Hola Juan,   
como te habiamos anticipado, te envio un pequeño ejercicio como para tener una mejor idea sobre tu conocimiento como Game Developer.   
La idea es que lo estes entregando como maximo el lunes 11-2, y a la brevedad vas a recibir una devolucion y como continamos con el proceso de seleccion.  
  
Te pido que ante cualquier duda, por favor cuentes conmigo para despejarla.  
  
Saludos!!  
-------------------------------------------------------------------------------------

Ejercicio Unity - Space Shooter

El objetivo es desarrollar un juego de naves, similar al Galaxian salvo algunas particularidades. Es recomendable jugar o ver un video de ese juego para entender mejor la consigna.

<https://youtu.be/NoZkFd_QH6I>

El Juego:

El juego inicia en la Escena de Menú Principal. Aquí debe haber como mínimo una opción para iniciar el juego y un indicador que muestre el Highscore, si se desea pueden agregarse otras, por ejemplo Configuración.  Al iniciar el juego, se debe pasar a la Escena de Juego.

El juego consistirá en una nave controlada por el jugador, posicionada en la parte inferior de la pantalla, y filas de enemigos que se moverán automáticamente en grupo rebotando de lado a lado, algunos de estos enemigos pueden desprenderse del grupo y “saltar” hacia abajo, a su vez, algunos enemigos tienen la capacidad de disparar al jugador. Los enemigos no colisionan entre sí, ni pueden matarse entre sí con sus disparos.

La disposición de los enemigos es siempre la misma, pero debe ser aleatorio qué tipo de enemigo aparece en cada una de las posiciones. La probabilidad de aparición de cada tipo de enemigo depende de la dificultad del mismo y del nivel en el que nos encontremos, como se aclara más abajo.

Los enemigos varían además en la vida que tienen, de este valor dependerá cuantas disparos hay que pegarle para que muera. Cada disparo hace 1 de daño, la vida es un valor flotante pero se redondea de forma que si por ejemplo el enemigo tiene 1.2 de vida, muera con 1 disparo, pero si tiene 1.8, muera con 2.

El jugador puede mover la nave de lado a lado y disparar, al impactar contra un enemigo o un disparo enemigo, pierde.

La interfaz del juego debe mostrar como mínimo el nivel y puntaje actuales, el Highscore, y un botón para finalizar la partida (este botón podría estar también en un menú de pausa).

El objetivo será destruir a todos los enemigos para avanzar al siguiente nivel.

Los niveles son infinitos y todos de igual disposición pero dificultad incremental. El incremento en la dificultad está dado por 2 factores:

Multiplicador de vida en los enemigos (vida final = vida inicial \* multiplicador vida),

Más probabilidad de aparición de enemigos difíciles: Cada enemigo deberá tener una probabilidad de spawneo asociada, al ir incrementando el nivel, los más difíciles deberán tener más probabilidad de aparición, y los más fáciles menos. (el nivel 2 debería spawnear en promedio más enemigos difíciles que el nivel 1, etc).

El juego termina cuando el jugador pierde o decide finalizar la partida desde el menú. En ambos casos se muestra un resumen de sus puntos, si se logró Highscore o no, y una opción para volver al menú principal o reiniciar el juego. El highscore debe mantenerse entre sesiones de juego.

Considerar para la estructuración del proyecto y del código, un escenario en el que a futuro se planee agregar nuevos tipos de enemigos con distintas habilidades a las actuales.

Enemigos ordenados de más fácil a más difícil:

No dispara. No se desprende del grupo. 1.5 vida, 1 punto.

Dispara siempre cada 5 segundos. No se desprende del grupo. 1 vida, 2 puntos

Dispara cuando el jugador pasa por debajo de el, y si se queda abajo sigue disparando cada 2 segundos. No se desprende del grupo. 1 vida, 2 puntos.

Dispara siempre cada 3 segundos. Se desprende del grupo en línea recta hacia abajo. 1.5 vida, 3 puntos

Dispara cuando el jugador pasa por debajo de el, y si se queda abajo sigue disparando cada 1 segundo. Se desprende del grupo haciendo un arco hacia el jugador. 1.6 vida, 3 puntos.

Escenas:

El juego debe contar con 2 escenas. Una escena inicial con el menú principal del juego. Otra con los elementos del juego y sus interfaces. Se debe pasar de una escena a otra al entrar o salir del juego.

Puntos a Evaluar:

Estructura del proyecto

Manejo de cambios de escena

Calidad de código. (Estructura de clases, Acoplamiento, Cohesión, Escalabilidad, etc)

Uso de patrones de diseño.

Uso de eventos.

Criterio en las decisiones de los aspectos del juego que no están especificados.

Forma de Entrega:

Archivo ZIP con el nombre del candidato que contenga el proyecto (carpetas Assets y ProjectSettings) y un archivo Readme especificando versión de Unity utilizada, desde qué escena se debe correr el juego y cómo jugar si es que no posee instrucciones ingame.