Plan de Proyecto y Timeline: Juego de Empresas (Python)

1. Introducción

Este documento detalla el plan de desarrollo y el cronograma estimado para el proyecto "Juego de Empresas", un simulador basado en Python que modela la competencia entre cuatro empresas utilizando cadenas de Markov, según la descripción proporcionada en el documento Juego de Empresas.pdf. El objetivo es crear un simulador funcional, modular y bien documentado.

2. Contexto Inicial

- **Fecha de Inicio Estimada:** 18 de Abril de 2025
- **Estado Inicial:** Se parte de una estructura de proyecto modular básica (MVP - Mínimo Producto Viable) que incluye las clases Company y Simulation, archivos de configuración y un bucle de simulación principal. La lógica de actualización de la matriz de Markov y la toma de decisiones de las empresas está simplificada o es estática en este punto inicial.
- **Objetivo Final:** Simulador completo que implementa la dinámica de la matriz de Markov basada en decisiones (Precio, Marketing, Tecnología), gestión de costes completa (fijos, variables, almacenamiento, ruptura), demanda variable y capacidad de guardar/cargar partidas.
- **Referencia Principal:** Juego de Empresas.pdf

3. Fases del Proyecto

El desarrollo se divide en las siguientes fases principales:

1. **Implementación de la Dinámica Central:** Enfocada en la actualización de la Matriz de Markov según las fórmulas del PDF.
2. **Interacción del Usuario y Gestión de Datos:** Permitir la configuración, toma de decisiones interactivas y persistencia (guardar/cargar).
3. **Refinamiento de Mecánicas y Reportes:** Añadir detalles como demanda variable, costes completos y mejor visualización de resultados.
4. **Pruebas, Documentación y Entrega:** Asegurar la calidad, robustez y usabilidad del simulador.

4. Timeline Detallado

Fase 1: Implementación de la Dinámica Central (Actualización Matriz de Markov)

- **Objetivo:** Que la simulación refleje el impacto de las decisiones (Precio, MKT, Tech) en la matriz de Markov y la cuota de mercado, siguiendo las fórmulas del

PDF.

- **Duración Estimada:** 2 - 3 Semanas
- **Periodo Estimado:** 18 de Abril – 9 de Mayo de 2025

Semana	Fechas Estimadas	Hitos / Tareas Clave	Estado
1	18 Abr - 25 Abr	**Efecto Precio:** - Tarea 1.1: Implementar función efecto PVP en M (fórmulas \delta_p, m_{ii}, redistribución, normalización) [source: 46, 47, 48, 50]. - Tarea 1.2: Integrar en simulation.py (probar con cambios simulados). **Meta:** M responde a cambios de precio.	Pendiente
2	28 Abr - 2 May	**Efecto Marketing y Tecnología (Ventas):** - Tarea 1.3: Implementar función efecto Marketing en M (fórmula logarítmica, redistribución, normalización) [source: 54, 55, 64]. - Tarea 1.4: Implementar función efecto Tecnología (Ventas) en M (fórmula hiperbólica/lineal, redistribución, normalización) [source: 81, 82]. **Meta:** M responde a MKT y Tech.	Pendiente
3	5 May - 9 May	**Efecto Tecnología (Costes) e**	Pendiente

		Integración: - Tarea 1.5: Implementar función efecto Tecnología en Costes Variables (CV) (fórmula logarítmica \Delta c_n) [source: 71, 73]. - Tarea 1.6: Integrar todos los efectos (Precio, MKT, Tech) en _update_markov_matrix. **Meta:** Núcleo matemático implementado. M y CV responden a las 3 decisiones.	

