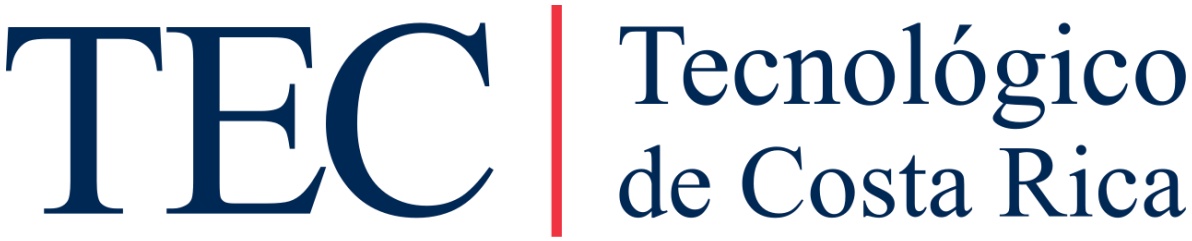
TECNOLÓGICO DE COSTA RICA

LICENCIATURA EN INGENIERÍA EN COMPUTADORES

INTRODUCCIÓN A LA PROGRAMACIÓN (CE1101)



Proyecto I – Vintage Bomberman Game

Documentación del Proyecto

Autor:

Steven Aguilar Alvarez

Profesor:

Jason Leitón Jiménez

viernes, 19 de abril de 2024

Tabla de Contenido

1. Introducción

Este proyecto del 'Vintage Bomberman Game' ha sido una experiencia emocionante desafiante para mí. Utilizando Python y Tkinter, he puesto en práctica mis conocimientos en programación para crear un juego clásico con una interfaz gráfica atractiva. El objetivo de este proyecto va más allá de simplemente desarrollar un juego; se trata de aplicar los fundamentos de la programación de manera creativa y funcional.

A lo largo de las semanas dedicadas a este proyecto, he invertido horas significativas en cada fase, desde el diseño inicial de la interfaz hasta la implementación de mecánicas como el movimiento del personaje, la generación de enemigos y la colocación estratégica de explosivos. Cada etapa ha representado un desafío único que he abordado con entusiasmo y determinación.

Este proyecto me ha permitido mejorar mis habilidades en programación, especialmente en el uso de la recursividad y la gestión de interfaces gráficas. Sin embargo, también he identificado áreas en las que puedo seguir creciendo, como la optimización del código y el aprovechamiento completo de las capacidades de Tkinter. Esta experiencia me ha brindado no solo un juego funcional, sino también un aprendizaje invaluable que aplicaré en futuros proyectos.

2. Descripción del Problema

El desafío principal de este proyecto fue crear un juego completo y funcional inspirado en el clásico 'Bomberman', utilizando Python y Tkinter. Este juego implica la integración de varios aspectos de la programación, desde el manejo de datos hasta la implementación de mecánicas de juego complejas y una interfaz gráfica atractiva.

Uno de los puntos clave fue diseñar un sistema de laberintos dinámicos, donde los jugadores pudieran moverse libremente, colocar bombas estratégicamente y enfrentarse a obstáculos y enemigos. La recursividad fue fundamental para abordar aspectos como la propagación de las explosiones y la generación dinámica de los niveles del laberinto.

