PROYECTO FINAL
PROGRAMACIÓN ORIENTADA A OBJETOS 2025-2

Este proyecto es una invitación a practicar los cuatro pilares de la programación orientada a objetos: abstracción, encapsulamiento, herencia y polimorfismo. El diseño debe evidenciar distintos tipos de relaciones, así como la creación, instanciación y destrucción de objetos. El propósito principal de este proyecto no es obtener un programa perfecto, sino evidenciar el razonamiento, la reflexión y la evolución cognitiva en el diseño e implementación de una solución orientada a objetos.

El proyecto sitúa al estudiante en el universo ficticio de **Lyrenhold**, donde los gremios de aventureros compiten en el Gran Torneo de la Arena. A partir de este contexto, cada equipo deberá diseñar y programar un sistema que gestione héroes, oponentes y objetos mágicos, simulando combates por turnos con resultados variables.

El enunciado detalla los **elementos técnicos y conceptuales** del proyecto: definición de héroes y roles, gestión del inventario, combate y registro de datos, así como las **etapas del desarrollo**, los **entregables**, el **uso permitido de inteligencia artificial**, y la **rúbrica de evaluación**.

Enunciado del proyecto

*En la ciudad de **Lyrenhold**, cuna de la magia y el acero, cada año se celebra el **Gran Torneo de la Arena**, donde los gremios de aventureros (guilds) compiten por honor, conocimiento y gloria.Estos gremios reúnen a individuos de distintas disciplinas, magos, guerreros, sanadores y exploradores, cada uno con habilidades y estilos únicos. Tu equipo ha sido nombrado **Maestro de una nueva guild** y su misión es crear un sistema que permita gestionar sus héroes y enfrentar rivales.*

Sobre los héroes y los oponentes

Los héroes son los miembros que conforman la guild del jugador. Representan las entidades principales del sistema y poseen un nombre, nivel, vida, ataque, defensa y rol. Cada héroe encarna un tipo de personaje con un estilo de combate distinto, lo que determina su forma de actuar dentro de la arena. A continuación se describen los tres roles principales recomendados para el diseño aunque los nombres y efectos pueden adaptarse libremente:

Guerrero

Representa la fuerza física y la resistencia. Su principal característica es infligir daño directo con ataques estables. En combate, los guerreros pueden tener una probabilidad de realizar **golpes críticos**, donde su daño se multiplica. Tienen mayor **vida** y **defensa**.

Mago

Domina el poder de la energía arcana. Puede infligir **daño elevado**, aunque su defensa es baja. Los ataques mágicos suelen **ignorar parcialmente la defensa del enemigo** o tener un **efecto variable** (por ejemplo, daño adicional aleatorio o posibilidad de quemar al oponente). Su comportamiento puede incluir el uso ocasional de objetos que potencien su poder o restauren su energía mágica.

Sanador

Es un apoyo en la guild. En lugar de atacar directamente, puede **restaurar la vida** de los aliados. Su intervención no garantiza el éxito, ya que la efectividad de las curaciones puede variar de forma aleatoria.

Ejemplo de acción

```
Turno 5: Taren canaliza energía curativa. Efectividad: 80%. Arthos recupera 24 puntos
de vida.
```

Otros roles adicionales pueden crearse libremente, como, Hechicero Oscuro (daño a ambos equipos), o Paladín (ataques moderados con defensa elevada). El sistema debe permitir ampliar estos tipos sin alterar el funcionamiento general. ***Esto puede pedirse por ejemplo durante la sustentación***

Los oponentes

Los oponentes representan a las guilds rivales que compiten en el torneo. Poseen las mismas características que los héroes (nombre, nivel, vida, ataque y defensa y rol) pero pertenecen al equipo contrario. Los oponentes deben ser lo suficientemente variados para representar desafíos reales: algunos más fuertes, otros más defensivos o con habilidades impredecibles

El inventario

En Lyrenhold, antes de cada torneo, las guilds reciben un conjunto limitado de objetos mágicos, conocidos como reliquias de batalla. Estos objetos se almacenan en un inventario global del torneo, al que el guild de los héroes tiene acceso antes de iniciar un enfrentamiento.

El número total de objetos disponibles es limitado, y cada héroe solo puede portar dos por combate). Una vez que un objeto se usa, desaparece del inventario global y no puede reutilizarse en la misma batalla.

