Documentación Externa

Proyecto #1

Taller de Programación Gr4



Mario Carranza Castillo

2019180212

Álvaro González Ballesteros

2019379863



Tabla de Contenidos

**Introducción ………………………………………………………………. 3**

**Descripción del Problema ………………………………………………. 4**

**Diagrama UML …………………………………………………………….. 5**

**Análisis de Resultados ………………………………………………….. 6**

**Bitácora de actividades …………………………………………………. 7**

**Estadísticas de Tiempos ………………………………………………... 11**

**Conclusión ………………………………………………………………… 12**

Introducción

Con el presente proyecto se pretende la creación de un juego en 2D de naves espaciales, por medio del uso de los programas Tkinter y Pygame destinados en gran parte para tal fin. Para efectos de su realización, se estarán utilizando ambos programas.

Se pretende, con el trabajo en este video juego, adquirir, por medio de investigaciones, los conocimientos necesarios para poder trabajar con los programas TKinter y Pygame en la creación y ejecución de video juegos en 2D. Inicialmente se tuvo que realizar una amplia investigación sobre la forma en que ambos programas trabajan, por ejemplo, el lenguaje que utilizan y los diferentes algoritmos que pueden usar para cargar imágenes, sonidos, etc.

Tras la investigación, pudimos darnos cuenta de que, al ser Pygame un lenguaje más ágil y sencillo de comprender, y además de que posee funciones más orientadas hacia la programación de juegos, y tomando en consideración que con TKinter se puede lograr obtener una mejor interfaz en la ventana de inicio del juego, podíamos utilizar ambos programas y unirlos de manera que el juego quede mejor estructurado estéticamente en comparación a si solo se hubiese utilizado uno. Debido a esta observación, se decidió que utilizaríamos ambos programas, para así lograr obtener una mejor interfaz en la pantalla de inicio del juego y además una mejor y más fácil ejecución del juego principal.

Durante el trabajo se reqirió de la inversion de mucho tiempo y dedicación, ya que, al tener que empezar practicamente desde cero con un programa nuevo, es un proceso de prueba y error, en el cuál, durante el momento de trabajo y la aplicación de lo investigado, se aprende gran cantidad de cosas nuevas que servirán de mucho en proyectos futuros.

Con todo lo realizado durante este proyecto se ve la gran posibilidad de crear juegos a partir de programas con lenguajes sencillos, como lo son Pygame y Tkinter, y que aunque no tienen las funciones necesarias para hacer trabajos más elaborados como juegos en 3D, son una muy buena forma de empezar a entender y adentrarse en el mundo se los video juegos.

