****

SEGUNDO PROYECTO DE TALLER DE PROGRAMACIÓN.

**ELABORADO POR:**

Yaritza Elena López Bustos.

María Fernanda Álvarez Martínez.

**PROFESOR:**

Antonio González Torres.

**I SEMESTRE**

**CARTAGO, 2019**

Contenido

[Introducción 3](#_Toc7385976)

[Descripción del problema. 4](#_Toc7385977)

[Descripción de la solución con UML. 6](#_Toc7385978)

[Análisis de resultados. 7](#_Toc7385979)

[Bitácora de actividades. 8](#_Toc7385980)

[Conclusión. 9](#_Toc7385981)

# Introducción

El presente proyecto consiste en la creación de un código para el juego Dakar Death, cuyo cual es un juego de carros con el único objetivo de sobrevivir el máximo tiempo posible, enfrentándose a otros jugadores y a diversos objetos distribuidos en el mapa, durante el juego se deberá destruir a los jugadores rivales. El nivel de dificultad aumentará pasados los 2 minutos de sobrevivencia.

# Descripción del problema.

Es un juego de carrera de automóviles en el que pueden competir dos o más jugadores, en la que cada jugador usa una computadora diferente. El juego debe utilizar una arquitectura cliente servidor, en la cual el tablero es controlado por el servidor y desplegado por cada cliente en su computadora. Las acciones que realizan los clientes son enviadas al servidor y este actualiza los tableros de todos los jugadores participantes. La velocidad y dirección de cada automóvil debe ser controlada por el jugador con las teclas de su computadora. El juego debe contar con al menos tres niveles de dificultad y su menú tendrá las siguientes opciones:

1. Registrar nombre de usuario: El sistema permite que el jugador ingrese el nombre de su usuario al cual se asociará el puntaje más alto que logre en las competiciones. El puntaje de cada usuario se almacenará en un archivo JSON.

2. Ver puntaje: Cuando esta opción es seleccionada, el sistema muestra el puntaje de los 5 mejores jugadores.

3. Iniciar partida: Al seleccionar esta opción el jugador debe seleccionar el nombre de usuario con el cual jugará la partida.

4. Guardar partida: El jugador puede guardar el estado de una partida al seleccionar esta opción de menú. El sistema debe almacenar las posiciones actuales de todos los carros del juego y los puntajes de los jugadores.

5. Reiniciar partida: Esta opción es utilizada para continuar una partida que había sido iniciada con anterioridad, con toda la información de las posiciones de los carros y los puntajes de los jugadores.

6. Salir: El usuario se sale del sistema y termina la partida, si estuviera jugando en ese momento.

Los jugadores pasan de nivel cuando han superado 2 minutos de juego sin sufrir una colisión. Los niveles superiores del juego deben contar con más obstáculos (definidos por el programador – la mayor dificultad favorece el otorgamiento de más puntos), como mayor número de carros, rocas, muros y árboles. Además, el juego tiene que contar con sonidos de aceleración, disminución de la velocidad, frenado y explosiones al chocar. En este juego el competidor debe sobrepasar o eliminar a los oponentes que son controlados por el mismo juego.

Las reglas básicas del juego son las siguientes:

1. El juego se desarrolla en un desierto lleno de minas y obstáculos y no existe una pista de carreras, por lo que los vehículos pueden correr de forma libre por cualquier parte.

2. El tiempo de cada partida del juego es de 2 minutos.

3. El juego permite que dos o más jugadores compitan a la vez.

4. El sistema controla un gran número de “dummy vehicles” que corren a una velocidad constante. 5. El punto de partida de la competencia y llegada se encuentra indicado por una zona marcada como un tablero de cuadrículas.

6. La velocidad de los vehículos de los jugadores es variable y depende de las maniobras que hagan con las teclas de las computadoras.

7. El puntaje de cada jugador se calcula con base en el número de “dummy vehicles” que elimina durante la competencia y la distancia recorrida.

8. Un jugador puede eliminar el vehículo del competidor rival y los “dummy vehicles” empujándolo para que se destruya contra un obstáculo o caiga a un barranco y también mediante disparos (puede usar cualquier método de destrucción).

9. El jugador que se sale de la pista y no se destruye sufre el rebajo de puntos, de acuerdo con las reglas que defina el programador.

10. Si un jugador sufre una colisión con un objeto letal al salirse de la pista puede quedar destruido.

# Descripción de la solución con UML.

Main

Player

Enemigo

Bala

Niveles

Player

Enemigo

* def movimientoDerecha
* def movimientoIzquierda
* def \_\_movimiento
* def destrucción
* def disparar
* def dibujar
* def hit
* def hit1
* def dibujar
* def comportamientoImages
* def \_\_movimientos
* def \_\_descenso
* def \_\_movimientolateral
* def \_\_ataque
* def \_\_disparo
* def hit
* def convertir
* def decodificar
* def cargar\_mapa

Niveles

* def trayectoria
* def dibujar
* def cargarEnemigo
* def detenerJuego

Bala

# Análisis de resultados.

1. En este proyecto se logró la movilidad de los jugadores con sus respectivos disparos cada uno, la creación de los diversos mapas, además se logró crear los diversos ¨enemigos¨.
2. Por otro lado, dicho proyecto también es deficiente en el registro de las puntuaciones en un archivo Json y asociarlo con el respectivo jugador.
3. Se logró el registro de los ¨nicknames ¨ en un archivo Json.
4. También se logro cargar los mapas para cada nivel.

# Bitácora de actividades.

|  |  |
| --- | --- |
| **Actividades.** | **Horas.** |
| Análisis de requerimientos. | 5 |
| Diseño de aplicación y diagrama de clases. | 26 |
| Investigación de funciones. | 25 |
| Programación. | 33 |
| Documentación Interna. | 3 |
| Pruebas. | 10 |
| Elaboración del documento | 5 |
| **TOTAL** | 107 |

# Conclusión.

Gracias a las diferentes interfaces graficas como lo son Tkinter y Pygame, se puede crear diversos proyectos como este. En los cuales, se desarrollan la programación orientada a objetos, con las cuales, se aprende sobre su uso, el manejo de clases, herencia y funciones. Por otro lado, se desarrolló el manejo de archivos Json para guardar diversos tipos de información.