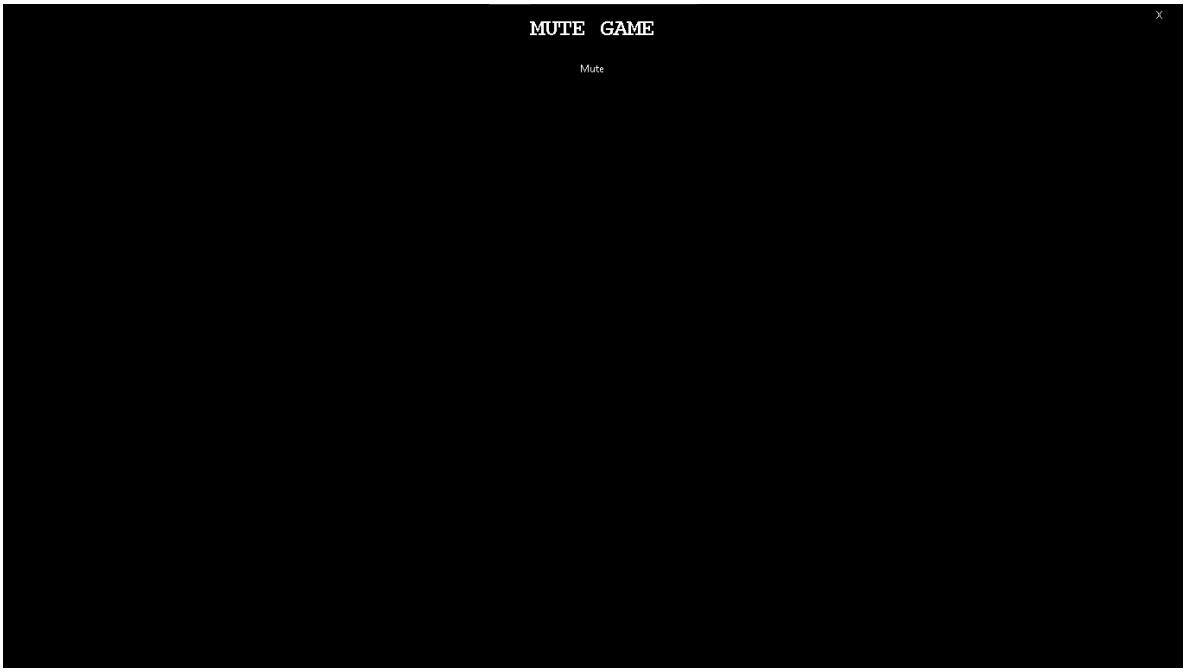
Además, se trabajó en la implementación de elementos como la vida del jugador, la búsqueda de una llave oculta para abrir una puerta que lleva al siguiente nivel, la colocación de puntos en el mapa y la gestión de la música de fondo para mantener la experiencia de juego envolvente.

En resumen, el problema principal fue desarrollar un juego completo con mecánicas desafiantes y una interfaz intuitiva, aprovechando al máximo las capacidades de Python y Tkinter para ofrecer una experiencia de juego emocionante y entretenida.