Descripción del Problema

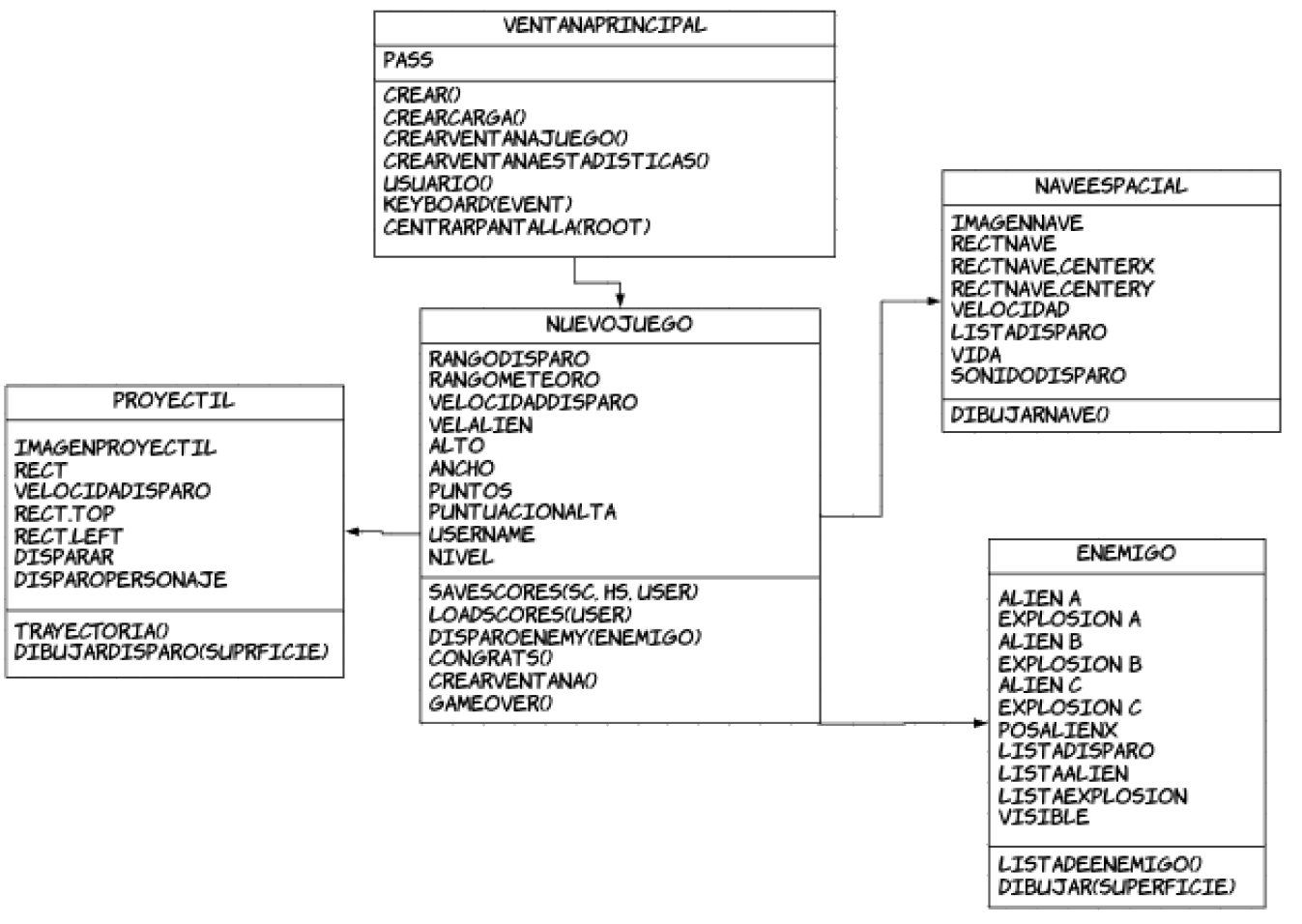
Crear un juego (Space Invaders) de 30 naves invasoras o aliens que se encuentran alineadas en la parte superior de la pantalla y una nave defensora, que se encuentra en la parte inferior de modo que la nave defensora debe destruir los aliens, el bloque de naves intrusas se mueve de forma constante de izquierda a derecha y se comienzan a desprender, una a una, las naves que se desprenden del bloque de naves invasoras se mueven en dirección de la nave protectora mientras disparan de forma aleatoria, a la vez que varios meteoritos en movimiento se mueven en dirección de la nave defensora, por lo que la nave defensora debe moverse para evadir los disparos y las colisiones con las atacantes y los meteoritos.

El juego debe permitir:

* La creación de jugadores y el almacenamiento de los resultados de los jugadores, y mostrará los 5 mejores resultados obtenidos por los jugadores.
* Crear usuarios nuevos o seleccionarlos de una lista cuando estos se encuentren registrados. Los usuarios deben ser almacenados en un archivo CSV.
* Almacenar el resultado de los 5 mejores jugadores en un archivo JSON.
* Presentar una pantalla con el resultado de los 5 mejores jugadores, antes de comenzar una partida.

El cálculo del puntaje obtenido por cada jugador se calculará en función del número de naves invasoras y meteoritos que la nave defensora logra destruir y se 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.

