# Trabajo Práctico N° 1 - Programación de Videojuegos I

## Fase de análisis:

Historia de usuario HO01:
**Snake**

**Quién:** Como jugador.
**Qué:** Quiero mover la serpiente por el tablero usando las teclas de dirección.
**Para qué:** Para poder crecer al comer animales y mantener el juego en marcha.

**Tabla EPS**

| Entrada | Proceso | Salida |
|---|---|---|
| Pulsación de teclas (↑, ↓, ←, →) | Actualizar la dirección de movimiento de la serpiente. | Cambio de dirección en pantalla. |
| Posición actual de la cabeza de la serpiente | Calcular nueva posición sumando la dirección. | La cabeza se mueve en el tablero. |
| Lista de posiciones de la serpiente | Actualizar cuerpo siguiendo a la cabeza. | Serpiente se desplaza con todo su cuerpo. |
| Colisión con sí misma o bordes | Verificar condiciones de pérdida. | Fin del juego si hay colisión. |

Historia de usuario HO02:
**Animales**

**Quién:** Como jugador.
**Qué:** Quiero que aparezcan animales (comida) en el tablero.
**Para qué:** Para que la serpiente pueda comerlos, crecer y aumentar el puntaje.

**Tabla EPS**

| Entrada | Proceso | Salida |
|---|---|---|
| Posición aleatoria dentro del tablero | Generar un nuevo objeto Animal en esa posición. | Un animal aparece en pantalla. |
| Posición de la cabeza de la serpiente | Verificar si coincide con la del animal. | Detección de colisión con comida. |
| Colisión detectada | Destruir al animal y notificar al Snake que crezca. | Serpiente aumenta su tamaño y puntaje sube. |

| | | |
|---|---|---|
| Estado del tablero | Comprobar que no se genere un animal en la misma posición de la serpiente. | Aparición válida de nuevos animales. |

