Instituto Tecnológico de Costa Rica

Centro académico de Limón

Escuela de ingeniería en computación

Curso: Taller de programación

Proyecto 2

Semestre 1

Profesor: Cristian Campos

Estudiante: Jahrell Gourzong Ortiz

Año: 2025

Para comenzar a jugar, primero necesitas tener Python instalado en tu computadora. El juego está diseñado para funcionar con Python 3.6 o versiones más recientes. No se requieren instalaciones adicionales de bibliotecas, ya que utiliza componentes estándar incluidos con Python.

Al ejecutar el programa, verás aparecer la pantalla principal con varias opciones. Para comenzar una nueva partida, simplemente haz clic en el botón "Jugar". Si deseas continuar una partida guardada anteriormente, selecciona "Cargar Partida".

Controles Durante el Juego

Mientras juegas, utiliza estas teclas para controlar las piezas:

- Tecla de flecha izquierda: Mueve la pieza hacia la izquierda
- Tecla de flecha derecha: Mueve la pieza hacia la derecha
- Tecla de flecha abajo: Acelera la caída de la pieza
- Tecla de flecha arriba: Rota la pieza 90 grados
- Barra espaciadora: Pausa o reanuda el juego

Guardando tu Progreso

Puedes guardar tu partida en cualquier momento haciendo clic en el botón "Guardar" ubicado en la parte derecha de la pantalla. El juego te pedirá que ingreses un nombre para tu archivo de guardado. Para continuar esta partida más tarde, simplemente selecciona "Cargar Partida" desde el menú principal.

Finalizando el Juego

El juego terminará automáticamente cuando las piezas alcancen la parte superior del tablero. También puedes salir en cualquier momento usando el botón "Salir" para volver al menú principal. Tus mejores puntuaciones se guardarán automáticamente en el ranking del juego.

Cómo jugar

El juego se controla completamente con el teclado. Usa las flechas izquierda y

derecha para mover la pieza actual horizontalmente. La flecha hacia abajo acelera su caída, mientras que la flecha arriba la rota 90 grados. Presiona la barra espacial para pausar el juego cuando lo necesites.

Tipos de piezas

Existen siete piezas diferentes, cada una con su propia letra y color de identificación: la I (cian), O (amarillo), T (morado), L (naranja), J (azul), S (verde) y Z (rojo). Todas pueden rotarse excepto la O (el cuadrado perfecto).

Puntuación

Cada línea completa que logres eliminar te dará 100 puntos. El juego lleva registro de tu mejor puntuación en el ranking, donde podrás competir por estar entre los 10 mejores jugadores.

Funciones especiales

Puedes guardar tu partida en cualquier momento desde el menú del juego, lo que te permitirá continuar después exactamente donde la dejaste. También puedes consultar el ranking para ver cómo te comparas con otros jugadores.

Estrategias básicas

Intenta dejar el menor número de huecos posibles. Es mejor completar varias líneas a la vez que una sola. Deja espacio en los laterales para las piezas más largas como la I. No te acostumbres a rotar siempre en el mismo sentido, practica ambas direcciones.

Cuando pierdas

Al terminar la partida, verás un resumen con tu puntuación final y cuántas líneas completaste.

Descripción del Problema

El proyecto consiste en desarrollar una versión funcional del clásico juego Tetris utilizando el lenguaje Python junto con la biblioteca Tkinter para la interfaz gráfica. El desafío principal radica en modelar fielmente las mecánicas del juego original mientras se cumplen los siguientes requisitos técnicos:

Aspectos Clave del Problema:

1. **Modelado del Tablero:**
    - Implementar una matriz de 22x12 que represente el área de juego
    - Gestionar bordes fijos (representados por '+') y celdas vacías (0)
    - Incluir obstáculos centrales como parte del diseño inicial
2. **Mecánica de Piezas:**
    - Programar los 7 tetrominós clásicos con sus respectivas rotaciones
    - Implementar movimiento fluido con restricciones de colisión
    - Garantizar generación aleatoria equilibrada de piezas
3. **Lógica del Juego:**
    - Detección y eliminación de líneas completas
    - Cálculo y actualización de puntuación (100 puntos por línea)
    - Condiciones precisas de game over
4. **Persistencia de Datos:**
    - Sistema de guardado/recuperación de partidas
    - Mantenimiento de ranking de puntuaciones (Top 10)
    - Almacenamiento en archivos de texto con formato estructurado

Diseño del Programa

El programa se estructura en tres componentes principales que trabajan en conjunto para recrear la experiencia clásica del Tetris:

Núcleo del Juego

La lógica central opera sobre una matriz de 22x12 que representa el tablero. Esta matriz utiliza símbolos específicos: '+' para bordes, '0' para espacios vacíos, y letras mayúsculas (I, O, L, etc.) para las piezas activas. El sistema de colisiones verifica

constantemente los límites del tablero y la superposición con piezas colocadas, utilizando algoritmos de comparación matricial que analizan posiciones futuras antes de permitir movimientos.

