ITCR

Ingeniería en Computadores

Algoritmos y Estructuras de Datos 1

Documentación Proyecto 3: Documento de Acreditación

Estudiantes:
Esteban Campos Abarca - 2022207705
Jafet Díaz Morales - 2023053249

II Semestre 2024

2. Tabla de contenidos.

3. Introducción.

El presente documento tiene como objetivo documentar que las personas estudiantes son capaces de diseñar soluciones creativas para problemas de ingeniería complejos, diseñando sistemas, componentes o procesos para satisfacer las necesidades identificadas. Esto trasciende a ámbitos tales como la salud y la seguridad pública, el costo total de la vida, el carbono neto cero, así como las consideraciones de recursos, culturales, sociales y ambientales según sea necesario.

En este documento de forma específica, se trata el diseño creativo y ordenado de un videojuego: Airwar. Este videojuego tiene como objetivo principal el destruir (con un “cañón” que se mueve) la mayor cantidad de aviones posibles a medida que estos se generan en un mapa. El videojuego será desarrollado en C#.

El propósito del videojuego, además de comprobar un diseño creativo, comprueba un diseño ordenado al emplearse métodos de documentación técnica empleados en el mundo laboral real (tales como User Stories, diagramas UML y de arquitectura, etc.).

Finalmente, el proyecto comprueba conocimientos básicos técnicos de la programación, tales como el paradigma orientado a objetos (POO), estructuras de datos como grafos y algoritmos de búsqueda y de ordenamiento.

4. Diseño

a. Listado de requerimientos descritos con el formato de historias de usuario.

Historia de Usuario				Criterio de Aceptación	Estado
	YO	QUIERO	PARA		
1	Como jugador	Un juego en el que pueda destruir aviones	Destruir la mayor cantidad posible	El juego dura un periodo de tiempo definido por el programador	DONE
2	Como jugador	Que se generen aleatoriamente aeropuertos, portaaviones y rutas.	Que hayan aviones a los cuales dispararles.	Se modela con un grafo utilizando listas de adyacencia.	DONE
3	Como jugador	Una batería antiaérea que se mueve en velocidad constante entre izquierda y derecha de la pantalla.	Para que dispare a los aviones	El jugador presiona click para disparar balas con trayectoria recta	DONE
4	Como jugador	Que entre aeropuertos y portaaviones, se generen rutas aleatorias	Que los aviones se muevan por ellas	Las rutas tendrán distintos pesos. El peso está dado por la distancia entre los puntos que conecta y el precio de la ruta y del destino (un portaaviones es más caro que un aeropuerto, ir por el océano es más caro que un continental)	DONE
5	Como jugador	Que al despegar el avión decida a qué destino quiere ir aleatoriamente	Que se mueva en el mapa y aterrice en algún lugar donde recargue combustible	El destino es aleatorio pero la ruta es dada en base a los pesos. Cuando aterriza espera una cantidad aleatoria de segundos y luego despega	DONE

				otra vez y recarga combustible	
6	Como jugador	Que los aeropuertos racionen el combustible de los aviones	Que los aviones no recarguen totalmente y el avión pueda caerse si se queda sin combustible	La forma de racionar es determinada por el programador	NOT DONE
7	Como jugador	Que los aeropuertos puedan construir nuevos aviones.	Que hayan aviones volando de forma constante	Se generan cada cierto tiempo. La cantidad que pueden generar está definida por un atributo de los aeropuertos que indica la cantidad de hangares. Los aviones deben tener un ID generado con GUIDs.	DONE
8	Como jugador	Que en la teoría/concepto del juego los aviones vuelen solos y tengan 4 módulos de AI: 1. Pilot 2. Copilot 3. Manteinance 4. Space awarness	Para poder conocer los datos del avión y de la tripulación (derribados) y ordenarla	Cada de uno tiene los siguientes atributos: 1. ID: es un identificador que consta de tres letras (de la A a la Z) generadas aleatoriamente a la hora de crear el avión. 2. Rol (piloto, copiloto, etc.) 3. Horas de vuelo El sistema muestra una lista de aviones derribados que se podrá ordenar por ID (GUIDs) usando Merge	NOT DONE