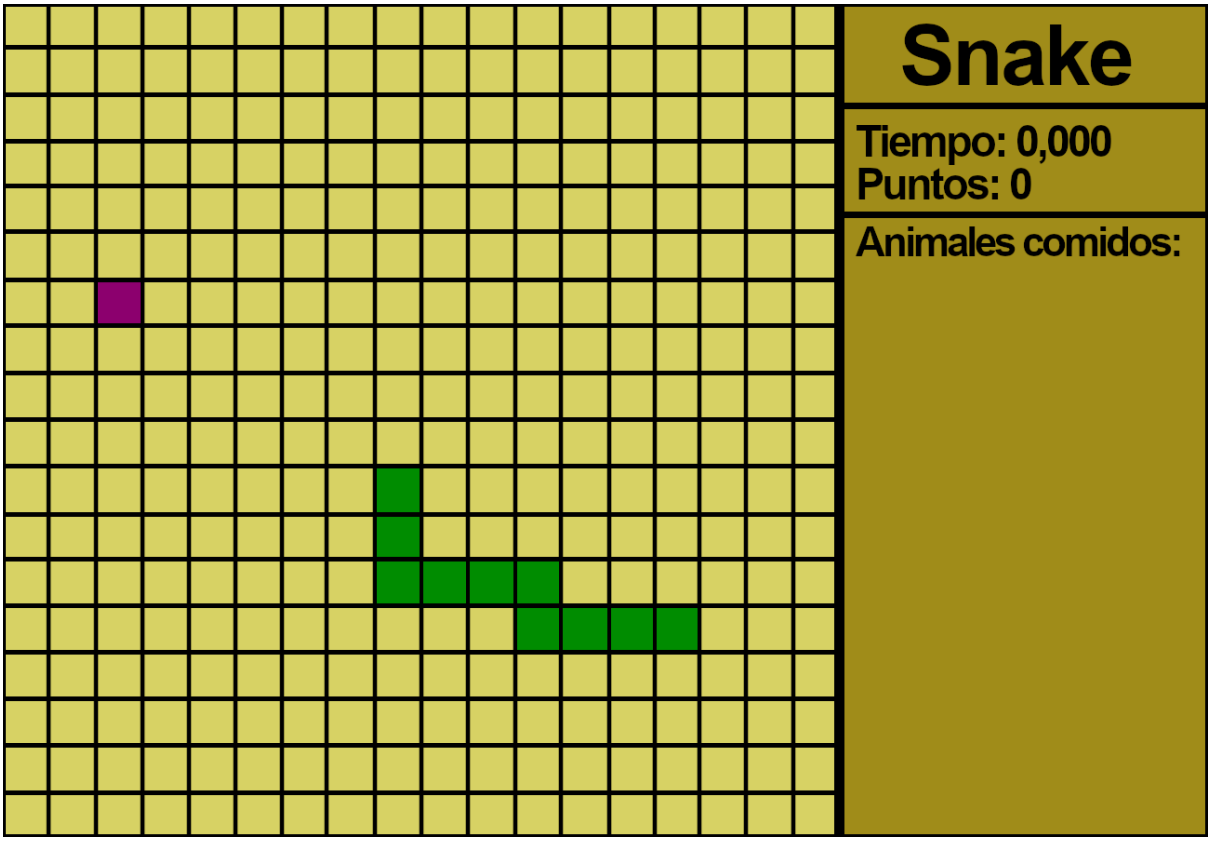
Historia de usuario HU03:
**HUD**

**Quién:** Como jugador.
**Qué:** Quiero ver mi puntaje y estado del juego en pantalla.
**Para qué:** Para saber cuántos puntos tengo y cuándo he perdido.

**Tabla EPS**

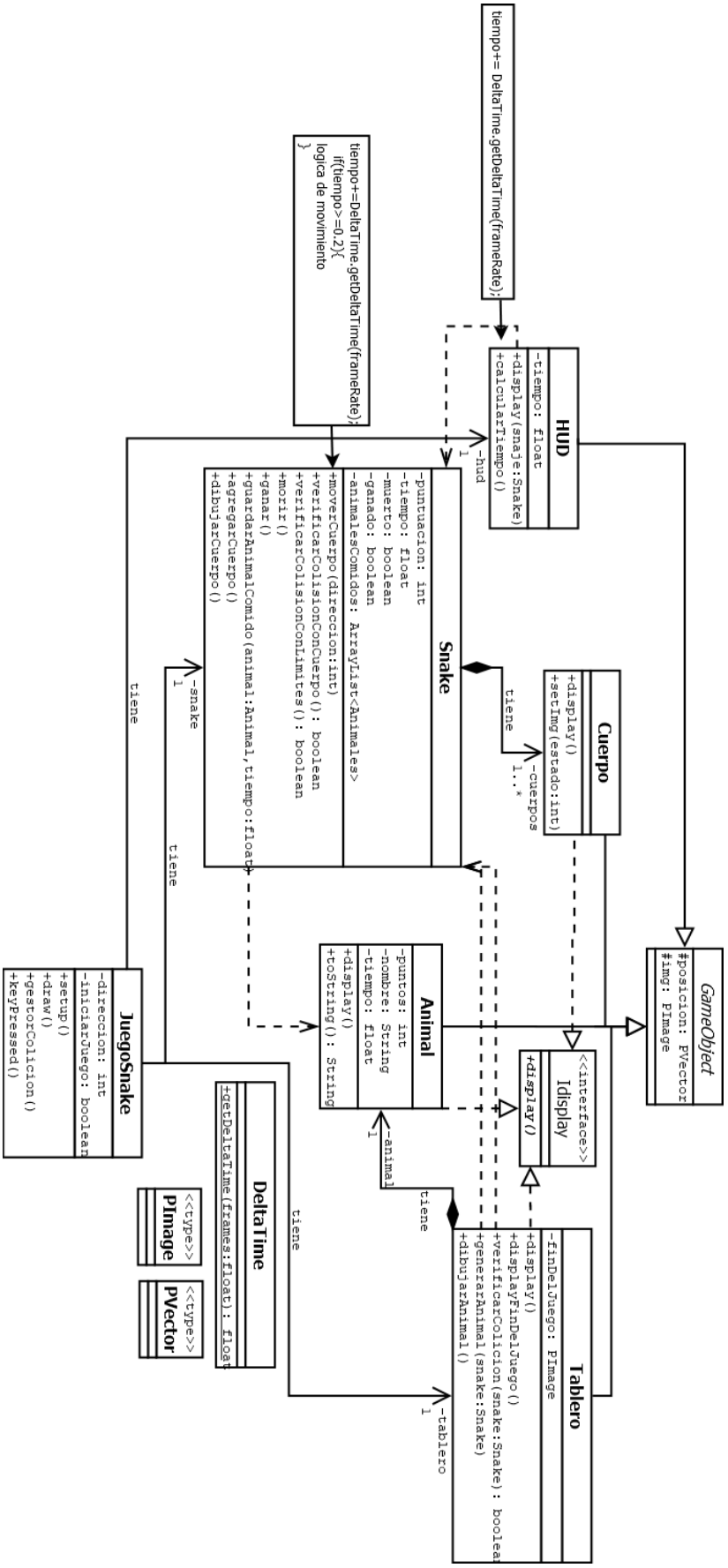
| Entrada | Proceso | Salida |
|---|---|---|
| Variable de puntaje del juego | Actualizar texto en pantalla. | Puntaje visible al jugador. |
| Estado del juego (jugando / game over) | Determinar qué mensaje mostrar. | “Game Over” o puntaje final. |
| Posición en pantalla | Dibujar HUD en el lugar definido (esquinas, centro). | Información siempre visible al jugador. |