Diagrama UML

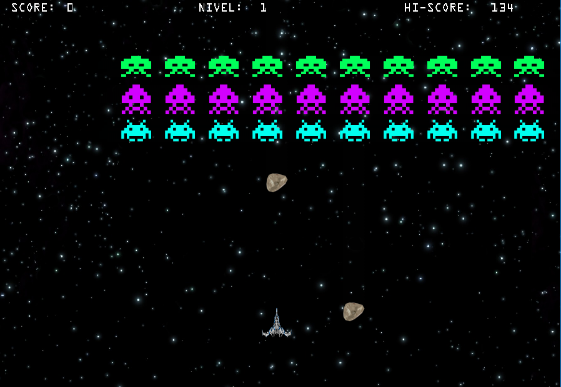
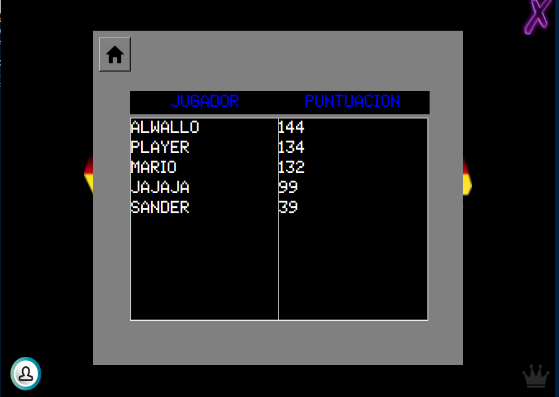


Análisis de Resultados

Llegando a la conclusión del proyecto, aprendimos una serie de cosas acerca de los módulos Tkinter y Pygame de Python, sus múltiples funciones y métodos, trabajar con ambos nos resulto mejor para tener una buena interfaz en la ventana inicial con Tkinter y una fácil y creativa ventana para el juego.

También al trabajar con archivos CSV y JSON se facilitó el manejo de estadísticas de juego al guardar las puntuaciones en CSV y las 5 mejores puntuaciones con JSON. Gracias a esto los resultados fueron satisfactorios en su mayor parte y se lograron la mayoría de objetivos.





Bitácora de actividades

**Sábado 6 y domingo 7 de abril:**

Ambos estudiantes empezamos a estudiar cada uno de los programas (TKinter y Pygame), con el fin de obtener el conocimiento necesario, para posteriormente dar inicio a la realización del juego.

Duración de la actividad aproximada: 4 horas.

**Lunes 8 de abril:**

Dentro del código utilizando Pygame, se comenzó buscando todas las imágenes que íbamos a necesitar para la creación del juego, como la imagen de la nave espacial defensora, las imágenes de los aliens, las de los disparos, etc. Se trabajó en la creación de la ventana del juego (sus dimensiones), y se cargaron las imágenes a utilizar en el código, para lo que se definió un ciclo “while” en el que nos apoyaríamos para ejecutar todas las acciones del juego.

Para el código en TKinter, se buscaron las imágenes a utilizar, se construyó la ventana destinada a ser la de inicio del juego, se cargaron las imágenes para tal fin, se creó un botón dentro de esa ventana llamado “START” destinado a ser el botón que de inicio al juego y se creó una pantalla en la cual se cargó una imagen con el mensaje “LOADING”, lo que luego daría paso a la ejecución del juego.

Duración de la actividad aproximada: 5 horas.

**Jueves 11 de abril:**

Dentro del código en Pygame, se trabajó en darle movimiento tanto a la nave defensora como a los aliens. Se trabajó también en la impresión en pantalla de un total de 30 enemigos, todo eso con la creación de dos funciones, una para la nave espacial defensora y otra para los enemigos, las cuales luego serían llamadas dentro del ciclo “while” ya definido. Para terminar, se imprimió texto dentro de la ventana del juego con los mensajes “SCORE” y “HI-SCORE” con el fin de usar esos dos espacios para llevar la cuenta de la puntuación durante el juego y la mayor puntuación alcanzada.