Fase 2: Interacción del Usuario y Gestión de Datos

- **Objetivo:** Permitir al usuario configurar, jugar interactivamente y guardar/cargar partidas.
- **Duración Estimada:** 2 Semanas
- **Periodo Estimado:** 12 de Mayo – 23 de Mayo de 2025

Semana	Fechas Estimadas	Hitos / Tareas Clave	Estado
4	12 May - 16 May	**Configuración y Decisiones Interactivas:** - Tarea 2.1: (Opcional) Cargar configuración inicial desde archivo (JSON/YAML). - Tarea 2.2: Modificar bucle principal para solicitar decisiones al usuario (PVP, MKT, Tech, Producción) usando input_utils. - Tarea 2.3: Pasar decisiones a la lógica de actualización de estado (Fase 1). **Meta:** Juego	Pendiente

		interactivo.	
5	19 May - 23 May	**Persistencia (Guardar/Cargar):** - Tarea 2.4: Implementar save_game para guardar estado completo (considerar pickle o json) [source: 2]. - Tarea 2.5: Implementar load_game para restaurar estado completo [source: 2]. - Tarea 2.6: Integrar opciones Guardar/Cargar en main.py. **Meta:** Progreso guardable y reanudable.	Pendiente

Fase 3: Refinamiento de Mecánicas y Reportes

- **Objetivo:** Incorporar demanda variable, costes completos y mejorar la visualización de resultados.
- **Duración Estimada:** 2 Semanas
- **Periodo Estimado:** 26 de Mayo – 6 de Junio de 2025

Semana	**Fechas Estimadas**	**Hitos / Tareas Clave**	**Estado**
6	26 May - 30 May	**Demanda Variable y Costes Completos:** - Tarea 3.1: Implementar fluctuación aleatoria de demanda en _update_market_demand [source: 120, 153]. - Tarea 3.2: Implementar cálculo completo de costes CALM, CRUPT, CNOSERV según ruptadm [source:	Pendiente

		105-108]. - Tarea 3.3: Implementar lógica de VENTASPENDIENTES [source: 86, 89]. **Meta:** Simulación con demanda variable y costes completos.	
7	2 Jun - 6 Jun	**Mejoras en Reportes y Visualización:** - Tarea 3.4: Mostrar resultados acumulados (beneficios, ventas) [source: 6]. - Tarea 3.5: Añadir gráficos de evolución temporal (presupuesto, stock, cuota) con matplotlib. - Tarea 3.6: (Opcional) Formatear salida similar a tablas del PDF [source: 141-149, 164-172]. **Meta:** Mejor feedback analítico y visual.	Pendiente

Fase 4: Pruebas, Documentación y Entrega

- **Objetivo:** Asegurar la calidad, robustez y usabilidad del simulador.
- **Duración Estimada:** 1 - 2 Semanas
- **Periodo Estimado:** 9 de Junio – 20 de Junio de 2025

Semana	Fechas Estimadas	Hitos / Tareas Clave	Estado
8	9 Jun - 13 Jun	**Pruebas y Validación:** - Tarea 4.1: Pruebas manuales con diferentes escenarios. - Tarea	Pendiente

		4.2: (Recomendado) Pruebas unitarias para funciones críticas (unittest/pytest). - Tarea 4.3: Validar cálculos contra el PDF, intentar replicar ejemplo (aprox.). **Meta:** Código probado y funcionalmente validado.	
9	16 Jun - 20 Jun	**Documentación y Limpieza:** - Tarea 4.4: Revisar y completar docstrings ("""Docstring"""). - Tarea 4.5: Escribir/Actualizar README.md (descripción, instalación, uso). - Tarea 4.6: Limpiar código (eliminar debug prints), asegurar estilo consistente. **Meta:** Proyecto completo, probado, documentado y listo.	Pendiente

5. Consideraciones Adicionales

- **Flexibilidad:** Este cronograma es una guía. La duración real puede variar. Se priorizará la funcionalidad central sobre los refinamientos si surgen retrasos.
- **Solapamiento:** Algunas fases, especialmente las pruebas y la documentación, pueden y deben realizarse de forma iterativa a lo largo del desarrollo, no solo al final.
- **Recursos:** Se asume un esfuerzo de desarrollo individual o de un equipo pequeño con dedicación razonable.
- **Herramientas:** Python, NumPy, Matplotlib (opcional), Git (recomendado para control de versiones).

Este plan proporciona una hoja de ruta estructurada para el desarrollo del simulador "Juego de Empresas".