				Sort	
9	Como jugador	Poder ver los datos de cada avión derribado	Poder ver su tripulación y ordenarla	El orden se da por ID, Rol o horas de vuelo usando Selection Sort	NOT DONE
10	Como jugador	Poder ver en la pantalla todos los datos relevantes del juego.	Ver como funcionan los aviones	Por ejemplo, se deben mostrar los caminos calculados por los aviones, los pesos de cada ruta, los atributos de los aviones, entre otros.	DONE

b. Seleccione cinco problemas que tengan más de una posible solución, y para cada uno de ellos haga lo siguiente

i. Proponga y explique dos alternativas de solución
ii. Para cada alternativa, deje claro cuáles son las ventajas y cuáles las desventajas
iii. Seleccione una de las dos alternativas y explique por qué la seleccionó

Problema 1	Generar las balas que serán utilizadas en el juego para destruir los aviones
Alternativa A	Utilizar un Objet Pool para generar las balas desde el inicio del juego, las cuales simplemente se activan, no es necesario crear y destruir objetos constantemente, lo cual ayuda al rendimiento y consume menos memoria.
Alternativa B	Realizar una función que cada vez que se de el input requerido cree un objeto bala nuevo.
Seleccionado	Se selecciona el B. El pool en la forma ideal es mucho mejor, sin embargo por simplicidad y manejo de tiempo se eligió la otra opción.

Problema 2	Generar colisiones para los aviones
Alternativa A	Utilizar el sistema de colisiones de Unity.
Alternativa B	Asignar colisiones de acuerdo a la posición de ambos objetos
Seleccionado	Se selecciona el A, debido a que el sistema de colisiones de Unity, aparte de ser preciso, requiere menos trabajo que hacer el código para cada colisión de acuerdo a la posición. Por un tema de simplicidad y no generar código innecesario, se utiliza esa opción.

Problema 3	Generar un mapa con un diseño parecido al de la tierra y aleatorio
Alternativa A	Generar una matriz de números binarios, generar esos números de forma aleatoria, asignar cada celda de acuerdo al número.