El inventario se distribuye durante el combate así:

1. ***Durante la preparación:*** *Antes de iniciar un combate, el jugador puede decidir cómo distribuir los objetos disponibles entre sus héroes. El sistema debe permitir asignar o retirar objetos del inventario antes de comenzar la batalla. El inventario global del torneo tiene una cantidad total limitada de objetos mágicos*
2. ***Durante el combate:*** *Solo los héroes que tengan un objeto asignado pueden usarlo, y únicamente una vez por batalla. El uso del objeto ocurre en el turno del héroe y debe generar un efecto que se muestra en consola (por ejemplo, curación, aumento de ataque o defensa).*
3. ***Después del combate:*** *Los objetos usados se eliminan del inventario global. Los objetos no utilizados pueden conservarse para futuros torneos.*

Tipos de objetos mágicos

El sistema debe incluir objetos mágicos, con efectos distintos que el equipo desarrollador debe definir. Además de los del ejemplo, el equipo debe deberán agregar ***tres objetos más*** *que sigan una lógica coherente con el mundo del torneo.*

Los objetos mágicos de ejemplo son:

Poción de Vida

Un objeto que restaura la salud de un héroe. Su efecto es ***variable y aleatorio****, recuperando entre 20 y 40 puntos de vida.*

Amuleto de Furia

Aumenta el ataque del héroe durante un turno. El incremento puede oscilar entre +5 y +10 puntos según el azar. Después de dos turnos, el efecto desaparece.

Escudo Bendito

Aumenta temporalmente la defensa del héroe durante un turno. El efecto depende del azar: puede aumentar entre +10 y +20 puntos de defensa.

Gestión y CRUD de Objetos Mágicos

El inventario global del torneo gestiona los objetos mágicos. Operaciones mínimas:

• Crear: registrar un tipo de objeto con su efecto (p. ej., curación 20–40) y stock inicial.
• Listar/Consultar: ver objetos y sus detalles (rango de efecto, usos por combate, stock).
• Actualizar: modificar stock y, si el equipo lo decide, parámetros del efecto (documentarlo en el README).
• Eliminar: solo si el stock = 0.
• Asignar/Retirar (previo al combate): asignar descuenta stock; retirar devuelve stock si no se usó.
Ejemplo de interacción por consola:

```
=== INVENTARIO GLOBAL DEL TORNEO ===
1) Crear objeto mágico
2) Listar objetos
3) Consultar objeto
4) Actualizar objeto o stock
5) Eliminar objeto (si stock = 0)
6) Asignar objeto a héroe
7) Retirar objeto asignado de héroe
8) Volver
```

Combates

Durante los combates, los oponentes deben actuar siguiendo un comportamiento lógico o semi-aleatorio que el equipo tiene que definir y explicar en la documentación del proyecto. Por ejemplo, alternar entre atacar y curar, dependiendo del tipo de héroe

Interacción con el sistema

Primera etapa

El sistema debe permitir visualizar los héroes existentes, agregar nuevos, consultar sus características y retirarlos de la guild. Esta funcionalidad representa la base del sistema de gestión de personajes.

Creación y gestión de la Guild

La Guild es el punto de partida del sistema. Al iniciar el programa, la Guild debe existir y presentar un conjunto mínimo de héroes precargados mediante un método de inicialización, de modo que el usuario pueda ejecutar una batalla sin necesidad de ingresar datos por consola. Así mismo deben cargarse los enemigos.

Requisitos:
• Crear la Guild al inicio, mostrando un mensaje de bienvenida y el estado inicial de sus miembros.
• Cargar automáticamente héroes base con atributos (nombre, rol, nivel, vida, ataque y defensa)

•Permitir la gestión de héroes y enemigos durante la ejecución: agregar, consultar, listar y retirar.
• Garantizar identificadores únicos y evitar duplicados.

Por ejemplo, la consola podría mostrar:

```
=== BIENVENIDO A LA GUILD DE LYRENHOLD ===
Se ha creado tu Guild.
Cargando héroes iniciales...
- H01 Arthos (Guerrero) Nivel 2
- H02 Lyra (Maga) Nivel 1
- H03 Taren (Sanador) Nivel 1

=== GESTIÓN DE HÉROES ===
1) Agregar héroe
2) Consultar héroe
3) Listar héroes
4) Retirar héroe
5) Volver
```

Validaciones esperadas:

• No retirar héroes inexistentes.

