PROYECTO FINAL: SPACE INVADERS (JUEGO ARCADE)

Nombre y Apellidos: Fátima Fuchun Illana Guerra

Grado: Ciencia de Datos e Inteligencia Artificial

Nombre del programa principal: SpaceInvaders.py

DESCRIPCIÓN GENERAL DEL JUEGO:

El programa desarrollado consiste en un juego gráfico de tipo arcade, implementado en Python, basado en el ya conocido "Space Invaders", que cuenta también con un menú principal, un menú secundario y una pantalla final, así como la posibilidad de guardar la puntuación obtenida, asociada al nombre del jugador.

DESCRIPCIÓN DETALLADA DEL JUEGO:

Para empezar, el nombre del jugador se pide al inicio, incluso antes de abrir la pantalla de pygame, de forma que se guarda y, una vez finalizada la partida, se le asocia la puntuación obtenida y ambos datos son almacenados en un archivo externo. El nombre de un usuario se puede repetir indefinidamente, ya que de esta forma un mismo jugador puede almacenar diversas puntuaciones tras jugar distintas partidas. Además, si el usuario no inserta ningún nombre, se establecerá uno predeterminado llamado "Jugador".

El menú inicial contará con las dos opciones básicas: iniciar el juego o salir del juego, representadas mediante dos botones mostrados en pantalla, que el usuario tendrá que pulsar con el ratón.

El menú secundario volverá a presentar una serie de opciones al usuario, esta vez con tres niveles distintos, según su dificultad. Cuanto más alto sea el nivel elegido a más velocidad irán apareciendo los enemigos y, por ende, más difícil será eliminarlos. De nuevo, las tres opciones estarán representadas por un botón cada una, el cual el usuario deberá pulsar con el ratón. Además, en este segundo menú existe otro cuarto botón que abre una tercera pantalla auxiliar con las instrucciones del juego, y un botón de retorno a la pantalla con la elección de niveles.

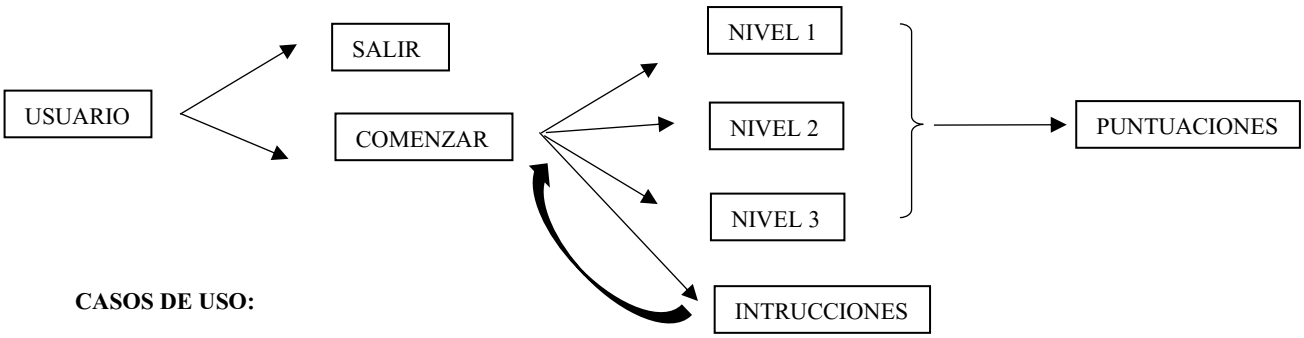
Una vez elegido el nivel, se abrirá finalmente la pantalla de juego en la que los enemigos irán saliendo del borde superior y bajando en el eje "y" hacia el avatar del jugador. El usuario tendrá entonces que usar las flechas de izquierda y derecha para mover la nave en el eje "x", y el botón del espacio en el teclado para disparar los proyectiles que eliminarán a los enemigos. Cada vez que un enemigo es eliminado el contador incrementará en un punto, pero otro será generado inmediatamente. Además, si los enemigos logran cruzar el borde inferior, es decir, si el jugador no logra eliminarlos, se restarán puntos a su contador. El juego finaliza cuando algún enemigo logra colisionar contra el propio jugador.

