Proyecto I – Battlespace

Instituto Tecnológico de Costa Rica
Área Académica Ingeniería en Computadores
CE2103 - Algoritmos y Estructuras de Datos II
Primer Semestre 2023
Valor: 25%

Objetivo General

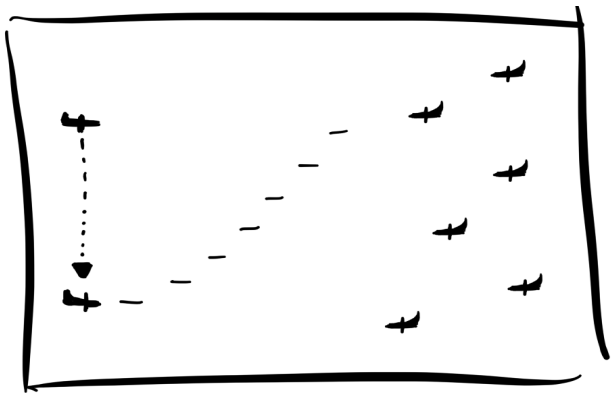
- ➔ Desarrollar una aplicación utilizando el lenguaje de programación C++.

Objetivos Específicos

- ➔ Aplicar conceptos de manejo de memoria.
- ➔ Investigar y desarrollar una aplicación en el lenguaje de programación C++
- ➔ Implementar una solución utilizando programación orientada a objetos en C++.
- ➔ Aplicar patrones de diseño en la solución de un problema.

Descripción del Problema

Deben implementar un juego de una nave espacial que debe combatir contra naves enemigas, con una interfaz similar a la siguiente imagen:



Al iniciar el juego, se debe seleccionar el nivel del juego (inicial, intermedio, experto). Este nivel de juego determina la cantidad de balas con las que se cuentan, la cantidad de naves en cada oleada y la velocidad con la que avanzan las naves enemigas. La configuración de cada uno de estos aspectos, de acuerdo con cada nivel, debe ser definido por cada grupo de trabajo, de manera que esta configuración pueda demostrar cada una de las funcionalidades de esta especificación.

La nave solo permite moverse de arriba a abajo y las balas se disparan de forma automática, con una velocidad que puede ser ajustada desde el mismo juego.

El juego está conformado por fases, donde cada fase está conformada por una cantidad de cinco oleadas, donde a su vez, cada oleada está conformada por una cantidad de naves enemigas. Cada una de estas naves tienen un nivel de resistencia al impacto de las balas. Algunas naves deben tener la posibilidad de moverse de forma autónoma en el eje y. Adicionalmente, cada nave tiene un color asignado. La información general de cada una de las cinco oleadas de las fases debe ser almacenada en un arreglo. La información de las naves que conforman cada oleada debe ser almacenada en una lista enlazada. Cada nave enemiga que sea destruida debe ser eliminada de la lista.

Bullet Collector

Cuando una bala no impacta una nave enemiga debe ser almacenada en el Bullet Collector, de manera que cuando se quede sin balas pueda reutilizar estas balas. Las balas que sean reutilizadas deben ver disminuido el impacto que causen a las naves enemigas. Cada bala debe implementarse como un objeto en memoria, de manera que al ser reutilizada, corresponde a la información del mismo bloque en memoria, solo que con una disminución en el impacto que causa a las naves enemigas.

Paged Powers

Cada grupo de trabajo debe definir una serie de poderes, los cuales van a estar asociados a una estrategia. Para esto se requiere que cada grupo establezca al menos cuatro estrategias y que cada una de estas tenga al menos dos poderes. Los poderes deben corresponder a aspectos del juego.

La información de cada estrategia, con sus respectivos poderes, deben estar almacenados en archivos XML. Un archivo por cada estrategia.

El juego solo debe permitir tener en memoria la información de poderes de dos estrategias. Cuando un jugador selecciona un poder, si la estrategia de este poder está en memoria, puede usar el poder de inmediato. Caso contrario, debe esperar 5 segundos, para poder cargar la información del poder (archivo XML). El usuario no debe tener forma de saber si el poder que desea utilizar se encuentra en memoria.