• No iniciar batallas sin héroes vivos.

• Al retirar un héroe, desaparece de todos los listados y no puede ser seleccionado para la arena.

Segunda etapa. La arena

En la arena se enfrentan los equipos. El combate es por turnos, alternando acciones entre los héroes y los oponentes. Cada turno puede involucrar ataques, curaciones, defensa o uso de objetos mágicos. Para hacer el sistema más realista, los resultados deben incluir elementos aleatorios, como golpes críticos, fallos de hechizo o variaciones de daño. En la Arena, cuando es hora del combate, cada tipo de héroe interpreta esa orden de manera distinta. Por ejemplo,

- *El Guerrero ataca con su espada.*
- *El Mago lanza un hechizo.*
- *El Sanador restaura la vida de un compañero.*

Los resultados deben incluir elementos de aleatoriedad controlada, para hacer las batallas menos deterministas. Los factores aleatorios pueden aplicarse en:

- *La variación del daño (por ejemplo, ±10%).*
- *La probabilidad de golpes críticos o fallos.*
- *La efectividad de habilidades curativas o de defensa.*
- *El efecto de los objetos mágicos.*

Finalización del juego

El combate en la Arena de Lyrenhold concluye cuando uno de los equipos ha perdido a todos sus integrantes. Un personaje se considera derrotado cuando sus puntos de vida (vida) llegan a 0 o menos.

- *Un equipo se considera eliminado cuando todos sus integrantes están derrotados.*
- *El ganador es el equipo que todavía tiene al menos un personaje con vida mientras que en el otro equipo todos los miembros han sido eliminados. En el flujo del programa, la comprobación del ganador ocurre al finalizar cada turno completo:*
    - *El héroe o enemigo realiza su acción.*
    - *Se actualizan los puntos de vida del objetivo.*
    - *El sistema verifica si ese objetivo ha caído (vida <= 0).*

- *Si todos los miembros de un equipo están caídos, el combate termina.*

La Arena debe mostrar un resumen del resultado, incluyendo:

- *Nombre del equipo ganador.*
- *Héroes supervivientes.*
- *Número total de turnos del combate.*
- *Objetos mágicos utilizados.*

```
Un ejemplo de terminación del juego sería el siguiente:
Turno 9: Arthos ejecuta Golpe Final. Daño: 27. Vida de Dravos: 0
Dravos ha sido derrotado.
Selene intenta curarlo... pero llega demasiado tarde.
=== FIN DEL COMBATE ===
Equipo ganador: Guild del jugador
Motivo: todos los oponentes han sido derrotados.
Héroes supervivientes: Arthos, Lyra
Duración: 9 turnos
Objetos usados: 3
```

Registro en archivos

El sistema debe permitir guardar y cargar una sola categoría de información (a elección del equipo de desarrollo): pueden ser los héroes, los oponentes, los objetos o las batallas en un archivo en formato JSON.

Alcance técnico

- Lenguaje y estándar: C++
- Estructuras: uso visible de vector y unordered_map.
- Memoria: manejo con punteros crudos (liberación explícita cuando aplique).
- Entrada/Salida: ejecución en consola con sintaxis básica (sin librerías externas de UI).
- Persistencia: una sola categoría en JSON
- Equipos de 2 personas.
- Todos los miembros deben practicar activamente la codificación.
- Uso continuo de git y github para mostrar avances del proyecto con commits de TODOS los miembros del equipo. Siguiendo esta regla:
  - Cada commit debe incluir un mensaje con la estructura:
    Qué cambié → Por qué lo hice → Qué espero que mejore

Uso de IA

Permitido

- Consultar dudas de sintaxis C++, STL, lectura/escritura de archivos.
- Pedir ejemplos que ayuden a entender cómo resolver aspectos del proyecto (sin copiar/pegar literal).
- Mejorar mensajes de consola y redacción del README.

No permitido

- Hacer el diseño de esta solución. Pueden revisar su propuesta, pedir explicaciones, no pedir que lo haga
- Solicitar código específico para resolver partes de este enunciado
- Se penalizará el proyecto que tenga código sobre el que los estudiantes no puedan hacer cambios, modificaciones, eliminaciones o adiciones con fluidez durante la sustentación. La nota del proyecto en ese caso será de 1 para

todos los integrantes. Desactive las asistencias de IA de su IDE para que no tenga riesgo de incluir sin intención código que el equipo no entiende.