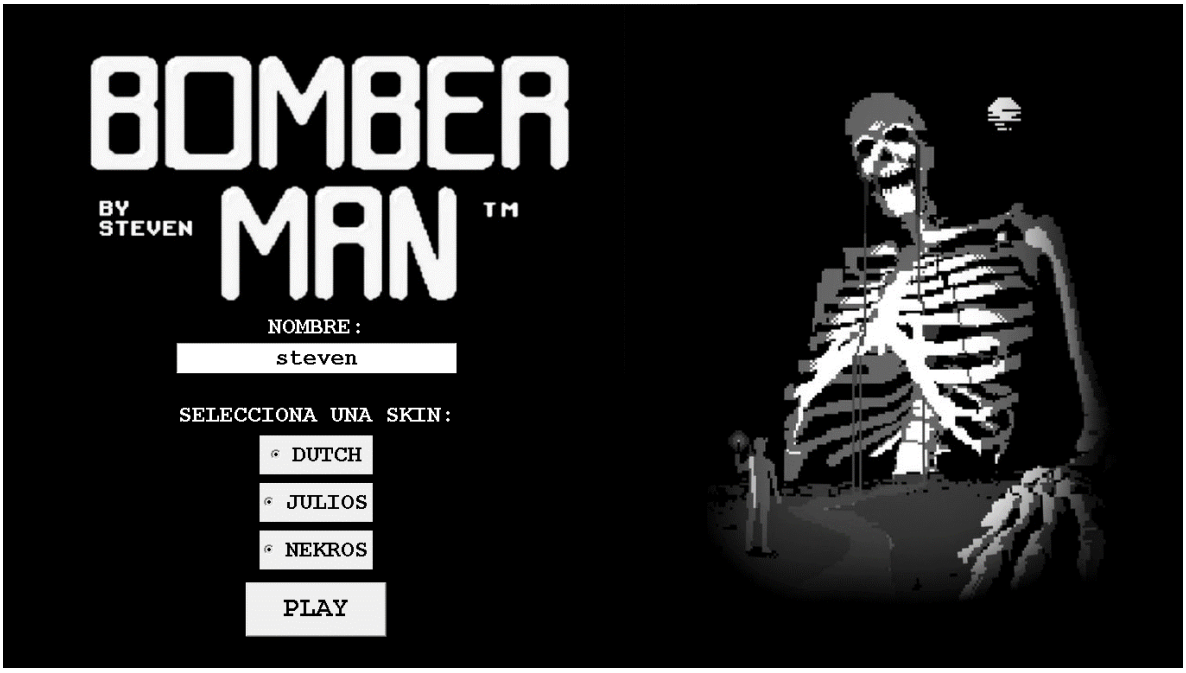
3. Análisis de Resultados

Pantalla de Inicio:

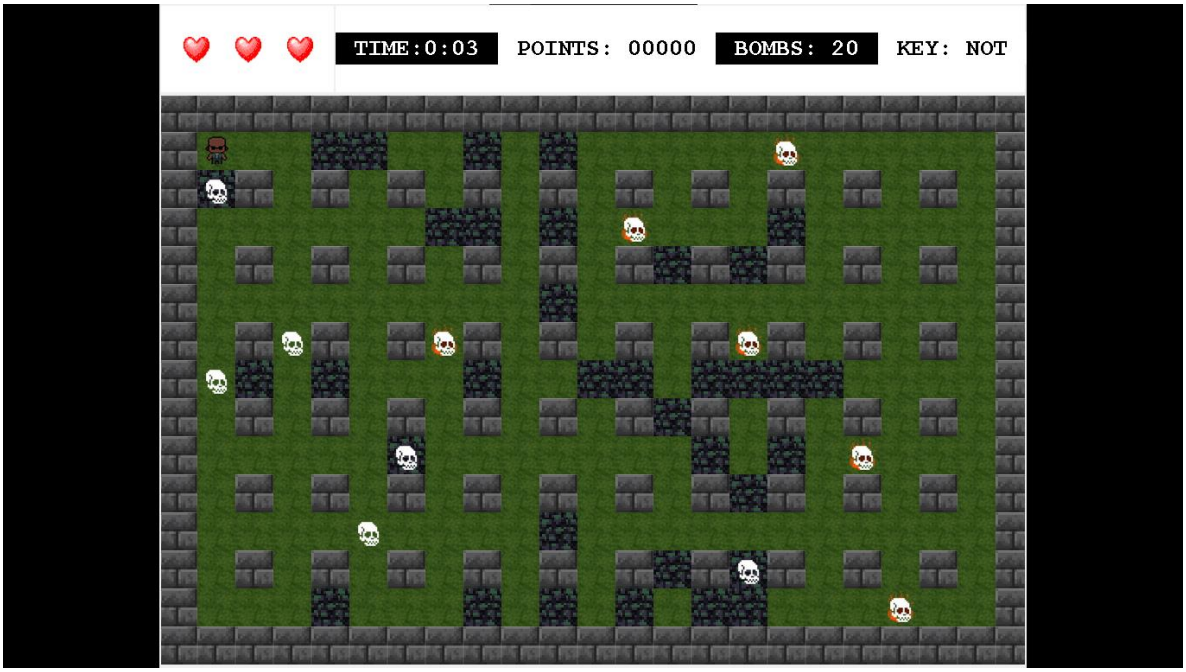
Pantalla de Configuración:



Pantalla de Personalización:



Pantalla de Juego:



Pantalla de Victoria:

Pantalla de Derrota:

Pantalla de Mejores Puntajes

4. Dificultades Encontradas

Estética del juego:

Invertí una gran cantidad de tiempo en elaborar la estética del juego, asegurándome de que todo tuviera un aspecto similar y fuera de mi agrado. Esto implicó trabajar en las imágenes, el posicionamiento, los colores y la simetría. Encontrar la combinación adecuada de estos elementos para lograr una apariencia visualmente atractiva y coherente fue un desafío significativo.

