Snake

Nombre: Bernardo A. Márquez González
Materia: Estructura de datos, C
Profesor: Zaira Zuviría

Github Estructura-de-datos/2do-parcial/snake

Fecha de entrega: 2023-06-30

El código proporcionado implementa el juego de la serpiente (Snake) utilizando la biblioteca ncurses en C. A continuación se presenta una explicación paso a paso del funcionamiento del código:

1. Incluye las bibliotecas necesarias, como stdlib.h, stdio.h, etc.
2. Define constantes para el ancho (WIDTH) y alto (HEIGHT) del tablero, así como para los tipos de objetos en el juego (CABEZA, CUERPO, MANZANA) y las direcciones posibles (DERECHA, ARRIBA, IZQUIERDA, ABAJO).
3. Define una estructura llamada Nodo para representar las coordenadas (x, y) y el tipo de un objeto en el juego.
4. Define una estructura llamada Culebra que representa un nodo de la serpiente. Tiene un puntero al nodo actual, así como punteros al nodo anterior (ant) y siguiente (sig) en la serpiente.
5. Declaraciones de funciones: setup(), draw(), grow(), endGame(), gameLogic(), userInput(). Estas funciones se implementarán más adelante en el código.
6. Variables globales: culebra (representa la serpiente), fruta (representa la posición de la fruta), run (bandera para controlar el bucle principal del juego), puntos (puntuación del jugador).
7. La función main() es el punto de entrada del programa. Crea dos hilos (gameLogicThread y userInputThread) utilizando la función pthread_create(). Luego, espera a que los hilos terminen utilizando la función pthread_join().
8. La función gameLogic() es ejecutada en un hilo separado y controla la lógica del juego. Comienza llamando a la función setup() para inicializar el juego. Luego, utiliza un bucle while para ejecutar el juego hasta que la bandera run sea falsa. En cada iteración, llama a la función draw() para dibujar el estado actual del juego en la pantalla ncurses. A continuación, verifica si la cabeza de la serpiente se encuentra en la posición de la fruta. Si es así, incrementa la puntuación y llama a la función grow() para hacer crecer la serpiente. Luego, actualiza la posición de cada nodo de la serpiente moviéndolos hacia adelante en la dirección actual. Finalmente,

utiliza la función system("sleep 0.1") para hacer una pausa de 0.1 segundos antes de la siguiente iteración.

9. La función userInput() se ejecuta en un hilo separado y maneja la entrada de usuario. Utiliza las funciones del sistema "stty -echo" y "stty cbreak" para desactivar el eco de entrada y habilitar el modo RAW. Luego, entra en un bucle while y lee el carácter de entrada del usuario utilizando la función getch(). Dependiendo del carácter de entrada, cambia la dirección de la serpiente o establece la bandera run en falso para terminar el juego. Al final del bucle, utiliza las funciones del sistema "stty echo" y "stty -cbreak" para restaurar el eco de entrada y volver al modo COOKED.
10. La función setup() inicializa el entorno de ncurses y otras variables necesarias. Configura la semilla aleatoria, establece las coordenadas iniciales de la cabeza de la serpiente, la dirección de movimiento, y genera la posición inicial de la fruta de forma aleatoria.
11. La función draw() borra la pantalla y dibuja el estado actual del juego utilizando las funciones de ncurses. Dibuja los bordes del tablero, muestra la dirección y puntuación actual, y dibuja la fruta y la serpiente en sus respectivas posiciones.
12. La función grow() hace crecer la serpiente. Crea un nuevo nodo en memoria, lo enlaza al final de la serpiente y genera una nueva posición para la fruta.
13. La función endGame() finaliza el juego. Establece la bandera run en falso, dibuja el estado final del juego en la pantalla y muestra un mensaje de "HAS PERDIDO!!!".

El reto mas grande para esta segunda entrega fue **recordar** que existen hilos los cuales se pueden usar para llevar procesos separados al mismo tiempo, esto y basando la lógica del juego en maquinas de estado, permitió tener un input de usuario y un movimiento de culebra mucho mas fluidos.