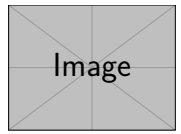
Diseño y Lógica del Juego Tetris

Lógica y Algoritmos - Universidad Autónoma de Zacatecas

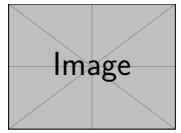
Carlos Alberto Bello Rosas

Ingeniería en software

14 de diciembre de 2025



Contenido



Introducción al Proyecto

Objetivo del Proyecto

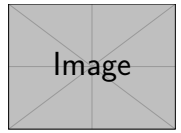
Diseñar e implementar el clásico juego Tetris aplicando conceptos de lógica, algoritmos y estructuras de datos.

Componentes principales:

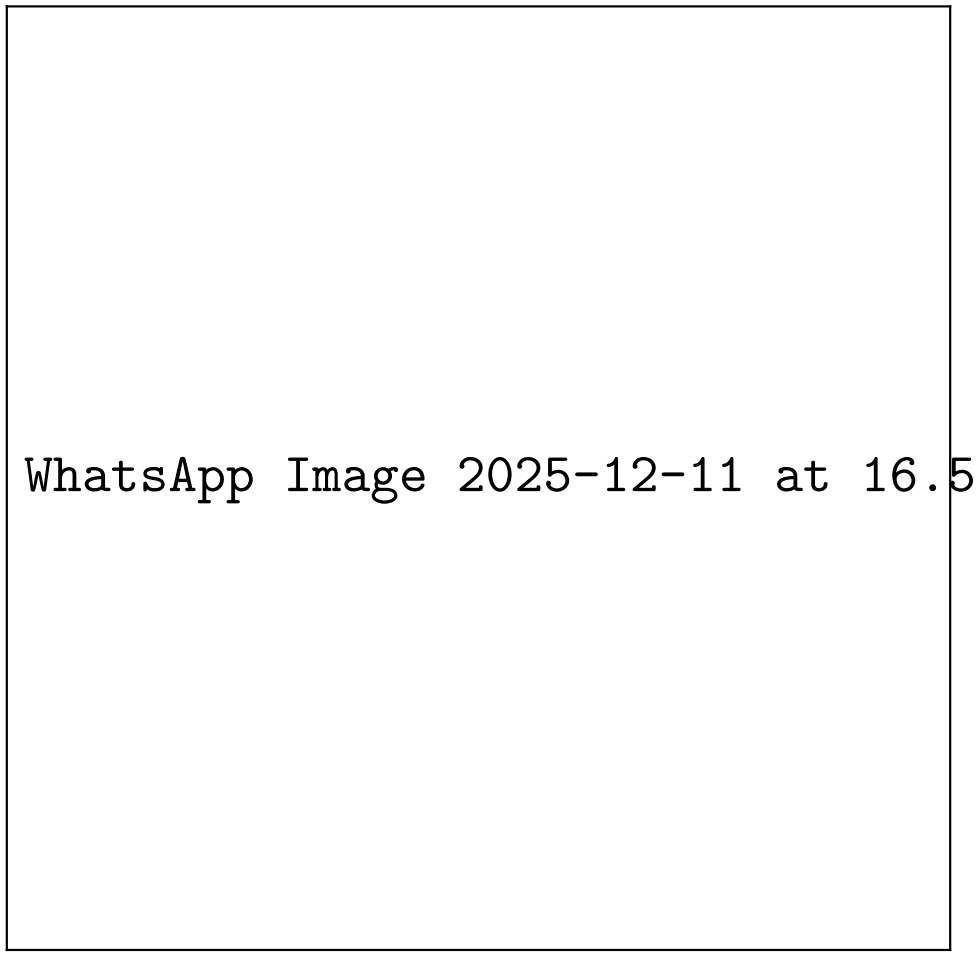
- Tablero de juego
- Sistema de piezas
- Motor de colisiones
- Sistema de puntuación

Conceptos aplicados:

- Matrices bidimensionales
- Lógica booleana
- Algoritmos de detección
- Programación estructurada