Dibujar el laberinto:

Resolver el dibujo del laberinto presentó una complejidad considerable. Sin embargo, logré abordarlo de manera efectiva utilizando una matriz y una función que dibujaba los elementos según el valor correspondiente en la matriz. Esta solución permitió una representación dinámica y adaptable del laberinto en el juego.

Movimiento del personaje:

Implementar el movimiento del personaje fue una tarea desafiante. Además de hacer que el personaje se desplazara correctamente, fue necesario cambiar la imagen según la dirección del movimiento. Esto se resolvió creando funciones separadas para cada dirección, en las cuales se verificaban las colisiones con enemigos y objetos, garantizando un movimiento fluido y coherente.

Generación de enemigos:

La generación de enemigos fue otro problema al que me enfrenté. En este aspecto, utilicé un bucle while y la biblioteca random para crear una cantidad establecida de enemigos de forma aleatoria en el mapa. Lograr una distribución equilibrada y desafiante de los enemigos requirió un cuidadoso ajuste y pruebas exhaustivas.

Colisión del personaje:

Como el juego se basa en una matriz de bloques, la detección de colisiones del personaje con enemigos y objetos fue un desafío interesante. Resumidamente, si la posición del enemigo coincidía con la del personaje, se consideraba una colisión. Implementar esta lógica de manera precisa y eficiente fue fundamental para garantizar una experiencia de juego justa y fluida.

Colocación y explosión de bombas:

La colocación de bombas en sí no fue demasiado compleja, pero la implementación de las explosiones presentó mayores dificultades. Determinar el tamaño de la explosión y su impacto en el entorno circundante requirió un cuidadoso ajuste de variables como el tamaño y la velocidad. Mediante el uso del valor absoluto del tamaño, se detectaron colisiones con el personaje según su posición actual. Después de un tiempo establecido, la animación de los fotogramas de la explosión se iniciaba, y la bomba se eliminaba del canvas.

Eliminación de enemigos y bloques:

De manera similar a la detección de colisiones con el personaje, utilicé el valor absoluto de la explosión para determinar si ésta impactaba en la posición de algún enemigo o bloque destructible. En caso afirmativo, se eliminaban del canvas, lo que permitía al jugador avanzar en el laberinto y progresar en el juego.

Manejo de puntos:

El manejo de los puntos no fue demasiado complicado. Creé una función que los aumentaba y la llamaba en las funciones donde deseaba que los puntos se incrementaran según una cantidad específica.

Generación de la llave:

La generación de la llave fue, de hecho, una tarea divertida. Primero, se buscaba en la lista de bloques destructibles y se elegía uno al azar. Luego, según un valor de probabilidad, se establecía la posición de la llave cercana o lejos del personaje. Este valor se podía ajustar según el nivel y la dificultad deseada.

Implementación del ranking:

Probablemente, lo más complicado fue implementar el ranking debido al manejo que hice del programa. Utilicé tres archivos .py: uno para la interfaz, otro para la selección de personaje y otro para el juego en sí. Esto complicó la creación del ranking, lo cual me tomó un día completo de trabajo.

El ranking funciona guardando el nombre en un archivo .txt llamado nombres.txt y los puntos en otro llamado puntos.txt. Estos datos se almacenan en un documento CSV. En la interfaz, se lee este archivo y se busca el puntaje más alto, ordenándolo con una función. El problema surgió cuando el documento se leía cada vez que se accedía a la interfaz, lo que provocaba la duplicación de la última persona que había jugado en el

ranking. Para solucionar esto, tuve que crear una función que buscara y eliminara estos nombres repetidos.

Una característica positiva es que, si se realiza una partida con un nombre específico y luego otra con el mismo nombre, no se duplica con otros puntajes, sino que se actualiza con los nuevos puntos. En la interfaz, tengo funciones que obtienen el nombre, los puntos y el ranking. La función del ranking primero elimina los repetidos, la de mostrar ranking lo ordena, y la de guardar ranking es la que repetía el guardado y duplicaba valores. En la selección de personaje, tengo la función que guarda el nombre en el archivo .txt, y en el juego, la que guarda los puntos en el archivo .txt correspondiente.