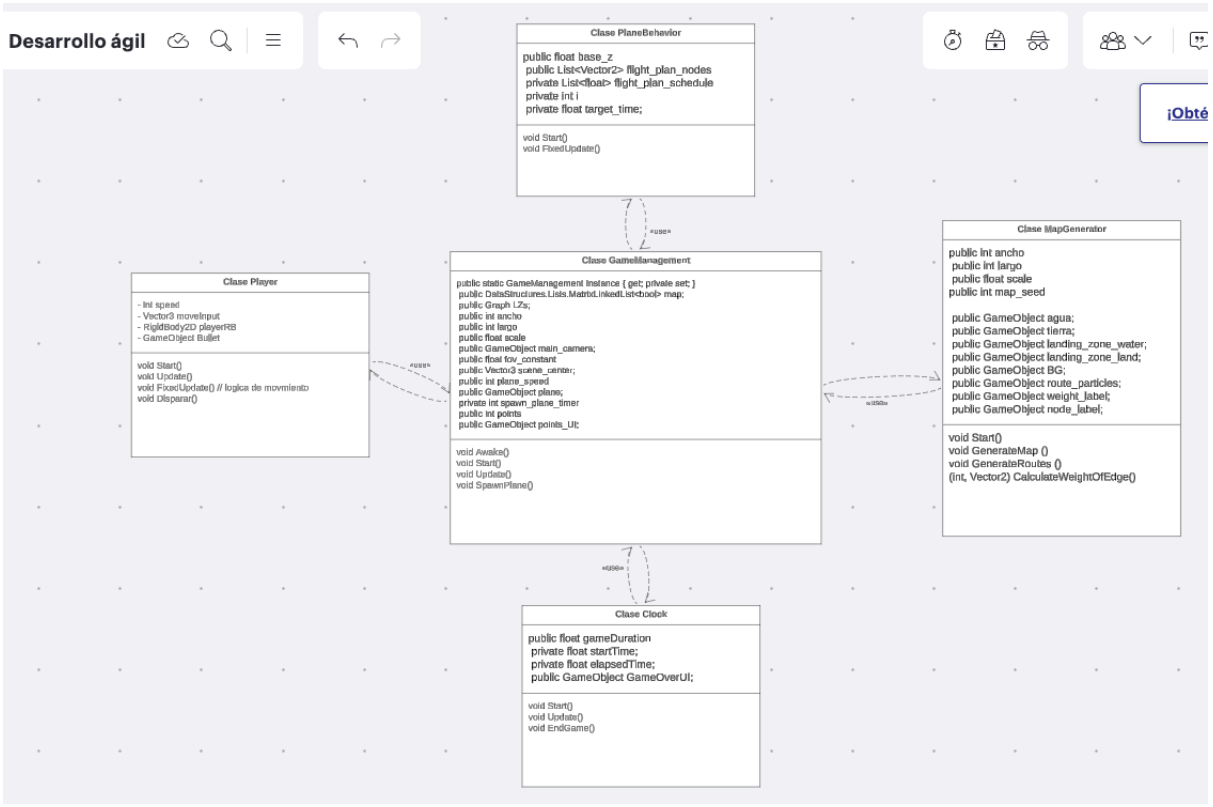
Alternativa B	Utilizar Ruido de Perlin para generar un mapa. Se dan inputs aleatorios en cada instancia.
Seleccionado	La opción seleccionada es la B, debido a que hacer la matriz es difícil por el hecho de que es difícil hacer que parezca un mapa en cada instancia. El ruido de perlin es un método muy utilizado para hacer mapas en videojuegos y es útil porque, además de que se puede hacer aleatorio, es fácil hacer que su forma siempre se parezca a la de un mapa. En conclusión, por facilidad, eficiencia y diseño agradable, Perlin es el más útil.

Problema 4	Lista de adyacencia para aristas en un grafo
Alternativa A	Lista de adyacencia son dos listas, una con nodos y otra con objetos tipo edge (objetos arista, guarda índices y peso)
Alternativa B	Se puede hacer una sola lista que guarde una sola información. La lista guardería objetos tal cual.
Seleccionado	Se utilizaron dos listas (Opción A) para tener un acceso más fácil. Cada nodo tiene un índice por aparte, lo cual hace más fácil su acceso y se puede revisar más fácilmente si hubiera error en cierto aspecto (ya sea peso, índice o el nodo en si) en vez de revisar un objeto con muchos atributos.