Entregables

- **Video de Presentación del Proyecto:**
  - Presentación del proyecto: breve descripción del propósito, funcionamiento y resultados obtenidos (máximo 2 minutos).
  - Reflexión metacognitiva: cada integrante explica, desde su perspectiva, cómo abordó el problema y qué estrategias utilizó (planificación, depuración, comparación, etc.); qué errores cometió, cómo los detectó y qué aprendió de ellos; qué cambió en su forma de pensar o programar durante el curso.
  - Reflexión colaborativa: el equipo describe cómo la interacción y retroalimentación entre compañeros influyó en la solución final.
  - Requisitos formales:
    - Duración máxima: 6 minutos.
    - Subir el video como "no listado" en YouTube y entregar el enlace en la plataforma.
    - Todos los integrantes deben participar y aparecer en el video, evidenciando su contribución y reflexión personal.
    - Se evaluará la calidad del razonamiento y la profundidad de la reflexión, en especial que se analice críticamente el propio proceso de aprendizaje, se identifiquen errores, estrategias y transformaciones personales, no la edición del video.
- **Diagramas UML como Mermaid dentro del proyecto:**
  - ***Versión inicial*** **(antes de programar).**
  - ***Versión ajustada*** **(después de implementar los héroes y objetos mágicos).**
  - ***Versión final*** **(después de integrar la arena).**
- **Código fuente del programa**. Use este github classroom para iniciar el proyecto:
- **README en el repositorio [Manual técnico]**: Con presentación general del proyecto, diagrama de clases UML y algunas imágenes del proyecto funcionando.
- **BITACORA.md**

  Cada equipo mantiene un archivo bitacora.md con tres columnas. Este archivo se debe ir construyendo a lo largo de la ejecución del proyecto. Cada miembro debe contribuir con al menos tres entradas personales donde analice su propio proceso de razonamiento.

  Por ejemplo

Fecha	Qué decidí	Por qué lo hice así	Responsable
8/nov	**Crear clase base Personaje**	**Para aplicar herencia y compartir atributos**	**Juan camilo**
9/nov	**Usar unordered_map para héroes**	**Permite búsqueda por ID eficiente**	**Equipo en conjunto**

Autoevaluación y coevaluación

- Informe de autoevaluación en un documento individual en PDF [SUBIDO EN EL BRIGHSPACE] en el que explique:
  - ¿Qué decisión de diseño fue más difícil de tomar y por qué?

- ¿Qué error cambió tu comprensión de la programación orientada a objetos?
- ¿Cómo influyó tu compañero en tu forma de resolver problemas?
- ¿Qué harías distinto si pudieras reiniciar el proyecto?
- ¿Qué aprendiste sobre ti como programador o programadora?
- Qué hizo cada uno de los miembros del equipo
- Evaluación y co-evaluación: califique de manera individual y para cada compañero cada uno de los siguientes elementos.
    - - Colaboración y trabajo en equipo
    - - Responsabilidad y compromiso
    - - Contribución al Desarrollo del Trabajo
    - - Nota consolidada (promedio de las anteriores)

Tenga en cuenta la siguiente rúbrica

Colaboración y trabajo en equipo

- **Excelente (5)** El estudiante participa activamente y mejora la dinámica del equipo mediante una colaboración efectiva, mostrando respeto y apertura hacia las ideas de los demás.
- **Bueno (4)** Participa regularmente y colabora bien, aunque en ocasiones puede ser pasivo. Respetuoso con los compañeros la mayoría del tiempo.
- **Adecuado(3)** Participa sin tomar un rol activo, realiza lo mínimo necesario para no obstaculizar el equipo. Puede mejorar en respeto y colaboración.
- **Insuficiente(2)** Participación mínima o negativa, a menudo no colabora o desestima a los demás, afectando negativamente la dinámica del equipo.

Responsabilidad y compromiso

- **Excelente (5)** Cumple con todas las tareas asignadas a tiempo, mostrando un alto nivel de compromiso y capacidad para tomar iniciativas adicionales.
- **Bueno (4)** Generalmente cumple con las tareas en los plazos establecidos y muestra un buen nivel de compromiso.
- **Adecuado(3)** Cumple con las tareas básicas, pero raramente toma iniciativas adicionales y a veces no cumple con los plazos.
- **Insuficiente(2)** Falta frecuentemente a sus responsabilidades, mostrando poco compromiso y afectando el rendimiento del equipo.