Cuando esto ocurra el juego se cerrará inmediatamente y aparecerá la última pantalla, con el encabezado "GAME OVER", y las cinco puntuaciones más altas almacenadas, junto al nombre de sus usuarios correspondientes.

El jugador puede, en cualquier momento, abandonar el juego si pulsa la cruz en la esquina derecha de la pantalla. No obstante, si decide salir en plena partida, o antes, ninguno, ni su nombre ni su puntuación, se guardarán.

Se pueden obtener puntuaciones negativas.

ESQUEMA DE LOS CASOS DE USO:



CASOS DE USO:

Caso de uso “salir”:

1. El programa cierra la ventana pygame.

Caso de uso “comenzar”:

1. El programa abre el menú secundario y pregunta por el nivel de dificultad a escoger.

Caso de uso “nivel 1”:

1. El programa inicia los frames/segundo en 30.
2. Inicia el juego.

Caso de uso “nivel 2”:

1. El programa inicia los frames/segundo en 40.
2. Inicia el juego.

Caso de uso “nivel 3”:

1. El programa inicia los frames/segundo en 50.
2. Inicia el juego.

Caso de uso “Instrucciones”:

1. Imprime las instrucciones del archivo “instrucciones”.

DISEÑO:

Para empezar, desarrollar el proyecto se ha importado la librería pygame, el módulo “sys”, “random”, listas, y un módulo creado específicamente para el juego llamado “cinco_máximos”, que se explicará posteriormente.

En la cabecera del programa definimos todas las constantes que usaremos, y definimos e iniciamos algunas herramientas que también serán fundamentales para el desarrollo del programa, como los colores, algunas imágenes, la tipografía, los sonidos y la música, las variables booleanas que inician los bucles de las pantallas y la ventana pygame. Creamos también las tres listas que usaremos para almacenar los tres objetos principales, del tipo “Sprite”: jugador, enemigos y disparos. Una de las listas contiene solo a los enemigos, otra solo a los disparos y otra a todos los Sprites del programa.

A continuación, definimos todas las clases de objetos (del tipo Sprite) que vamos a usar:

- La clase Botón estará compuesta por superficies rectangulares, con una imagen determinada, y contará con las operaciones “mostrar,” que dibujará el botón correspondiente en la ventana, y la operación “pulsado” que determinará si el botón está siendo o no pulsado.

- Después está la clase Barrera, que estará compuesta por un único objeto rectangular, igual de ancho que la pantalla y situado tras el borde inferior de la pantalla, usado para detectar a aquellos enemigos que el jugador no logre eliminar.
- La clase Disparo está compuesta también por superficies rectangulares, pequeñas, que se ven moviendo el eje y (método "update").
- La clase Enemigo contiene a los enemigos, cuyo método "update" incrementa el eje y, para crear movimiento.
- La clase Jugador contiene al avatar del usuario. Su método "update" contiene todos los comandos del jugador, y detecta el movimiento del avatar en función de los botones pulsados. Además, cuenta también con el método "disparar" que detecta si se ha pulsado la barra espaciadora y, en caso afirmativo, crea un objeto de la clase Disparo.

Después tenemos las funciones que usaremos:

- La función "crear_texto", obtiene el texto a escribir, la fuente a utilizar, el color de las letras, y la posición en la que escribir el texto, y lo muestra en la pantalla.
- La función "imprimir fichero" imprime en pantalla el contenido de un archivo.
- La función "colisiones_sprite_enemigo" detecta la colisión entre cualquier Sprite y cualquier enemigo, que están almacenados en una lista de Sprites. Si colisionan, se elimina al enemigo de todas las listas a las que pertenece. Ocurre lo mismo con la función "colisiones_disparo_enemigo" solo que entre dos grupos de Sprites, los enemigos y los disparos.
- "crear_enemigos" crea un número n de enemigos y los añade a la lista, lista_enemigos.