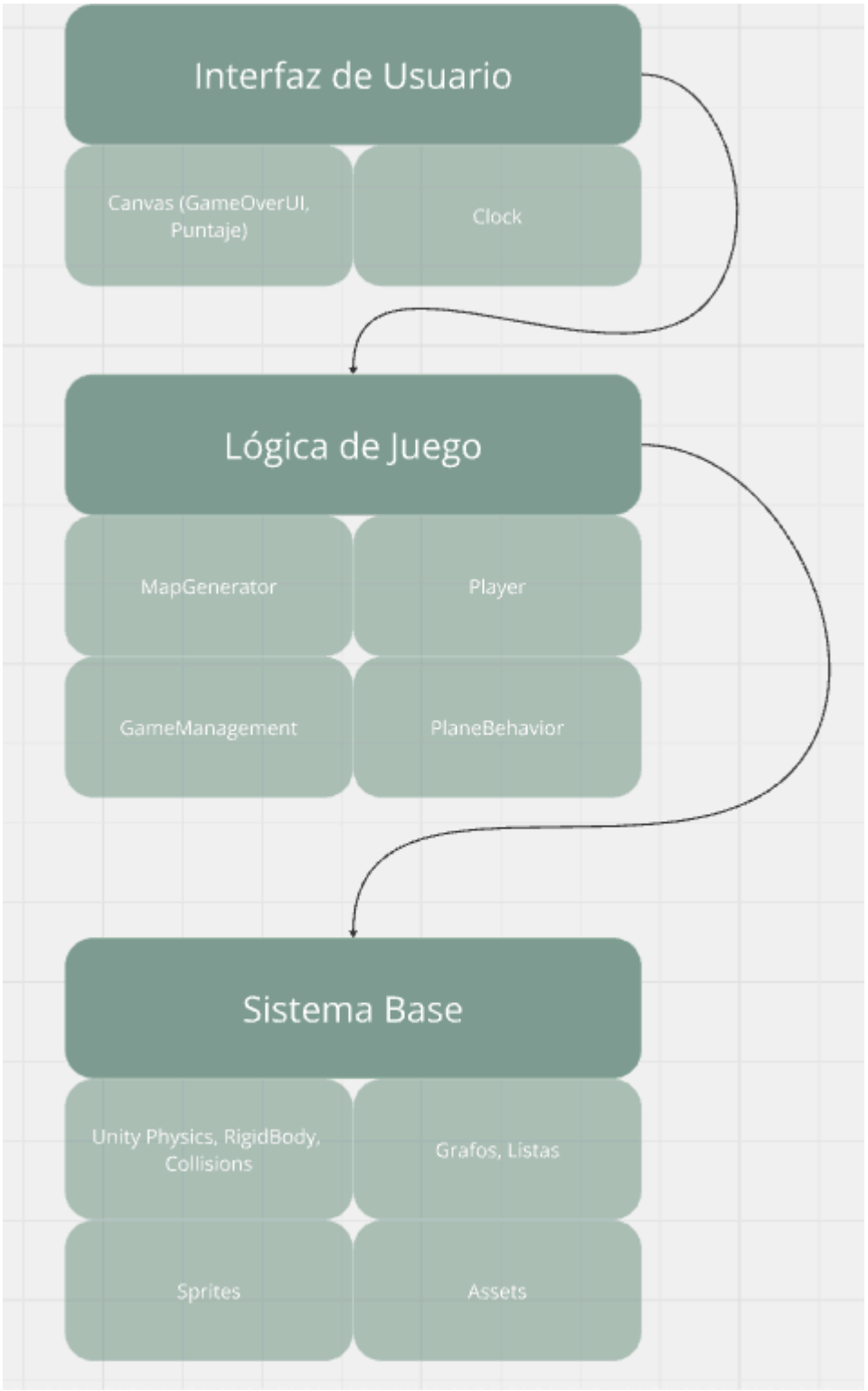
Problema 5	Aviones
Alternativa A	Los aviones utilizan dos listas, una con nodos y otra con flotantes que representan momentos en el tiempo en el que tienen que llegar a cada nodo. El avión solo calcula una vez la distancia y luego registra su tiempo de llegada.
Alternativa B	El avión tiene una lista de nodos a los que quiere ir únicamente. Cuando llega, vuela al otro. La desventaja es que en cada frame hay que calcular la distancia y el tiempo.

Seleccionado	Se selecciona el A puesto que, como el avión aparece cada cierto tiempo porque el juego dura cierto tiempo, todo avión sabe cuando tiene que llegar a cada lugar, esto hace los accesos más sencillos y hay que hacer menos cálculos.

c. Elabore un diagrama de clases usando UML



d. Elabore un diagrama de arquitectura



e. Describa, mediante un checklist, las historias de usuario que fueron implementadas y las que quedaron pendientes.

Se puede visualizar en la tabla del inciso a).