La Culebrita

Mateo Vergara Gonzalez

Jesús Gonzalez Guzmán

Instrucciones del Juego

El juego consiste en controlar una serpiente que se desplaza dentro de un tablero, comiendo comida **(*)** para crecer y ganar puntos. El objetivo es evitar que la serpiente choque contra los bordes del tablero o contra su propia cola.

Controles

Teclas para controlar la serpiente:

- W: Mover hacia arriba
- A: Mover hacia la izquierda
- S: Mover hacia abajo
- D: Mover hacia la derecha
- Q: Salir del juego

Arquitectura del Juego

El juego está programado en C++ y consta de varios elementos que interactúan entre sí. A continuación, se describe cómo se han diseñado y programado los elementos principales.

Elementos Principales

1. **Tablero:** Representa el espacio de juego. Tiene un ancho **(W)** y una altura **(H)** definidos como constantes. Se dibuja utilizando caracteres en la consola, con bordes representados por ¨#¨ **.**
2. **Serpiente :** Está formada por una cabeza ¨**O**¨ y una cola ¨**o**¨. Su posición se almacena en dos arreglos: ***headPos*** para la cabeza y ***tailPosX***, ***tailPosY*** para las partes de la cola. La longitud de la cola se controla con ***tailLenght***.
3. **Comida**: Representada con el carácter ¨ ***** ¨. Su posición es aleatoria dentro del tablero y se almacena en un arreglo llamado **foodPos**.
4. **Puntuación**: Se almacena en la variable **score**, que se incrementa cada vez que la serpiente come comida.

Interacción entre los Elementos

1. **Movimiento de la serpiente:** La cabeza de la serpiente se mueve en la dirección seleccionada por el jugador. La cola sigue la posición de la cabeza para simular el movimiento.
2. **Colisiones:** El juego termina si la cabeza de la serpiente choca contra los bordes del tablero o contra alguna parte de su propia cola.
3. **Interacción con la comida:** Si la cabeza de la serpiente coincide con la posición de la comida, se aumenta la puntuación, la longitud de la cola crece y se genera una nueva comida en una posición aleatoria.

Breve Explicación de Arrays

Un array es una estructura de datos que son utilizados para almacenar múltiples valores en una única variable. bajo un mismo nombre, organizados en posiciones consecutivas. Cada elemento del array se accede usando un índice numérico.

Declaracion:

tipo_de_dato nombre_del_vector[tamanio];

Por ejemplo, en este juego usamos ***¨headPos[2]¨*** para almacenar las coordenadas (x, y) de la cabeza de la serpiente. El primer elemento (headPos[0]) contiene la posición en el eje X y el segundo (headPos[1]) la posición en el eje Y.

Método render

Este método es responsable de dibujar el tablero, la serpiente, la comida y mostrar la puntuación en pantalla. Todo se dibuja de manera dinámica en la consola.

1. **Bordes del tablero**:
    a. Dibuja líneas horizontales superiores e inferiores usando # para los bordes.
    b. También coloca bordes verticales a los lados del área de juego.
2. **Elementos dentro del tablero**:
    a. **Cabeza de la serpiente**: Representada por O, su posición está controlada por headPos que almacena sus coordenadas de la matriz.
    b. **Cola de la serpiente**: Dibujada como una secuencia de o, sus posiciones están en los arreglos tailPosX y tailPosY.
    c. **Comida**: Representada por *, su posición está definida por foodPos.
3. **Optimización**:
    a. Utiliza SetConsoleCursorPosition para actualizar la pantalla sin parpadeos, moviendo el cursor al inicio de la consola antes de redibujar.

Método input

Este método lo que básicamente hace es que nos ayuda a obtener por teclado las teclas que oprime el jugador dándole un significado a cada una.

Detalles:

1. **Captura de teclas**:
    a. Se usa la función _**kbhit()** para verificar si una tecla fue presionada y _**getch()** para capturarla.
2. **Direcciones**:
    a. Cambia la dirección de la serpiente basándose en la tecla presionada:

        W: Mover hacia arriba.

        A: Mover hacia la izquierda.

        S: Mover hacia abajo.

        D: Mover hacia la derecha.

    b. Si el jugador presiona Q, el juego termina.
3. **Velocidad**:
    a. Se utiliza **Sleep(8)** para ajustar el tiempo de respuesta en ciertos movimientos (como arriba y abajo).

Método gameLogic

Este método es aquel que contiene toda la lógica del juego, como el movimiento de la serpiente, la detección de colisiones y la interacción con la comida.

Detalles:

1. **Movimiento de la cola**: Se usa un sistema de variables previas **(prevTailPosX y prevTailPosY)** para "arrastrar" las posiciones de las partes de la cola, haciendo que sigan la cabeza.

2. **Movimiento de la cabeza**: La dirección de la cabeza se actualiza según el valor de **snakeDirection:**

   LEFT: Disminuye la posición X.

   RIGHT: Aumenta la posición X.

   UP: Disminuye la posición Y.

   DOWN: Aumenta la posición Y.

   → Hay que recordar que el tablero se maneja como matriz, he aquí la explicación de lo que puede ser un poco confusa la idea en subir y bajar.

3. **Colisiones**:

Con los bordes: Si la cabeza supera los límites del tablero (W o H), el juego termina.

Con la cola: Si la cabeza coincide con cualquier parte de la cola, el juego también termina.

4. **Interacción con la comida**: Si la cabeza toca la comida **(headPos == foodPos):**

Se incrementa la puntuación **(score += 10).**

La longitud de la cola **(tailLenght)** aumenta.

La comida se reposiciona en una nueva ubicación aleatoria dentro del tablero.

Relación entre estos métodos:

- **render:** Muestra los cambios visuales provocados por **input** y **gameLogic.**
- **input:** Permite al jugador interactuar, alterando la dirección de la serpiente.
- **gameLogic:** Define qué pasa después de cada acción, controlando las reglas del juego.