Controller

Se debe implementar un control físico, implementado con un Arduino, que permita las siguientes funcionalidades:

1. Mover la nave sobre el eje y.
2. Mediante un potenciómetro regular la cantidad de balas que se puedan disparar por segundo
3. Mediante un sonido, alertar cuando una nave enemiga sobrepase la posición, en el eje x, de la nave manipulada por el jugador.
4. Siete segmentos que indique la cantidad de oleadas restantes de la fase.

Requerimientos no funcionales

1. Cabe resaltar que es de manera obligatoria realizar la maqueta del control para interactuar con la aplicación. No se permite presentar el circuito únicamente en la protoboard.
2. Bibliotecas para comunicación serial con Arduino: jSerialComm o RXTX.

Documentación requerida

1. Internamente, el código se debe documentar siguiendo los estándares de documentación para cada lenguaje utilizado.
2. La documentación externa deberá desarrollarse en Latex utilizando la plantilla llamada IEEE Conference Template, que se puede descargar desde aquí. Para el desarrollo en Latex se recomienda el uso de Overleaf. La documentación debe contener las siguientes secciones:
    a. Breve descripción del problema.
    b. Diagrama de clases.
    c. Descripción de las estructuras de datos desarrolladas.
    d. Descripción del funcionamiento del algoritmo de paginación
    e. Protocolo de comunicación entre cliente y servidor
    f. Problemas encontrados: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.

3. **Planificación y administración del proyecto:** se utilizará Azure DevOps para la administración del proyecto. Debe incluir:
    a. Lista de features e historias de usuario identificados de la especificación.
    b. Plan de iteraciones que agrupen cada bloque de historias de usuario de forma que se vea un desarrollo incremental
    c. Descomposición de cada user story en tareas.
    d. Asignación de tareas entre los integrantes del grupo

Aspectos operativos y evaluación:

1. **Fecha de entrega: De acuerdo con el cronograma del curso y lo establecido en el TEC Digital**
2. El proyecto tiene un valor de 25% de la nota del curso.
3. El trabajo es **en parejas.**
4. Deben entregar un documento con el link del repositorio de GitHub, Azure DevOps y el PDF de la documentación. En ambas herramientas deben dar acceso al correo del profesor.
5. Es obligatorio utilizar un Git y GitHUB para el control de versiones del código fuente y evidenciar el uso de Commits frecuentes.
6. Es obligatorio integrar toda la solución.
7. El código tendrá un valor total de 60%, la implementación del control físico (arduino) un valor de 20% y la documentación externa 20%.
8. De las notas mencionadas en el punto anterior se calculará la Nota Final del Proyecto.
9. Se evaluará que la documentación sea coherente, acorde a la dificultad/tamaño del proyecto y el trabajo realizado. Se recomienda que realicen la documentación conforme se implementa el código.
10. La nota de la documentación externa es proporcional a la completitud del proyecto.
11. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
12. Los estudiantes pueden seguir trabajando en el código hasta 15 minutos antes de la primera cita de revisión oficial.
13. Aún cuando el código y la documentación externa tienen sus notas por separado, se aplican las siguientes restricciones
    a. Si no se entrega documentación externa, automáticamente se obtiene una nota de cero en la nota final del proyecto.
    b. Si no se utiliza un manejador de código se obtiene una nota de cero en la nota final del proyecto.
    c. Si la documentación externa no se entrega en la fecha indicada se obtiene una nota de cero en la nota final del proyecto.
    d. Sí el código no compila se obtendrá una nota de cero en la nota final del proyecto, por lo cual se recomienda realizar la defensa con un código funcional.
    e. El código debe ser desarrollado en C++ (Linux), en caso contrario se obtendrá una nota de cero en la nota final del proyecto.
14. La revisión de la documentación podría ser revisada antes, durante o después de la cita de revisión del proyecto.
15. Cada grupo tendrá como máximo 15 minutos para exponer su trabajo al profesor y realizar la defensa de éste. Es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa.
16. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
17. Cada grupo es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberán avisar al menos 2 días antes de la revisión al profesor para coordinar el préstamo de estos.
18. Durante la revisión únicamente podrán participar el estudiante, asistentes, otros profesores y el coordinador del área.