Tablero de Juego

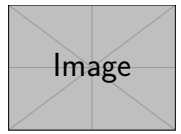


Estructura del Tablero

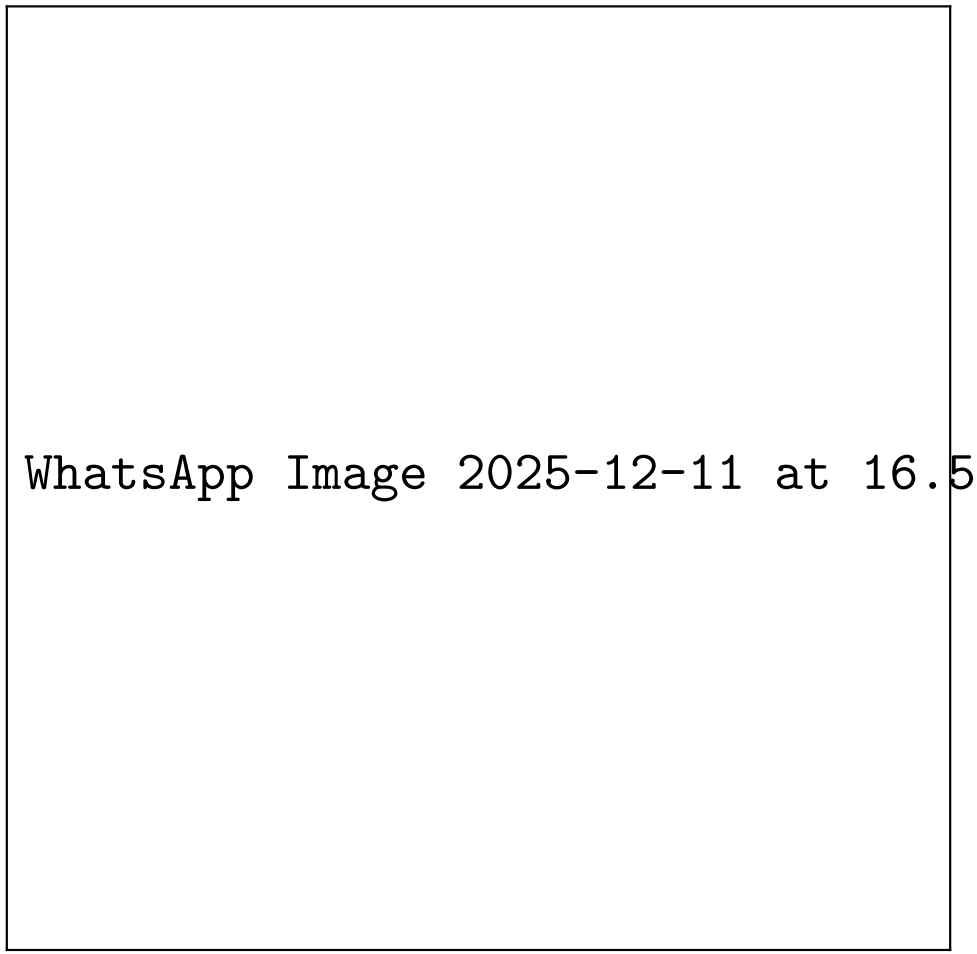
- Matriz de 10 columnas $\times$ 20 filas
- Cada celda: vacía (0) u ocupada (1)
- Coordenadas (x, y)

