Tecnológico de Costa Rica

Curso: Taller de programación

Proyecto#2: Tetris

Profesor: Cristian Campos Agüero

Estudiante: Rodney Solís Sánchez

Carnet: 2025119454

Semestre I

Grupo GR 61

Año 2025

Manual de usuario

1. Descripción General

Este proyecto consiste en una versión extendida del clásico videojuego Tetris, desarrollada en Python. El objetivo principal fue fortalecer habilidades en lógica, estructuras de control, manejo de matrices y desarrollo de interfaces gráficas.

El juego mantiene la dinámica original donde el jugador manipula piezas geométricas (tetrominós) que caen, buscando completar filas. Al completar una fila, esta se elimina y se otorgan puntos. El juego termina cuando ya no se pueden colocar más piezas.

Como mejoras adicionales a la versión clásica, esta implementación incluye:

- Obstáculos fijos para aumentar la dificultad.
- Un sistema de niveles que incrementa la velocidad de caída de las piezas conforme se acumulan puntos.
- Funcionalidad de guardar y cargar partidas usando archivos de texto.
- Registro de estadísticas del jugador, como puntaje máximo, tiempo jugado, nivel alcanzado y número de partidas.
- Un menú de navegación interactivo con una interfaz visual amigable.

Todo el sistema opera localmente, sin necesidad de conexión a internet, lo que facilita su ejecución en cualquier equipo con Python instalado.

2. Requisitos del Sistema

Para ejecutar el juego, se necesita:

- Python 3.8 o superior.
- Un sistema operativo compatible con entornos gráficos (Windows, macOS o Linux).
- Un editor de texto o entorno de desarrollo (IDLE, Visual Studio Code, PyCharm, etc.) que permita ejecutar archivos Python.
- Acceso a la terminal o consola del sistema operativo.

No se requieren librerías externas, ya que se utilizan solo las librerías estándar de Python:

- `tkinter` para la interfaz gráfica.
- `os` para la gestión de rutas y archivos.
- `random` para la aleatorización de piezas.
- `time` para el control de animaciones y estadísticas.

3. Instrucciones de Instalación y Compilación

La instalación y ejecución son sencillas, ya que no se requiere compilación ni librerías externas.

Pasos para la instalación:

1. Verificar que Python esté instalado (se puede hacer desde la consola o un entorno gráfico).
2. Crear una carpeta para el proyecto y colocar el archivo principal `Proyecto 2.py` dentro.
3. Si se desean conservar estadísticas o partidas guardadas, incluir los archivos `estadisticas.txt` y `partida_XX.txt` (donde `XX` es un número) en la misma carpeta.
4. Acceder a la carpeta del proyecto desde un entorno de desarrollo o la terminal.
5. Ejecutar el archivo del juego.

Una vez iniciado, se mostrará el menú principal, listo para usarse sin configuraciones adicionales.

4. Instrucciones de Ejecución

Al iniciar el programa, se abre una ventana gráfica con el menú principal y las siguientes opciones:

- **Iniciar nuevo juego**: Comienza una nueva partida. Se solicita el nombre del jugador para el registro de estadísticas.
- **Ver estadísticas**: Muestra un resumen de todas las partidas jugadas, incluyendo puntaje máximo, jugador destacado, nivel más alto, tiempo total jugado y líneas eliminadas.
- **Cargar partida**: Permite reanudar partidas guardadas previamente.
- **Salir**: Cierra la aplicación.

Durante el juego, se interactúa con las piezas usando las siguientes teclas:

- **Flecha izquierda**: Mover la pieza a la izquierda.
- **Flecha derecha**: Mover la pieza a la derecha.
- **Flecha hacia abajo**: Acelerar la caída de la pieza.
- **Barra espaciadora**: Rotar la pieza en sentido horario.
- **Letra P**: Pausar o reanudar la partida.

La velocidad de caída de las piezas aumenta a medida que el jugador sube de nivel.

5. Estructura de Archivos

El sistema utiliza archivos de texto para guardar información persistente. Estos archivos son creados y gestionados automáticamente por el programa:

- `estadisticas.txt`: Almacena datos acumulados de todas las partidas, como cantidad de juegos, mejor puntaje, mejor jugador y tiempo total.

- `partida_XX.txt`: Archivos individuales numerados (del 01 al 99) que guardan el estado completo de una partida, incluyendo el tablero, la puntuación, la pieza activa y el nombre del jugador.

No se necesita que el usuario realice configuraciones adicionales; el sistema crea los archivos cuando son necesarios.

6. Consideraciones Adicionales

- Las estadísticas se actualizan automáticamente al finalizar cada partida.
- Se pueden guardar hasta 99 partidas diferentes, cada una con un número secuencial en el nombre del archivo.
- El tablero se puede modificar manualmente en el código fuente, en la sección `MATRIZ_PERSONALIZABLE`, para añadir obstáculos fijos ('X') y aumentar la dificultad.
- El juego tiene múltiples niveles; la velocidad de caída de las piezas aumenta progresivamente con el puntaje, incrementando el desafío.

