UNIDAD DE INGENIERÍA EN COMPUTACIÓN
Centro Académico de Limón
Análisis de Algoritmos
Profesor: Kenneth Obando Rodríguez

TEC | Tecnológico de Costa Rica

Proyecto 2 (15%) Fecha de Entrega: diciembre, 2025

GA-Arcade

1 Descripción

Implemente, en el navegador (HTML, CSS y JavaScript **sin librerías externas**), un **juego clásico 2D** y un **agente** controlado por un Algoritmo Genético (AG) que aprenda una política para jugarlo. Puede elegir **uno** de los siguientes títulos:

- **Pac-Man** (laberinto, pellets, fantasmas).
- **Tetris** (tetrominós, tablero 10×20).
- **Arkanoid/Breakout** (barra, bola, ladrillos).
- **Space Invaders** (nave, alienígenas, proyectiles).

El proyecto consta de dos partes inseparables: (1) un motor mínimo del juego (lógica + render con `<canvas>`) y (2) un AG que **evoluciona** los parámetros de una **política** de juego. No se permiten frameworks ni motores de juego (*p5.js*, *three.js*, *Phaser*, etc.).

2 Alcance y restricciones

2.1 Motor del juego

- Render con `Canvas` 2D. FPS objetivo: 30–60.
- Bucle de juego con `requestAnimationFrame`; **no** usar librerías.
- Física simple suficiente (colisiones AABB o por píxel según corresponda).
- Entradas del agente: estado *suficiente y legal* (p. ej., grilla de *Tetris*, distancias relativas en *Pac-Man*, etc.).

2.2 Agente y codificación (genotipo → política)

Elija **una** representación y documente sus genes:

1. **Reglas heurísticas ponderadas**: vector de pesos $\mathbf{w}$ que evalúa estados/acciones.

2. **Tabla de decisión finita** (*lookup*): discretiza observaciones y mapea a acciones.
3. **FSM** (máquina de estados finitos): genes definen transiciones por condiciones simples.

No usar redes neuronales ni librerías de ML.

3 Algoritmo Genético (mínimos)

- Población $N \geq 20$; generaciones $G \geq 50$ (parametrizables).
- Selección: torneo/ranking/ruleta (documentar).
- Cruzamiento: 1–2 puntos o uniforme (según codificación).
- Mutación: gaussiana en pesos o % de bits/entradas cambiadas; tasa parametrizable.
- Reemplazo con **elitismo** (al menos 1 individuo).
- **Semilla** aleatoria configurable para reproducibilidad.

4 Función de *fitness*

Cada equipo debe **diseñar e implementar** una estrategia de *fitness* **propia**, pertinente al juego elegido y **justificada** con base en referencias de *Reinforcement Learning* por políticas (policy-based). El *fitness* debe reflejar fielmente el objetivo del juego, evitar sesgos obvios y ser robusto a pequeñas variaciones de nivel/semilla.

4.1 Implementación y validación

- **Reproducibilidad:** exponga seed, γ, m, horizonte T y escalas de recompensa.
- **Robustez:** evalúe con ≥3 semillas y (si aplica) 2 variantes de nivel; reporte media y desviación.
- **Ablación:** muestre impacto de quitar/añadir un término (p. ej., penalización de huecos en Tetris).
- **Coste:** estime evaluaciones × pasos por episodio y su efecto en G y N.

4.2 Restricciones

- El *fitness* solo puede usar información observable por un jugador legal del juego.
- Prohibidos *shortcuts* que inspeccionen el “estado interno” para otorgar puntos que no surgen de la dinámica (p. ej., leer #ladrillos desde estructuras internas sin colisión).
- No se permiten redes neuronales ni librerías de ML; la política debe ser compatible con la codificación elegida (pesos, tabla, FSM).

5 Interfaz y ejecución

Al cargar la página:

- Parámetros: N, G, % de **selección**, **cruzamiento** y **mutación** (**deben sumar 100%** entre las tres categorías operativas), tamaño de torneo, semilla, FPS de simulación, episodios por individuo.
- Controles: `Start`, `Pause/Resume`, `Reset`, `Export best.json`, `Demo best`.
- Métricas en vivo: mejor y promedio de *fitness* por generación (gráfico lineal), tiempo total y tiempo promedio por generación.
- Vista del **mejor individuo** jugando (modo *demo* a velocidad normal).