Para el código en Tkinter, se hizo funcionar el botón de inicio del juego creado en la ventana principal para que se cargara la pantalla con la imagen “LOADING” y luego la pantalla del juego.

Duración de la actividad aproximada: 12 horas.

**Lunes 15 de abril:**

Para el código de Pygame, se creó una clase llamada “NuevoJuego”, la cual contendría todo el código del juego. Además, se crearon dos clases más, una para la nave espacial, una para los disparos o proyectiles, tanto para los de la nave espacial como para los de los enemigos.

Duración de la actividad aproximada: 10 horas.

**Jueves 18 de abril:**

En el código de Pygame, se logró hacer que los enemigos disparen contra la nave espacial.

Duración de la actividad aproximada: 6 horas.

En el código de TKinter, se crearon tres botones en la pantalla principal; uno para que al presionarlo el programa permita salir del juego, otro que contiene los 5 puntajes más altos registrados y el otro para que se pueda crear un nuevo jugador.

Duración de la actividad aproximada: 5 horas.

**Viernes 19 de abril:**

En el código de Pygame, se creó una clase para los enemigos utilizando como referencia la función que ya había sido establecida para enemigos.

Duración de la actividad aproximada: 2 horas.

En el código de Tkinter, se creó un archivo CSV para guardar los nuevos perfiles de los jugadores que se crean en el botón de crear nuevo jugador.

Duración de la actividad aproximada: 3 horas.

**Sábado 20 de abril:**

En el código de Pygame, se modificó la clase de los enemigos al hacer que los enemigos estén dentro de una matriz con el fin de lograr que disparen aleatoriamente y al momento de que se les dispare, se eliminen de su posición respectiva dentro de la matriz, y se utilizaron números aleatorios para el disparo de los enemigos.

Duración de la actividad aproximada: 5 horas.

**Domingo 21 de abril:**

En el código de Pygame, se validaron condiciones para controlar las colisiones de los rectángulos de las imágenes, se agregaron efectos de sonido a los disparos de la nave defensora.

Se hizo una clase para el comportamiento de los meteoritos, en la que además se cargan las imágenes de los meteoritos.

Duración de la actividad aproximada: 6 horas.

**Lunes 22 de abril:**

En el código de Pygame, se hizo una condición para detener el juego en caso de que el jugador pierda. Se hizo que se sume un punto por cada enemigo o meteorito destruido y que se guarde el puntaje obtenido de cada jugador. Se hizo que se actualice el “HI-SCORE” de acuerdo con el puntaje del jugador, si éste es mayor que el “HI-SCORE” registrado.

Se creó un archivo JSON con el fin de poder guardar las puntuaciones más altas de los jugadores.

Se añadió un efecto de sonido a la destrucción de los enemigos y se perfeccionó la clase de meteoritos para que se dibujen en la ventana del juego en posiciones diferentes.

Duración de la actividad aproximada: 10 horas.

**Martes 23 de abril:**

En el código Pygame se corrigieron todos los errores, se validaron parámetros, se crearon los nuevos niveles para el juego, se agregó un fondo para cuando se terminen los niveles y se logre ganar, se agregó sonido de victoria y se dieron los últimos retoques al código.

Se terminó la documentación interna y externa.

Duración de la actividad aproximada: 8 horas.

Estadísticas de Tiempos

|  |  |
| --- | --- |
| Análisis de requerimientos | 1 horas |
| Diseño de la aplicación y diagrama de clases | 22 horas |
| Investigación de funciones | 3 horas |
| Programación | 25 horas |
| Documentación interna | 4 horas |
| Pruebas | 22 horas |
| Elaboración documento | 4 horas |
| **TOTAL** | 81 horas |

Conclusiones

* Se logro terminar el proyecto según los parámetros indicados.
* Se reflejaron las horas trabajadas en un cuadro y en la bitácora.
* Se logro el aprendizaje de los módulos Pygame y Tkinter asi como otros módulos para leer y escribir en archivos CSV y JSON.