Las 7 Piezas Básicas

WhatsApp Image 2025-12-11 at 16.58.52(1).jpeg



Sistema de Puntuación

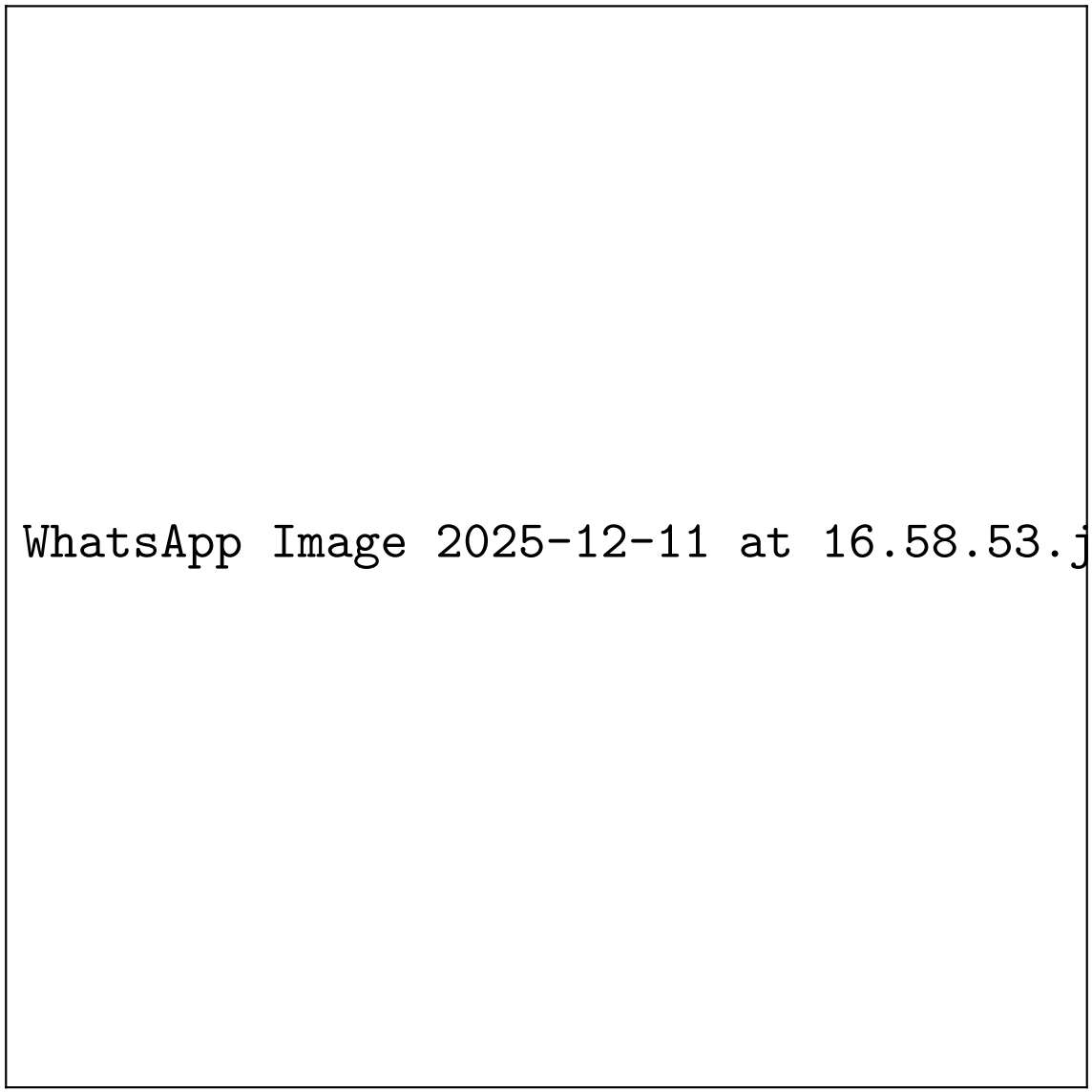


Fórmula de Puntuación

Puntos = **Base** $\times$ (**Nivel** + 1)

Líneas	Base
1 línea	100

Flujo del Juego



Estados del Juego

Lógica de Colisiones

WhatsApp Image 2025-12-11 at 16.58.52(1).jpeg

Verificaciones:

1. ¿Toca borde izquierdo?
2. ¿Toca borde derecho?
3. ¿Toca el fondo?
4. ¿Choca con otra pieza?

Función de Colisión

Algoritmo de Eliminación

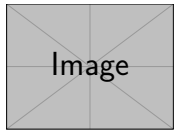
Proceso de Eliminación

Para cada fila (de abajo hacia arriba):

1. Verificar si todos los bloques están llenos
2. Si SÍ:
   - Eliminar fila
   - Mover filas superiores
   - Sumar puntos
3. Si NO: Continuar

Código de Verificación

`bool lineaCompleta(int fila)`
Verifica si todos los bloques en una fila están ocupados



Algoritmo de Rotación

Rotación Matricial

Para rotar una matriz 90° en sentido horario:

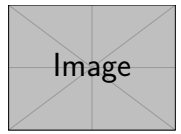
- NuevaMatriz[j][n-i-1] = Matriz[i][j]
- Donde n es el tamaño de la matriz

Ejemplo - Pieza T:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Wall Kick:

- Ajuste de posición si la rotación causa colisión
- Intentar mover 1-2 posiciones
- Garantizar jugabilidad



Sistema de Velocidad

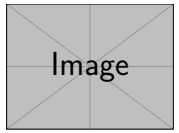
Control del Tiempo

- La pieza cae automáticamente cada cierto tiempo
- Tiempo entre caídas disminuye con el nivel
- Velocidad base: 1000 ms
- Reducción por nivel: 50 ms
- Velocidad mínima: 100 ms

