Proyecto 1: Battle City Tec

Profesor: Ernesto Rivera Alvarado

I. Introducción

Este proyecto tiene la finalidad de que el estudiante repase los conceptos de programación básicos dado que son indispensables para la comprensión correcta del presente curso, específicamente en el lenguaje de programación C. Como se verá, el lenguaje C tiene varias características que hacen que su dominio por parte de los ingenieros en electrónica sea indispensables. Dichas características son:

- Provee acceso de bajo nivel a la memoria y a los componentes de hardware.
- La traducción entre C y el lenguaje ensamblador es de manera directa.
- C es ampliamente utilizado en la industria y en áreas afines a la ingeniería electrónica tal como lo son el diseño de sistemas en tiempo real, programación de sistemas empotrados, virtualización de arquitecturas computacionales, desarrollo de sistemas operativos, desarrollo de equipo de red y computación de alto desempeño.
- Es utilizado a nivel didáctico para explicar el flujo de traducción entre lenguajes de alto nivel y lenguaje máquina de una arquitectura dada.

Para ello, el estudiante hará uso de la rutina de gráficos SDL y se programará en C sobre Linux el juego Battle City de la consola Nintendo Entertaiment System. La dinámica específica del juego se puede apreciar en el siguiente link: https://www.youtube.com/watch?v=MPsA5PtfdL0.

Los detalles del funcionamiento del juego se pueden apreciar en el siguiente link: https://strategywiki.org/wiki/Battle_City/Gameplay.

II. Configuración del sistema y código base

- Para desarrollar el proyecto, es necesario que el estudiante instale en su computadora un sistema operativo Linux de su preferencia (no se admite el uso de máquina virtual).
- Posterior a esto, es necesaria la instalación de la biblioteca SDL en Linux.
- El estudiante debe basarse en el código proporcionado por el profesor a través del Tec Digital en la carpeta *Proyectos*.
- Todo el código debe realizarse en C sobre Linux.

III. Descripción

Tal como se comentó en la introducción, se le solicita al estudiante replicar el videojuego Battle City, específicamente los 5 primeros niveles. Debe de contar con las siguientes características, las cuales se ponderarán en partes iguales para un total de 60 puntos del proyecto. Cualquier duda que el estudiante tenga sobre la implementación de una dinámica del juego debe ser resuelta mediante el análisis del video provisto como referencia en este instructivo, o jugando el juego. El juego debe desarrollarse para un jugador únicamente. Las características que se deben incluir son:

- El juego cuenta con pantalla de inicio que describe el curso, universidad y autores, en donde al presionar una tecla seleccionada a criterio del estudiante, se inicia el juego.
- El jugador se mueve en los ejes “x” y “y”.
- Todas las colisiones del movimiento del jugador están correctamente implementadas.
- Implementa correctamente en mecánica y funcionalidad, el enemigo *basic tank*.
- Implementa correctamente en mecánica y funcionalidad, el enemigo *fast tank*.
- Implementa correctamente en mecánica y funcionalidad, el enemigo *power tank*.
- Implementa correctamente en mecánica y funcionalidad, el enemigo *armor tank*.
- Contiene los cuatro enemigos diferentes que se aprecian en el video.
- Si el número de vidas es cero, el juego termina y vuelve a la pantalla de inicio.
- Implementa correctamente en mecánica y funcionalidad, el *power-up grenade*.
- Implementa correctamente en mecánica y funcionalidad, el *power-up helmet*.
- Implementa correctamente en mecánica y funcionalidad, el *power-up shovel*.
- Implementa correctamente en mecánica y funcionalidad, el *power-up star*.
- Implementa correctamente en mecánica y funcionalidad, el *power-up tank*.
- Implementa correctamente en mecánica y funcionalidad, el *power-up timer*.
- Implementa correctamente los 6 “*power ups*”.
- La mecánica de aparición de los enemigos es igual a la del juego.
- La mecánica del escudo del jugador es idéntica a la del juego.
- Los enemigos presentan sus variantes en las que parpadean con un color rojo.
- Los enemigos se comportan de la misma manera que en el juego.
- Implementa correctamente el nivel 1 tal como en el video.
- Implementa correctamente el nivel 2 tal como en el video.
- Implementa correctamente el nivel 3 tal como en el video.
- Implementa correctamente el nivel 4 tal como en el video.
- Implementa correctamente el nivel 5 tal como en el video.
- El juego consta de 5 niveles diferentes.
- Los niveles imitan el estilo, dificultad y comportamiento de los primeros 5 niveles del juego (ladrillos, bloques de

metal, bloques de árboles, etc).
- El juego cuenta con contadores de enemigos restantes y de vidas que se comportan tal como en el juego.
- Se replica la mecánica en la cual el jugador puede perder el juego.
- Se replica la mecánica para eliminar enemigos.
- Se replica la mecánica de movimientos, disparos y uso del personaje que el jugador humano controla.
- Se replica la dinámica en que los disparos afectan el nivel.
- El juego funciona libre de *bugs*, *glitches* o cualquier otra evidencia de mal funcionamiento.
- Incorpora cualquier otra mecánica del juego que se evidencie en el video y que no se contempla en la lista anterior.