**Boceto:**



## Fase de diseño:

Diagrama de clases:



## HU01: Snake

**Objetivo central:** Implementar la serpiente que se mueve en el tablero y crece con nuevos segmentos (Cuerpo).

## Lista de Tareas

- ☐ Diseñar la estructura de la serpiente con cabeza y lista de segmentos de cuerpo.
- ☐ Implementar el movimiento de la cabeza según la entrada del teclado.
- ☐ Hacer que los segmentos sigan la posición de la cabeza (actualizar lista de Cuerpo).
- ☐ Implementar detección de colisión contra sí misma y contra los bordes del tablero.
- ☐ Probar en ejecución que la serpiente responde al input y se desplaza correctamente.

## Criterios de Aceptación

- ☐ El jugador debe poder mover la serpiente en las 4 direcciones cardinales con el teclado.
- ☐ La serpiente debe moverse de forma continua, manteniendo la coherencia entre cabeza y cuerpo.
- ☐ Al chocar con su propio cuerpo o con los bordes, el juego debe terminar.
- ☐ Al crecer, los nuevos segmentos deben añadirse correctamente al final de la serpiente.

## HU02: Animales

**Objetivo central:** Implementar animales como elementos recolectables que hacen crecer la serpiente y aumentan el puntaje.

## Lista de Tareas

- ☐ Diseñar la clase Animal con atributos de posición y tamaño.
- ☐ Implementar generación de animales en posiciones aleatorias del tablero.
- ☐ Verificar que los animales no aparezcan sobre la serpiente (ni su cabeza ni su cuerpo).
- ☐ Detectar colisión entre la cabeza de la serpiente y un animal.
- ☐ Al producirse la colisión, eliminar el animal y notificar al Snake que debe crecer y sumar puntaje.
- ☐ Generar un nuevo animal tras cada recolección.

## Criterios de Aceptación

- ☐ Siempre debe haber un animal visible en el tablero durante la partida.
- ☐ Cuando la serpiente toca un animal, este desaparece.
- ☐ La serpiente debe crecer en longitud al comer un animal.
- ☐ El puntaje debe aumentar en 1 (o la cantidad definida) al recoger un animal.

## HU03: HUD

**Objetivo central:** Mostrar al jugador información sobre el estado del juego (puntaje y mensajes clave).

## Lista de Tareas

- ☐ Diseñar la clase HUD con capacidad de mostrar puntaje y estado del juego.
- ☐ Implementar actualización visual del puntaje cada vez que se recolecta un animal.
- ☐ Implementar un temporizador que inicie junto con el juego y que limite el tiempo del juego a 60s
- ☐ Mostrar mensajes en pantalla al inicio y al final de la partida (ejemplo: “Game Over”).
- ☐ Definir posiciones fijas para los elementos del HUD (ejemplo: esquina superior para puntaje, centro para mensajes).
- ☐ Probar que el HUD se actualice en tiempo real mientras se juega.

## Criterios de Aceptación

- ☐ El HUD debe mostrar siempre el puntaje actualizado del jugador.
- ☐ Debe mostrarse un mensaje claro cuando el juego termina.
- ☐ El HUD no debe interferir con la jugabilidad (no tapar la serpiente ni los animales).
- ☐ La información debe ser legible en todo momento.
- ☐ Si el contador llega a 60s el juego termina.