Implementación de selección de skins:

Se añadió la funcionalidad de seleccionar skins para el personaje, permitiendo al jugador personalizar la apariencia del personaje principal. La selección de skins se guarda en un archivo de texto que es leído por el código del juego, lo que proporciona flexibilidad y personalización al usuario.

Movimiento de los enemigos:

Se resolvió el problema del movimiento de los enemigos implementando un algoritmo de movimiento aleatorio. Los enemigos tipo 1 ahora se mueven de manera aleatoria por todos los espacios vacíos en el laberinto, mientras que los enemigos tipo 2 evitan los bloques destructibles y se desplazan aleatoriamente por el resto de los espacios accesibles. Esta mejora añade más desafío y dinamismo al juego, haciendo que los enemigos sean más impredecibles y difíciles de anticipar para el jugador.

Funcionalidad de Silenciamiento:

Se añadió la capacidad de silenciar todos los sonidos del juego a través de una configuración guardada en un archivo de texto. Esta configuración, representada por un valor de 0 o 1, indica si el juego está en modo silencioso o no, ofreciendo mayor control sobre la experiencia auditiva del jugador.

5. Bitácora de Actividades

Semana 1:

Diseño de la interfaz gráfica: Se elaboró la interfaz gráfica inicial utilizando la biblioteca Pygame, incluyendo la pantalla de inicio, configuración, personalización y juego. Implementación de funcionalidades: Se añadió la lógica para la música de fondo, la primera skin del personaje y la generación de enemigos.

Horas invertidas: 20 horas

Semana 2:

Reimplementación en Tkinter: Se rehízo la interfaz gráfica utilizando Tkinter y se integraron las funcionalidades previamente desarrolladas en Pygame. Se trabajó en la generación dinámica del laberinto, la posición inicial del personaje y el movimiento del mismo.

Horas invertidas: 25 horas

Semana 3:

Finalización del movimiento: Se completó la lógica para el movimiento del personaje, se implementaron los bloques destructibles, la colocación de bombas y se inició el desarrollo de la explosión de las bombas. Se trabajó en e la ventana del juego.

Horas invertidas: 25 horas

Semana 4:

Conexión de archivos y funcionalidades: Se conectaron los diferentes archivos .py del proyecto y se implementaron la generación de la llave, la posición de la puerta, la vida del personaje, la ventana de selección de personaje, el tiempo transcurrido, los puntos acumulados y las colisiones.

Horas invertidas: 25 horas

Semana 5:

Implementación de características adicionales: Se añadió el sistema de ranking, se trabajó en la implementación de más personajes, se diseñaron nuevos niveles y se pulieron detalles estéticos y funcionales del juego, además se añadieron todas las skins.

Horas invertidas: 25 horas

Semana 6:

Refinamiento del código y documentación: Se perfeccionó el código existente, se finalizó la documentación del proyecto y se optimizó el movimiento de los enemigos en el juego. Además, se completaron y ajustaron los diferentes niveles del juego para una experiencia de usuario mejorada.

Horas invertidas: 20 horas

6. Estadística de Tiempos

Semana	Actividad	Explicación	Horas Invertidadas
Semana 1	Elaboración de interfaz gráfica inicial y funcionalidades básicas	Se inició el proyecto con el diseño de la interfaz gráfica y la implementación de funcionalidades básicas como la música de fondo y la generación de enemigos.	20 horas
Semana 2	Reimplementación en Tkinter y desarrollo de lógica de juego	Se rehízo la interfaz en Tkinter y se trabajó en la lógica central del juego, incluyendo la generación del laberinto y el movimiento del personaje.	25 horas
Semana 3	Finalización del movimiento del personaje y detalles estéticos	Se completó la implementación del movimiento del personaje, se añadieron los bloques destructibles, la colocación de bombas y se trabajó en detalles estéticos de la ventana del juego.	25 horas
Semana 4	Conexión de archivos y funcionalidades principales	Se conectaron los archivos del proyecto y se implementaron funcionalidades principales como la vida del personaje y la generación de la llave.	25 horas
Semana 5	Implementación de características adicionales y pulido de detalles	Se añadieron características adicionales como el sistema de ranking y se pulieron detalles estéticos y funcionales del juego, se integraron todas las skins.	25 horas
Semana 6	Perfección y terminación de código	Se perfeccionó el código, se finalizó la documentación, se completó el movimiento de los enemigos y se implementaron los niveles del juego.	20 horas
Horas totales			140 horas