Valide la suma de porcentajes; si no cumplen, deshabilite `Start` y muestre advertencia.

6 Diseño experimental y análisis

- **Baselines**: compare contra un agente *greedy* sencillo o heurística fija del juego.
- **Ablación**: altere un componente (p. ej., sin elitismo o distinta tasa de mutación) y discuta el efecto.
- **Repeticiones**: al menos 3 corridas por configuración (semillas distintas); reporte media y desviación.
- **Complejidad**: estime coste por generación (render/evaluación) e identifique el cuello de botella.

7 Artículo científico (obligatorio)

Cada equipo debe entregar un **artículo** con los resultados experimentales. Se evaluará *dentro de* C5 (análisis experimental) y C6 (estilo y limpieza) de la rúbrica.

7.1 Formato y extensión

- Extensión: **4–6 páginas** (sin contar referencias).
- Formato: PDF, tipografía legible, márgenes estándar; citas con `biblatex` estilo `ieee`.
- Estructura recomendada (IMRyD):
  1. **Introducción:** problema, juego elegido, contribución.
  2. **Metodología:** codificación del agente (genes), operadores genéticos, *fitness* propuesto (fórmulas y justificación), protocolo experimental (semillas, episodios, hiperparámetros).
  3. **Resultados:** tablas y gráficas (mejor/promedio por generación; comparación con baseline; ablaciones). Reporte media $\pm$ desviación y discusión de significancia práctica.
  4. **Discusión:** limitaciones, amenazas a la validez, sensibilidad a hiperparámetros.
  5. **Conclusiones y trabajo futuro.**

7.2 Requisitos técnicos del artículo

- Incluir **pseudocódigo de evaluación** del *fitness* por individuo y del bucle del AG.
- Describir **complejidad temporal** por generación y el cuello de botella identificado.
- **Reproducibilidad:** listar semillas, versiones de navegador/SO, y adjuntar en el `.zip` los archivos de configuración y logs.
- Citar fuentes (p. ej., **sutton2018**, **eiben2003**) y añadir cualquier referencia usada en `referencias.bib`.

Checklist mínimo (incluya una tabla en el artículo):

- Juego elegido y reglas implementadas.
- Codificación de la política (genes) y operadores (selección, cruce, mutación, reemplazo).
- Definición formal de recompensas y *fitness*.
- Protocolo experimental (semillas, N, G, episodios, γ, tasas).
- Baseline(s) y ablaciones.
- Gráficas/tablas con estadísticas (media, desviación).

8 Evaluación

- **C1. AG funcional y correcto (35%)**: población, selección, cruzamiento, mutación, reemplazo, elitismo, semillas.
- **C2. Motor de juego (20%)**: colisiones, bucle estable, estado consistente para el agente.
- **C3. Fitness propio y resultados (20%)**: diseño original y bien justificado (policy-based), consistencia con objetivos, mejora sostenida, semillas múltiples y ablación.
- **C4. Interfaz y UX (10%)**: parámetros, validaciones, gráfico en vivo, exportación y *demo*.
- **C5. Análisis experimental (10%)**: **incluye la calidad del artículo** (claridad de metodología y resultados, rigor estadístico, discusión).
- **C6. Estilo y limpieza (5%)**: redacción del artículo y legibilidad del código/README.

9 Entrega

Suba a **TecDigital** antes de las **10:00 p. m. del 21 de noviembre de 2025** un archivo `.zip` con:

- Código fuente (HTML/CSS/JS) y README con instrucciones (navegador probado).
- `config/` con parámetros usados (`.json`) y `logs/` con históricos de *fitness*.
- `best.json` (genotipo/política del mejor individuo) y `replay/` opcional.
- **`paper.pdf`** (4–6 páginas) siguiendo la Sección *Artículo científico*.
- `referencias.bib` consistente con `biblatex`.

Ejecución local. Use un servidor estático simple (`python -m http.server`) para evitar restricciones de CORS.

10 Aclaraciones y correcciones

- **31 de octubre de 2025**: Publicación inicial de la especificación.
- **Actualización**: Se hace **obligatorio** el *artículo científico* y se detalla su formato.

11 Referencias sugeridas