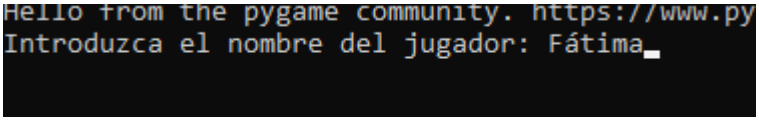
Algunas herramientas específicas que se usan en el programa:

- "pygame.mixer" para cargar sonidos y música.
- "pygame.sprite.spritecollide" y "pygame.sprite.groupcollide" para detector las colisiones entre un Sprite y un grupo de Sprites, y dos grupos de Sprites.
- ".rstrip()" para eliminar los saltos de línea al leer un archivo.

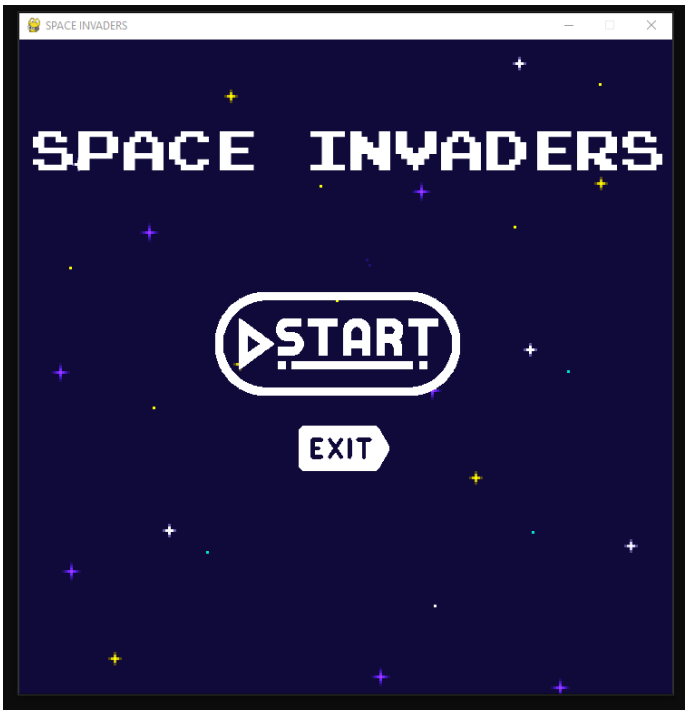
Módulo utilizado: "cinco_máximos". Este programa independiente del principal sirve para recorrer el archivo de puntuaciones y devolver las cinco más altas, junto a sus jugadores correspondientes. Está compuesto por 6 funciones distintas, cada una apoyándose en la anterior, más la principal que es "ganadores". Las tres primeras simulan el método de selección, implementado en la práctica 11, solo que devolviendo la lista decreciente en vez de la creciente.
Después, la función "lista_split" recorre el archivo, leyéndolo línea a línea, y crea una lista compuesta por las listas resultantes de aplicar la función predeterminada "Split" a cada línea. A continuación, la función "lista_puntuaciones" recorre la lista que devuleve "lista_split" y crea otra aparte, únicamente con las puntuaciones. Se le aplica entonces el método de selección y la función "cinco_mayores", que solo devuelve los cinco primeros valores de esa lista, que serán las cinco puntuaciones más altas.
Y, por último, la función "ganadores" recorre ambas listas, la que contiene a los cinco valores más altos, y a la creada inicialmente por "lista_split", escogiendo a los cinco primeros usuarios cuya puntuación sea igual a los 5 valores más altos.