Mecánicas de Piezas

Cada tetrominó se implementa como una submatriz rotable, donde la rotación se resuelve mediante transposición de matrices combinada con reversión de filas. El generador de piezas emplea un sistema pseudoaleatorio que garantiza distribución equilibrada, evitando sequencias prolongadas de una misma figura. El movimiento descendente automático se controla con un temporizador configurable que acelera progresivamente según la puntuación.

Interfaz y Datos

La capa gráfica construida con Tkinter sincroniza dos representaciones paralelas: el modelo matricial abstracto y su visualización en canvas con colores distintivos. El sistema de persistencia guarda estados de juego en archivos CSV legibles, almacenando no solo la matriz actual sino también metadatos como puntuación y tipo de pieza activa. El ranking implementa un mecanismo de ordenamiento que actualiza automáticamente el top 10, con validación para prevenir duplicados o registros corruptos.

La arquitectura emplea principios de encapsulamiento, donde cada componente expone interfaces claras: el módulo de juego procesa interacciones, el renderizador traduce estados a gráficos, y el gestor de datos maneja la persistencia sin afectar la lógica principal. Esta separación permite modificar aspectos visuales o de almacenamiento sin impactar el núcleo del juego.

Librerías Utilizadas

El proyecto aprovecha varias librerías estándar de Python para implementar sus diferentes funcionalidades:

Tkinter sirve como el cimiento de la interfaz gráfica, proporcionando los elementos visuales básicos como ventanas, botones y el canvas donde se dibuja el tablero de

juego. Esta librería nativa de Python permite crear la representación visual de la matriz del juego y capturar los eventos del teclado para controlar las piezas.

Para la generación aleatoria de las piezas que caen, se emplea **random**, específicamente su función choice() que selecciona al azar entre los diferentes tipos de tetrominós disponibles. Esto garantiza una distribución impredecible pero equilibrada de las piezas durante la partida.

La librería **os** se utiliza en el manejo de archivos para las funcionalidades de guardado y carga de partidas. Permite verificar la existencia de archivos previos, crear nuevos registros y leer los datos almacenados, haciendo posible continuar juegos interrumpidos y mantener un historial de puntuaciones.

Para los cuadros de diálogo que permiten guardar partidas o mostrar mensajes al usuario, se aprovechan componentes de **filedialog** y **messagebox**, ambos módulos incluidos en Tkinter. Estos proporcionan interfaces estándar para interacciones comunes como seleccionar nombres de archivo o mostrar alertas.

El proyecto utiliza **simpledialog** para solicitar información básica al usuario, como su nombre al comenzar una nueva partida. Esta funcionalidad está integrada en la interfaz gráfica principal y mantiene la coherencia visual con el resto de la aplicación.

Análisis de Resultados

El desarrollo del juego Tetris en Python cumplió satisfactoriamente con los objetivos principales planteados inicialmente. La implementación logró recrear las mecánicas fundamentales del Tetris incluyendo el movimiento y rotación de piezas, la detección de colisiones y la eliminación de líneas completas. El sistema de puntuación funciona correctamente, asignando 100 puntos por cada línea eliminada y actualizando en tiempo real el marcador visible para el jugador.

En cuanto a la interfaz gráfica, se consiguió una representación visual clara del tablero y las piezas, utilizando colores distintivos para cada tipo de tetrominó. Los controles responden adecuadamente a las teclas direccionales, permitiendo una experiencia de juego fluida. La función de pausa mediante la barra espaciadora

opera como se esperaba, deteniendo temporalmente la caída de las piezas sin afectar el estado del juego.

El sistema de guardado y carga de partidas demuestra robustez, preservando correctamente todos los elementos del estado del juego, incluyendo la disposición del tablero, la pieza actual y las estadísticas. El ranking de puntuaciones mantiene un registro preciso de los mejores resultados, ordenándolos automáticamente y limitando la lista a las diez mejores marcas.

Como aspectos mejorables, se identificó que la velocidad de caída automática podría ajustarse de manera más gradual para aumentar el desafío a medida que avanza la partida. También se observó que la rotación de piezas cerca de los bordes podría optimizarse para evitar situaciones donde movimientos técnicamente válidos son rechazados por el sistema de colisiones.

La implementación de obstáculos centrales, un requisito específico del proyecto, añadió complejidad inesperada al desarrollo, pero finalmente se integró satisfactoriamente, modificando el algoritmo de colisiones para reconocer estas estructuras como elementos inamovibles del tablero.

Conclusión

El desarrollo de este clon de Tetris en Python demostró ser un proyecto desafiante pero enriquecedor que permitió aplicar diversos conceptos de programación en un contexto práctico. La implementación logró capturar la esencia del juego clásico, desde sus mecánicas fundamentales hasta la experiencia de juego fluida.

A través del proceso de desarrollo, se pudo apreciar la importancia de un diseño bien estructurado, donde la separación entre la lógica del juego, la interfaz gráfica y el manejo de datos resultó crucial para mantener un código organizado y mantenible. El uso de Tkinter demostró ser una elección adecuada para la interfaz, ofreciendo un equilibrio entre simplicidad y funcionalidad suficiente para este tipo de aplicación.