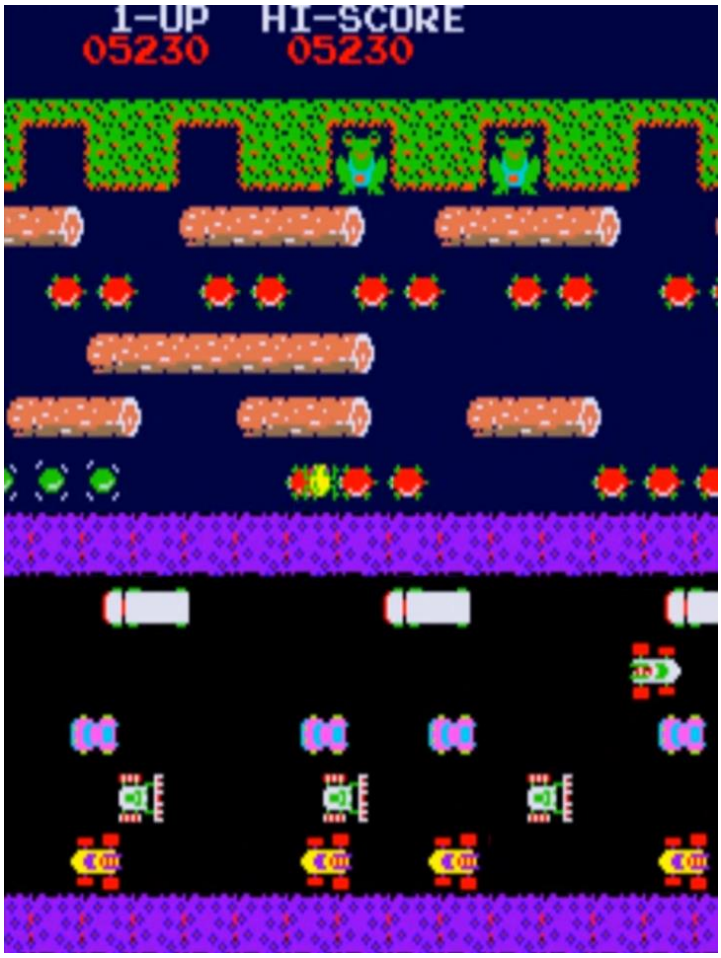
Tarea 1: Frogger

Introducción

En esta tarea usted debe implementar una versión simplificada de Frogger, el clásico juego de Arcade. Para esto debe utilizar C++20 y CMake para el proyecto general, BoxRenderer para los gráficos y Miniaudio para el audio. Su proyecto debe ser capaz de compilarse con Visual Studio Community 2022 y ejecutarse en el sistema operativo Windows.



https://www.youtube.com/watch?v=l9fO-YuWPSk

El video en youtube muestra las características de gameplay, Frogger se mueve desde la parte inferior hasta alguno de los casilleros disponibles de la parte superior. Frogger puede morir atropellado o ahogado en su trayectoria.

Note que Frogger se mueve discretamente sobre una grilla regular, con la única excepción de cuando se traslada por los troncos. A pesar de eso, al salir de los troncos, su movimiento sigue siendo a través de la misma grilla, *¡hint para la implementación!*.

Simplificaciones

- Solo es necesario que considere dos tipos de vehículo (de tamaños 1 y 3 celdas), y dos tipos de troncos (de tamaños 2 y 4 celdas).
- Omita el sistema de puntaje.

- Una sola iteración del juego es suficiente, es decir, solo un Frogger debe completar este desafío (y no 5). La condición de victoria puede pasar en la parte superior por completo, no es necesario programar/habilitar los 5 slots.
- Puede implementar solo un sector: calle o rio, no es necesario implementar ambos. Si lo hace, no otorga puntaje adicional. Note que el problema es el mismo, uno es el "negativo" del otro.
- Con BoxRenderer puede dibujar exclusivamente cajas de colores, utilice estas, o composiciones de las mismas, para las representaciones visuales.
- Modelos gráficos y audios deben ser definidos por usted como parte del mismo código. Es decir, sin cargar imágenes o sonidos externos, sino que definiéndolos vía código.

Bibliotecas de terceros

- Miniaudio: Para reproducción y procesamiento de audio 2D.
  - https://miniaud.io/ , https://github.com/mackron/miniaudio
  - Ejemplo de uso: examples/simple_playback_sine.c
- BoxRenderer: Para dibujar y controlar cajas de colores.
  - https://github.com/dantros/box_renderer
  - Ejemplo de uso: samples/hello_box_renderer.cpp

Entregables

- Archivo adjunto: Pdf con la autoevaluación
  - Comente en una tabla como la mostrada en la sección "puntajes", que ítems le faltaron y el puntaje que usted cree que tendría. Cada categoría tendrá 0, 0.5 o 1.0 según completitud.
  - Si no adjunta autoevaluación, no tendrá derecho a reclamo.
- Enlaces en la descripción de la entrega:
  - Link a repositorio GitHub **privado**.
    - No incluya archivos de Visual Studio en su repositorio, el equipo docente los generará desde CMake.
    - Miniaudio y BoxRenderer deben ser incluidos como submodulos git.
    - Deberá invitar como colaboradores a todo el equipo docente. Se publicarán usuarios GitHub del equipo docente en u-cursos/Novedades.
  - Video o link a video de instrucciones y demostración.
    - Recuerde que u-cursos posee un límite al tamaño máximo de archivos

Importante

- Aún no se ve modelo de "GameObjects" ni arquitectura general de un motor de videojuegos, este es un ejercicio principalmente de programación en C++ para desarrollar estas habilidades. No se complique "sobre modelando" el software. Mantener el loop de iteración y variables globales es suficiente por ahora.
- El código de su tarea no debe ser compartido de ninguna forma durante el periodo de desarrollo de esta tarea. Por ejemplo, **no debe almacenar su código en un repositorio GitHub público**. Siempre puede debatir ideas y algoritmos, pero no puede compartir código.
- Tome decisiones razonables para obtener un prototipo funcional, el enunciado no posee letra chica. En cada categoría de puntaje se otorgará 0, 0.5 o 1.0, según criterio de completitud.

Puntuación en siguiente página …

Puntajes

	Software	Puntaje
P1	Entradas de usuario a través de flechas del teclado	1.0
P2	Movimiento automático de autos y/o troncos, algunos hacia la derecha y otros hacia la izquierda ordenados en pistas.	1.0
P3	Condiciones de inicio y término de juego (ganar y perder, detectar colisiones)	1.0
P4	Organización del proyecto en varios archivos C++, conectando el ejecutable a bibliotecas estáticas o dinámicas vía CMake.	1.0
P5	Gráficos y sonidos.	1.0

	Video	Puntaje
P6	Se debe apreciar la generación de archivos de Visual Studio vía CMake, la compilación y la ejecución del videojuego, y la presentación de las funcionalidades implementadas. El equipo docente seguirá este video para revisar si su tarea funciona. Edite el video para que en ningún caso exceda los 5 minutos, pero mientras más breve, mejor.	1.0