7. Solución de Problemas

- **Problema: El archivo de estadísticas no se encuentra.**
  - **Solución**: El programa crea automáticamente `estadisticas.txt` al finalizar la primera partida; no es necesario crearlo manualmente.
- **Problema: Error al intentar cargar una partida.**
  - **Solución**: Asegurarse de que el archivo de la partida no haya sido modificado manualmente, ya que errores de formato impiden que el sistema reconstruya la información.
- **Problema: La interfaz no se abre al ejecutar el programa.**
  - **Solución**: Verificar que el sistema operativo permita ventanas gráficas y que Python esté correctamente instalado.

8. Créditos

Este sistema fue desarrollado como parte del curso Taller de Programación del Tecnológico de Costa Rica, durante el primer semestre de 2025. El proyecto integró diversas habilidades aprendidas, tales como:

- Programación estructurada y modular en Python.
- Diseño de interfaces gráficas con `tkinter`.
- Manipulación de estructuras de datos (listas y matrices).
- Gestión de archivos para almacenamiento y recuperación de información.
- Implementación de lógica condicional y ciclos para el control del juego.

Este desarrollo no solo reforzó conocimientos técnicos, sino también la capacidad de análisis, solución de problemas y diseño lógico de sistemas interactivos.

9. Análisis de Resultados

Durante el desarrollo, se implementaron todas las funcionalidades requeridas y algunas adicionales que mejoraron la experiencia de usuario.

Objetivos alcanzados:

- Creación de un tablero funcional de 22 filas por 12 columnas con bordes y obstáculos.
- Implementación de todos los tetrominós (O, I, L, J, T, Z, S, + y U) con sus rotaciones, restricciones y comportamientos.
- Movimiento de piezas (izquierda, derecha, abajo y rotación) con validación de colisiones.
- Eliminación automática de filas completas con asignación de puntaje.
- Registro de estadísticas persistentes (nombre del jugador, nivel, tiempo jugado y puntaje máximo).
- Aumento progresivo de la dificultad mediante el incremento de la velocidad de caída de las piezas.
- Interfaz gráfica funcional y amigable con Tkinter.
- Funcionalidad de guardado y carga de partidas desde archivos de texto.

Objetivos no alcanzados:

- No se implementó la visualización de la siguiente pieza.
- El sistema de ranking solo muestra al "mejor jugador", no un top 10 completo.

Observaciones:

- El programa se comportó de manera estable durante las pruebas.
- Las funciones críticas (guardado/recuperación, rotación, eliminación de líneas) respondieron correctamente.
- La interfaz gráfica fue intuitiva incluso para usuarios sin experiencia técnica.

10. Diseño del Programa

El diseño del programa se estructuró en módulos y funciones para facilitar la comprensión, mantenimiento y futuras ampliaciones. Se dividió en las siguientes secciones clave:

- **Funciones auxiliares**: Desarrollo de funciones personalizadas para medir listas, verificar pertenencia y manejar matrices, evitando el uso de funciones integradas no permitidas.
- **Interfaz gráfica**: Uso de Tkinter para diseñar las ventanas del menú principal, tablero de juego, selector de partidas guardadas y módulo de estadísticas, organizadas con botones, etiquetas y marcos.
- **Lógica del juego**: Contiene las reglas para el movimiento y rotación de piezas, validación de colisiones, inserción en el tablero, eliminación de líneas y verificación de condiciones de finalización.

- **Persistencia de datos**: Las estadísticas globales se guardan en `estadisticas.txt`, y cada partida individual en archivos numerados (`partida_01.txt`, etc.), lo que permite una recuperación ordenada.
- **Estilo de codificación**: Se mantuvo un estilo claro y coherente, con comentarios explicativos, nombres de variables descriptivos y separación entre la lógica y la presentación.

Este diseño modular facilitó la programación y permitió agregar nuevas funciones, como obstáculos o el manejo de pausas, sin afectar las existentes.

11. Conclusión

Este proyecto representó un desafío técnico y personal que permitió consolidar conocimientos clave en el desarrollo de software con Python. Al ser un juego interactivo con múltiples componentes visuales y lógicos, requirió una planificación metódica, implementación progresiva y constante validación del sistema.

Además de fortalecer habilidades técnicas, este trabajo fomentó el pensamiento lógico, la resolución de problemas, la atención al detalle y la paciencia ante errores. La experiencia adquirida es aplicable a futuros proyectos más complejos y a situaciones reales de desarrollo de software.

En resumen, el proyecto cumplió satisfactoriamente con los objetivos establecidos, entregando una aplicación funcional, sólida y completa que demuestra la capacidad del estudiante para diseñar, construir y documentar un sistema interactivo real.

Link al video de YouTube:

https://www.youtube.com/watch?v=KTfOnP1SeVs