PROGRAMA EN EJECUCIÓN:

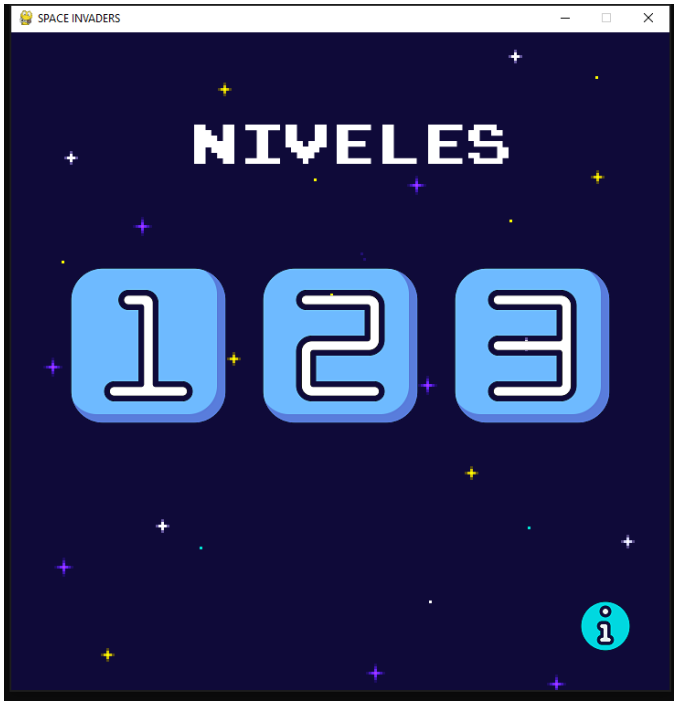
1- El programa pide al usuario que introduzca su nombre.



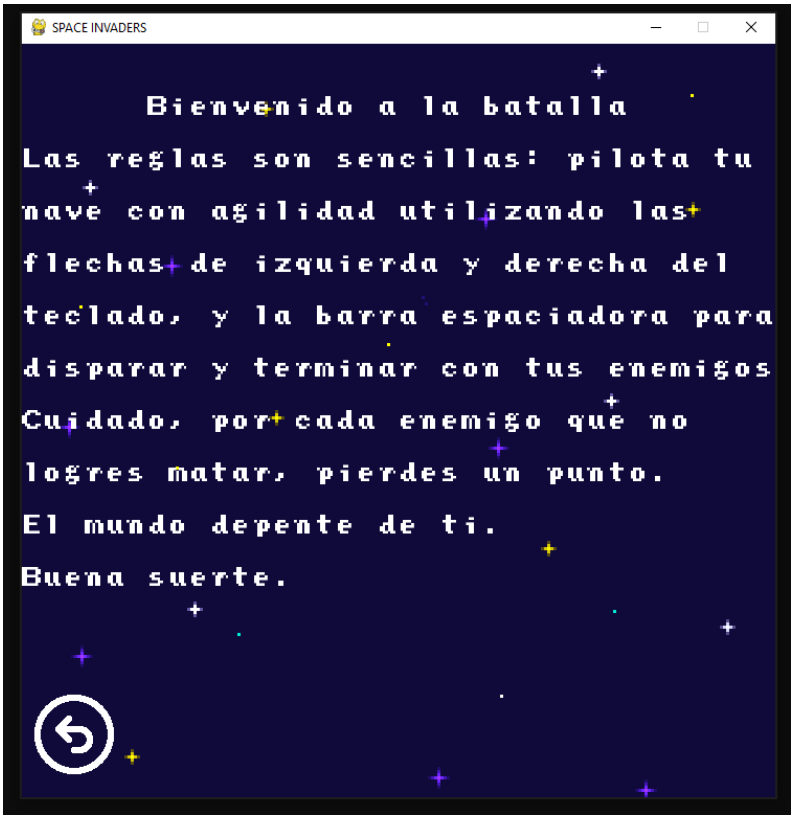
2- Menú principal:



3- Menú secundario:



4 – Instrucciones:



5- Juego:

6- Pantalla Game Over con puntuaciones:

REFERENCIAS:

1- La canción utilizada en el juego se titula “Back on Track” de DJVI, utilizada en el juego “Geometry Dash”

2- Todas las imágenes están sacadas de la página web “flaticon.es”