7. Conclusión

Durante el desarrollo de este proyecto, logré importantes avances en mi habilidad para programar de manera recursiva. Esta técnica me permitió abordar problemas complejos de manera más eficiente y estructurada, lo que mejoró significativamente la calidad y la lógica de mi código. Además, exploré y mejoré mi habilidad en el diseño estético de interfaces gráficas, lo cual resultó en una experiencia visual más atractiva y funcional para el usuario final.

Otro aspecto destacado de mi aprendizaje fue el manejo de archivos, especialmente con formatos como CSV y TXT. Esta experiencia me brindó una comprensión más profunda de cómo almacenar y manipular datos de manera efectiva dentro de mis programas. Además, aprender a usar Aseprite para crear skins en píxeles fue una adición valiosa a mi conjunto de habilidades, ya que pude personalizar y mejorar la apariencia de mi juego de manera más creativa.

En cuanto a las áreas de mejora identificadas, una de ellas es el orden y la estructura al usar Tkinter para interfaces gráficas. Reconozco la importancia de una organización clara y eficiente en el código, así como la capacidad de consolidar todo en un solo documento para una gestión más sencilla. Además, aunque adquirí conocimientos significativos sobre Tkinter, reconozco la necesidad de seguir profundizando en esta biblioteca para aprovechar al máximo sus capacidades y crear interfaces más complejas y dinámicas en futuros proyectos.

8. Literatura o Fuentes Consultadas

1. ChatGPT: OpenAI. (n.d.). ChatGPT. https://openai.com/chatgpt

2. CodeProjects. (2021, April 16). Juego Snake en Python con Tkinter [Video]. YouTube. https://www.youtube.com/watch?v=-bdv082X4ww&t=10s

3. Claude.AI. (n.d.). Chats. https://claude.ai/chats

4. Magno Efren. (2023, March 24). ASMR Programming - Python Cómo crear un juego de Flappy Bird en Python con Tkinter - No Talking [Video]. YouTube. https://www.youtube.com/watch?v=6oPWu7qRoCA&t=11s

5. Atlas. (Year). Working with multiple windows in tkinter [Video]. YouTube. https://www.youtube.com/watch?v=unZlSLhzNOU&pp=ygUPdGtpbnRlciB3aW5kb3dz

6. Bing Images. (n.d.). https://www.bing.com/images/create

7. GameFabrique - Bomberman NES game: GameFabrique. (n.d.). Nintendo Bomberman. https://gamefabrique.com/games/bomberman-nes/

8. BizmasterStudios. (2017, August 31). Key Icons [2D Art]. OpenGameArt. https://opengameart.org/content/key-icons

9. Cellusious. (2013, April 18). ECHRO'es [Digital image]. DeviantArt. https://www.deviantart.com/cellusious/art/ECHRO-es-366413859

10. Eduardo_gpg. (2020, 13 de julio). Manipulación de archivos con Python. PyWombat. Recuperado de https://pywombat.com/articles/mainipuacion-archivos-python

11. scizzie. (Año pasado, hace 1 año). Aquatic Ambience [Archivo de video]. YouTube. https://www.youtube.com/watch?v=DP3rDP02lE0

12. SlowedVintageMusic. (2023, septiembre 26). Dorian Concept - Hide (cs01) (mysteriorious dungeon/slowed + reverbed) [Archivo de video]. YouTube. https://www.youtube.com/watch?v=C-bTE0N5f3I

13. Karoll Nájera Molina, (2024, abril 19). Video de historia.

14. Leiton Jiménez, J. (13 de marzo de 2024). tkinter.7z [Código]. Tecnológico de Costa Rica.