 o más nuevo**
- Cualquier compu del último... no sé, 10 años? jaja

Qué Tan Bueno Quedó

- **Velocidad**: Corre suave, como a 60 FPS
- **Código**: Está organizado y con comentarios
- **Para agregar cosas**: Es fácil poner nuevos modos o personajes
- **Para jugar**: Es intuitivo, no necesitas leer manual

Por Qué Es "Sostenible"

- No necesita compu gamer
- No gasta mucha batería
- Lo puedes modificar y mejorar fácilmente
- Te enseña a pensar en pasos y planear

Este documento es para el Proyecto 2 del curso de Introducción a la Programación del TEC, segundo semestre del 2025. Ojalá les guste nuestro juego!