DOCUMENTACION

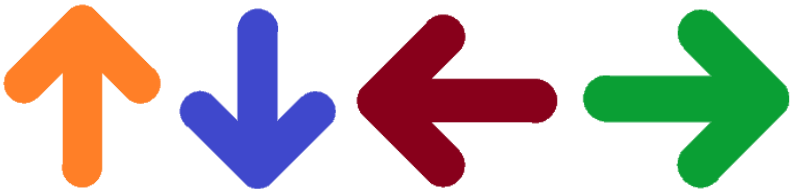
INTRODUCCIÓN

Simon es un juego electrónico creado por Ralph Baer y Howard J. Morrison en 1978. Su gran éxito fue durante los 80. Tiene forma de disco, en una de sus caras se puede ver cuatro cuadrantes, cada uno con un color: **verde**, **rojo**, **azul** y **amarillo** en su versión original. Su nombre se debe por el conocido juego tradicional del mismo nombre: Simón dice, de donde se inspira.

El juego de forma aleatoria va iluminando los cuadrantes de colores, y a la vez que se ilumina cada cuadrante emite un sonido propio. Después de esperar, el usuario debe ir introduciendo la secuencia mostrada en el orden correcto, ayudándose de su memoria visual y sonora.

Si lo consigue, éste responderá con una secuencia más larga, y así sucesivamente. Si falla, el usuario debe volver a empezar.

Para el proyecto se hizo la emulación de dicho juego, pero con la variante de no mostrar cuadrantes, sino imágenes de flechas, las cuales irá presionando a través del teclado para repetir la secuencia.



DESCRIPCION DEL JUEGO

- ¿CÓMO FUNCIONA?

El juego es muy sencillo, básicamente inicia con **SIMON**, quien muestra poco a poco la secuencia a repetir.

Empieza mostrando la primera dirección de flecha, posteriormente tú la repites con la respectiva flecha del teclado. En caso de ser la flecha adecuada **SIMON** te vuelve a mostrar la secuencia a repetir, pero con un elemento más.

Los elementos posibles sólo son cuatro ⇧, ⇩, ⇦, ⇨.

El juego termina hasta que te equivoques en la secuencia, sino sigue mostrando cada vez más elementos.

- ¿CUÁLES SON LAS REGLAS?

Solo hay una regla:

- ❖ Haz lo que **SIMON** dice.

IMPLEMENTACIÓN DEL PROGRAMA

- DESCRIPCIÓN DEL ALGORITMO

El algoritmo se apoya de las funciones de pygame fundamentalmente.

Primero que nada, todo el juego es un constante loop que sólo termina cuando el algoritmo determina que el usuario se equivoca y por lo tanto pierde, o si hace click en la x de la ventana.

Dentro de dicho loop existen condicionales que controlan los turnos, pues mientras es el turno de **SIMON** se muestra la secuencia, pero si este termina, entonces ya es turno de que el usuario emule la secuencia con las teclas.
Para esta hazaña hacemos uso de los eventos que pygame nos proporciona y evaluamos que si presiona alguna tecla le asignamos un valor si la tecla pertenece a las flechas y comparamos con la lista de **SIMON** para determinar si corresponde a lo que simon dice.

En caso de ser verdad continua el ciclo hasta que se apriete la siguiente tecla y evalué los valores con **SIMON**. En el caso de equivocarse muestra un último mensaje de "Perdiste" y se indica el fin del ciclo para posteriormente cerrar la ventana, si es que termina la secuencia que simon dijo, entonces viene el turno de **SIMON** y se le agrega un nuevo numero aleatorio a la lista de **SIMON**.

Para mostrar las imágenes de la secuencia de **SIMON** se hace uso de una variable *imagen* que apoya como auxiliar para saber que mostrará durante cada ciclo que se realice, además de un retraso con *time.sleep()* para que de tiempo de que el usuario observe la imagen, y el texto.

También para llevar un control del orden a mostrar se hacen uso de contadores tanto para la secuencia de **SIMON** como para contar los intentos que el usuario realiza por cada tecla que presiona.

Con dichos contadores y los turnos se evalúa el momento en que se debe hacer una pausa para mostrar la respuesta de la interacción o "limpieza" de la pantalla para ir al siguiente turno, o inclusive mostrar una imagen de que la secuencia estuvo completamente bien.

Finalmente cabe mencionar que se hace uso de funciones de pygame para ir mostrando en el display o la ventana del juego, así como para ir actualizando lo que se muestra al usuario. Además de un reloj para declarar los frames por segundo en que se desarrollará el juego.

- DESCRIPCIÓN DE FUNCIONES

Dentro del juego se hace uso de muchas funciones de otras bibliotecas, las cuales ya están documentadas, así que las omitiré para dar una mejor descripción de las funciones que yo definí.

- mostrar(imgObj*, x, y)*

Esta función simplemente renombra la función *gameDisplay.blit()* que posiciona una superficie, en este caso una imagen con las coordenadas en (x, y) a partir del origen que se encuentra en la parte superior derecha de la pantalla. Y dibuja la parte superior derecha de la imagen a partir de las coordenadas dadas.

- text_objects(text, font)

Esta función retorna la superficie creada a partir de un texto y su fuente, así como el tamaño y coordenadas creadas para dicha superficie. Con el uso de la función *font.render()* la cual necesita como parámetros el texto, si será en itálicas, y el color.

- message_display(text, centerX, centerY)

Esta función muestra un texto en la ventana a partir de sus coordenadas por un tiempo de 1 segundo. Para ello transforma el texto recibido como parámetro en una superficie y centra el área de la superficie en las coordenadas dichas, para posteriormente hacer uso de gameDisplay.blit() e imprimir la superficie con las coordenadas dadas por el rectángulo de la superficie.

- game_loop()

La función más importante del programa, pues esta función lleva todo el algoritmo del juego.
Esta función realiza un loop, el cual solo termina cuando el algoritmo determina que el usuario perdió. Dentro del loop, que en realidad es un while enorme, realiza las acciones de **SIMON** cuando es su turno y evalúa las acciones del usuario cuando llega el suyo.

- DESCRIPCIÓN DE BIBLIOTECAS

Para el desarrollo del programa se hizo uso de 3 bibliotecas: *pygame*, *random* y *time*.

- **PYGAME**

Pygame es una biblioteca que proporciona múltiples funciones y objetos para el desarrollo de videojuegos, facilitando la interacción con el usuario a través de eventos, así como el desarrollo de una interfaz gráfica a través de imágenes, dibujos y el posicionamiento de los mismos.

- **RANDOM**

Random es una biblioteca que proporciona funciones para el desarrollo de números o decisiones aparentemente aleatorias.

- **TIME**

Time es una biblioteca que nos otorga la posibilidad del manejo del tiempo con funciones como sleep, strftime, timezone. Con lo cual nos permite saber el tiempo actual, pasado o futuro con operaciones a través de los segundos, así como la posibilidad de hacer “pausas” con él.

CONCLUSIONES

El desarrollo de este juego permite que se aprendan nuevas formas de resolver un problema, en este caso la interacción con el usuario, la cuál es cada vez más importante pues a cada segundo este puede realizar alguna acción que el programador no contempla. Por lo cual es importante limitarlo lo más que se pueda en cuanto a las acciones que no debe realizar, así como proporcionarle la manera de comunicarse con el programa. También resalta las diferentes formas que puede existir para resolver un problema en especial, todo radica en el conocimiento que se tenga sobre el lenguaje, así como las herramientas o en este caso bibliotecas o módulos que se tengan a la mano para apoyarte en la resolución de dicho problema.