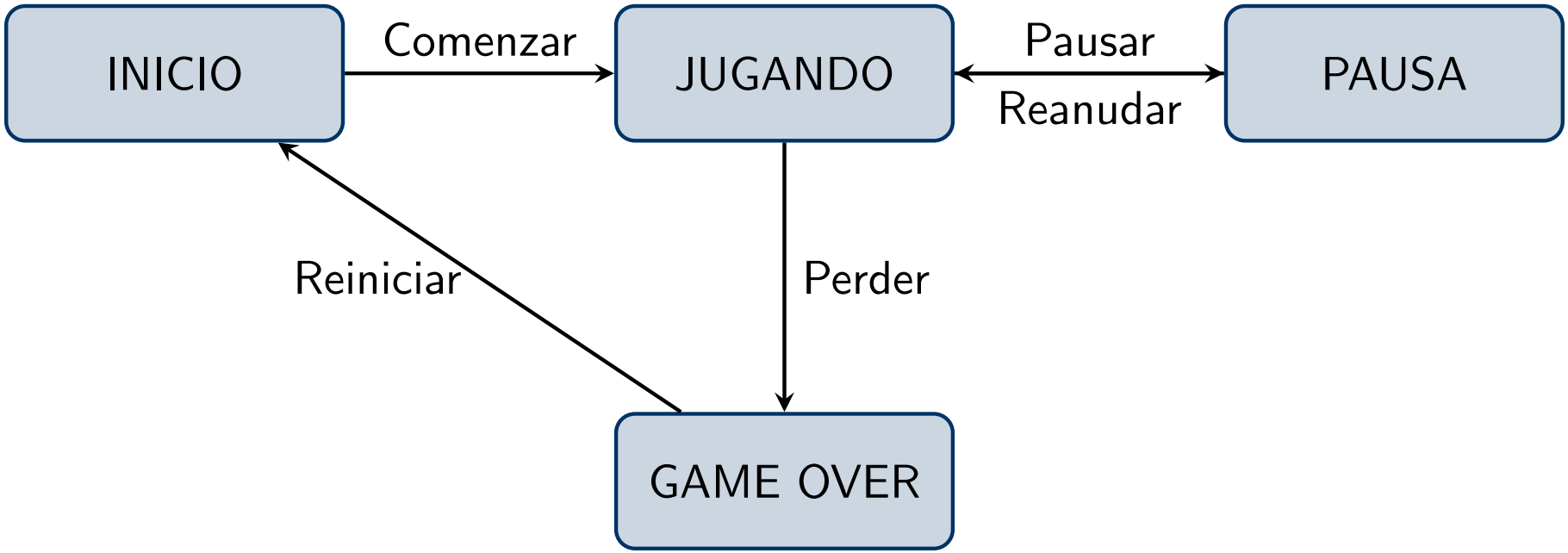
Fórmula de Velocidad

`velocidad = 1000 - (nivel × 50)`

Si `velocidad < 100` entonces `velocidad = 100`



Máquina de Estados



Variables de control:

- juegoActivo
- juegoPausado
- juegoTerminado

Transiciones:

- Inicio → Jugando
- Jugando Pausa
- Jugando → Game Over
- Game Over → Inicio

Image

Desafíos y Soluciones

Desafío 1: Rotaciones

Problema: Piezas rotadas fuera del tablero **Solución**: Sistema de wall kick

Desafío 2: Aleatoriedad

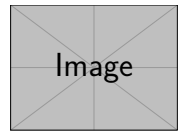
Problema: Secuencias imposibles
Solución: Sistema de bolsa de 7 piezas

Desafío 3: Rendimiento

Problema: Actualización ineficiente
Solución: Doble búfer, FPS fijo

Desafío 4: Tiempo

Problema: Velocidad variable **Solución**: Temporizador independiente



Conclusiones y Aprendizajes

Conceptos Aplicados

- **Matrices**: Tablero y piezas
- **Lógica booleana**: Colisiones y condiciones
- **Algoritmos**: Eliminación de líneas, rotación
- **Máquinas de estado**: Control del juego

Habilidades Desarrolladas

- Diseño de algoritmos
- Resolución de problemas
- Organización de código
- Optimización de rendimiento

Aplicación Práctica

Este proyecto demostró la aplicación de conceptos teóricos en un sistema interactivo y complejo.