****

PRIMER 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. 5](#_Toc7385978)

[Análisis de resultados. 6](#_Toc7385979)

[Bitácora de actividades. 7](#_Toc7385980)

[Conclusión. 8](#_Toc7385981)

# Introducción

Este proyecto trata de la creación de un código para el juego Space Invaders, el juego inspirado en las películas como la Guerra de las Galaxias y La guerra de los Mundos, este fue uno de los primeros en utilizar disparos para atacar y defenderse de los rivales.

El juego se desenvuelve de esta forma, las naves atacan de forma masiva a la nave defensora, conforme las naves invasoras disparan, se acercan de forma progresiva a la nave del jugador por esto el jugador tiene que tratar de esquivarlas y no solo a estas, sino que también a los meteoritos he intentar destruir a las naves invasoras.

# Descripción del problema.

El juego tiene que iniciar con un grupo de 30 naves invasoras que se encuentran alineadas en la parte superior de la pantalla y la nave defensora, que se encuentra en la parte inferior. Cuando el juego inicia el bloque de naves intrusas se tiene que mover de forma constante de izquierda a derecha y comenzar a desprenderse, una a una, de forma aleatoria, a la vez que varios meteoritos en movimiento se mueven en dirección de la nave defensora. Las naves que se desprenden del bloque de naves invasoras se mueven en dirección de la nave protectora mientras disparan de forma continua, por lo que esta última debe moverse para evadir los disparos y las colisiones con las atacantes y los meteoritos. También debe permitir la creación de jugadores y el almacenamiento de los resultados de los jugadores, de forma que cuando el juego no ha comenzado aparecerá una pantalla que mostrará los 5 mejores resultados que han sido obtenidos por los jugadores.

El se tiene que calcular el puntaje obtenido por cada jugador se hará en función del número de naves invasoras y meteoritos que la nave defensora logra derrotar o demoler con disparos (el jugador obtiene un punto por cada una nave o meteorito destruido). La nave defensora tiene solo una vida y en caso de ser destruida por un disparo, el jugador debe comenzar una nueva partida. El juego tiene tres niveles de dificultad, inicial, intermedio y avanzado. Los niveles superiores se acceden una vez que los niveles inferiores han sido superados.

# Descripción de la solución con UML.

Main

Jugador

Enemigos

Bala

Enemigos

Jugador

* def dibujar
* def comportamientoImages
* def \_\_movimientos
* def \_\_descenso
* def \_\_movimientolateral
* def \_\_ataque
* def \_\_disparo
* def movimientoDerecha
* def movimientoIzquierda
* def \_\_movimiento
* def destrucción
* def disparar
* def dibujar

Bala

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

# Análisis de resultados.

Se logro hacer los enemigos del juego, la nave defensora o la del jugador, los disparos del jugador que son con la tecla ‘s’ , los enemigos disparan aleatoriamente y los meteoritos. No se logro hacer el paso al siguiente nivel.

# Bitácora de actividades.

|  |  |
| --- | --- |
| **Actividades.** | **Horas.** |
| Análisis de requerimientos. | 3 |
| Diseño de aplicación y diagrama de clases. | 21 |
| Investigación de funciones. | 13 |
| Programación. | 20 |
| Documentación Interna. | 2 |
| Pruebas. | 6 |
| Elaboración del documento | 4 |
| **TOTAL** | 69 |

# Conclusión.

En conclusión con este proyecto ahora manejamos mejor las clases, se aprendió mas sobre pygame y las funciones para crear jugadores y sus movimientos el importar funciones y todas las variables que se necesitan para la orientación de los objetos.