Considere que las características mencionadas anteriormente deben funcionar de manera correcta y evidenciarse en la ejecución del juego para que se les cuente en el apartado de completitud del proyecto. En cuanto a la programación en C, el estudiante debe cumplir con los siguientes requisitos.

- Definición y utilización de estructuras para representar los diferentes elementos del juego.
- Creación de elementos dinámicamente a través de `malloc` así como su destrucción correcta. Los enemigos del juego deben crearse y eliminarse de esta manera.
- Los enemigos se deben gestionar a través de listas enlazadas, en donde en su creación se agregan a la lista, y en su destrucción se eliminan de la lista.
- Su código debe seguir buenas prácticas de desarrollo. Se le recomienda revisar https://www.doc.ic.ac.uk/lab/cplus/cstyle.html y https://users.ece.cmu.edu/~eno/coding/CCodingStandard.html

IV. Revisión y completitud del proyecto

Es de suma importancia que el estudiante incluya dentro de su proyecto todas las características mencionadas en la descripción, ya que este será un aspecto que se evaluará fuertemente en la revisión del mismo. Por cada característica que el estudiante no incluya, se rebajará de la nota final 3^k puntos hasta un máximo de 40, donde k corresponde a la cantidad de características que usted no incluyó, esto aplica para $k > 1$. Adicional a esto, las funcionalidades no incluidas, o que no funcionan correctamente en el proyecto se rebajarán adicional al rubro de completitud mencionado anteriormente.

La evaluación de este proyecto está sujeta a la presentación oral del mismo, en la que el estudiante debe mostrar un dominio completo del trabajo realizado a nivel de código. En caso de que el estudiante no muestre un dominio completo del trabajo, se considerará el trabajo como "no revisable" y se asignará la nota de cero. La defensa del proyecto será grabada.

En caso de trabajar en pareja, ambos integrantes deberán tener un dominio completo del trabajo que están presentando. El profesor podrá hacerle preguntas a cualquiera de los integrantes y este deberá contestar acordemente. En caso de que se evidencia que uno de los integrantes desconoce el funcionamiento de alguna parte del trabajo presentado, se le asignará la nota de cero a ambos.

V. Requisitos indispensables

La ausencia de uno solo de los siguientes requisitos vuelve al proyecto "no revisable" y recibe un 0 de calificación inmediata:

- Se sigue a cabalidad lo solicitado en la Sección II del presente documento.
- No debe presentar "*Segmentation Fault*" o "Violación de segmento" por ninguna razón.
- Los estudiantes deben demostrar dominio completo de su proyecto en la presentación oral.
- Los estudiantes deben utilizar el sistema de control de versiones Git tal como se describe en este documento.

VI. Desarrollo del proyecto

El proyecto está pensado para desarrollarse individualmente, sin embargo, los estudiantes que deseen reforzar la habilidad de trabajo en equipo pueden entregar el proyecto en parejas. El profesor les hace la aclaración de que el trabajo con un compañero conlleva dificultades de coordinación, división de trabajo y sobre todo de "pegar o juntar ambas partes". En experiencias propias del profesor, se les comenta que en ocasiones el trabajo de juntar, acoplar y corregir partes desarrolladas por diferentes personas conlleva más tiempo y trabajo que la realización individual.

VII. Git

Los estudiantes deberán utilizar el sistema de control de versiones Git, específicamente a través de GitLab. Para ello configurarán desde sus máquinas Gitlab, para poder hacer cualquier acción (ejemplo *push*, *pull*, *commit*) desde **consola de texto**, sin la necesidad de ingresar la contraseña constantemente. Además agregarán al repositorio del proyecto al usuario Ernesto.cursos, donde el profesor podrá observar los avances y desarrollos del proyecto por semana, día, hora, minuto y segundo.

VIII. Consideraciones sobre plagio

El trabajo presentado por los estudiantes (ya sea a nivel individual y de parejas) debe ser de su propia autoría. No se permite utilizar código realizado por un tercero (sin importar la licencia de dicho código) o entre compañeros del mismo curso, y en caso de que esto ocurra, será considerado plagio por lo que se seguirá el proceso correspondiente. Códigos encontrados en "github" de los cuales solo se tomaron "partes" será considerado plagio, el estudiante es el responsable de desarrollar absolutamente todo su código, a excepción del código que se menciona como base en el presente documento. De igual manera, cualquier código que el estudiante no tenga la capacidad de explicar, será considerado como plagio.

IX. Fecha de entrega

Las demostraciones se harán en la **semana 5** asignadas por citas distribuidas aleatoriamente en las dos lecciones de la semana.

La carpeta comprimida `.zip` de su proyecto debe contener los archivos fuentes `.c` y complementarios, además

del archivo `Makefile` que genera el ejecutable del código fuente. El nombre de la carpeta comprimida debe de ser los apellidos de integrantes del grupo de trabajo (por ejemplo `rivera-alvarado.zip`) y este será subido al correo del profesor y al Tec Digital el día de la revisión a antes de las 6 am. Los estudiantes deberán defender el trabajo realizado y mostrar un dominio completo del trabajo desarrollado. El no tener conocimiento de cómo funcionan partes del proyecto invalida la entrega y se asignará la nota de cero.