Contribución al Desarrollo del Trabajo

- **Excelente (5).** Aporta ideas y soluciones que son esenciales para el proyecto. Participa de manera activa en la planificación y ejecución de tareas clave.
- **Bueno(4).** Realiza contribuciones que benefician al proyecto. Participa en la planificación y ejecución de tareas.
- **Adecuado(3)** Realiza contribuciones que cumplen con los requisitos básicos del proyecto. Participación regular en la planificación y ejecución de tareas.
- **Insuficiente(2)** Realiza pocas o ninguna contribución al desarrollo del trabajo. Participa poco o nada en la planificación y ejecución de tareas.

Cronograma

Para favorecer el aprendizaje del proceso y no solo del producto final, el proyecto se desarrollará con entregables parciales en las próximas tres semanas. Este es el cronograma propuesto

Semana	Funcionalidades	Descripción
1	**Revisión del diseño inicial avanzado, inicio de la implementación de los héroes y la Guild**	**Diagrama UML con justificación de clases y relaciones.**
1	**Implementación funcional de la Guild (gestión de héroes e inicialización), gestión de objetos mágicos**	**Implementación funcional de la Guild (gestión de héroes e inicialización).**
2	**Asignación y regreso de objetos a los héroes, simulación combates en la árena/ documentación/bitácora**	**Incorporación del inventario de objetos y objetos mágicos**
3	**Persistencia, menus finales, preparación entregables finales**	**Persistencia, menús finales, elaboración de entregables**
3	**Sustentación final**	**Ajustes y sustentación final**

La fecha de sustentación es el 28 de noviembre, según se asignará su turno de sustentación. Durante la sustentación (individual) tendrá que hacer cambios en vivo en el proyecto. La sustentación durará 15 minutos. En ese tiempo tiene que ser capaz de hacer los cambios que solicite la profesora. Debe tener las asistencias de IA desactivadas.

Rúbrica de evaluación

La nota de la profesora dependerá de los criterios que se describen a continuación y su capacidad para sustentar su trabajo.

	5	4	3	2	1	0
Entregables (10%)	Los informes contienen toda la información solicitada y tiene alta calidad en cuanto a estilo y formato.	Se entregaron todos los informes. En términos de contenido están completos pero podría mejorar en cuanto a estilo o formato	Se entregaron todos los informes. En términos de contenido están completos pero podría mejorar en cuanto a estilo Y formato	Faltan algunos de los informes pero los entregados tiene buen estilo y formato	Faltan algunos de los informes y los entregados necesitan mejoras de estilo y formato	No se entregaron los informes
Diseño (30%)	El diseño refleja un pensamiento abstracto sólido: el estudiante justifica la estructura elegida, explica las relaciones entre clases y muestra comprensión profunda de los principios de POO (abstracción, encapsulamiento,	El diseño cumple técnicamente y muestra razonamiento adecuado. Se justifica parcialmente la elección de clases y relaciones.	El diseño es correcto pero poco argumentado. Hay comprensión parcial de las relaciones y de la lógica detrás de ellas ó Faltó detectar algunas clases importantes. Faltó	Faltó detectar la mayoría de clases importantes Faltó detectar muchos de los atributos y métodos importantes. Tiene muchas relaciones incorrectas	El diseño no satisface los requisitos ó no hay justificación ni coherencia en el diseño; las decisiones parecen imitativas o sin criterio propio.	No se entregó el diseño

	herencia y polimorfismo). Los diagramas evidencian decisiones fundamentadas.		detectar algunos de los atributos y métodos importantes. Tiene algunas relaciones incorrectas. El diseño no corresponde a lo codificado.			
Funcionalidad (30%)	Cumplió con todos los requisitos.	Fueron desarrollados mínimo el 75% de los requisitos	El diseño responde al 75% de los requisitos / podría mejorarse	Fueron desarrollados mínimo el 25% de los requisitos	Fueron desarrollados menos del 25% de los requisitos	La funcionalidad no responde a lo solicitado
Proceso y reflexión (30%)	El equipo demuestra pensamiento crítico en todo el proceso: identifica errores, explica cómo los resolvió y qué aprendió. Justifica las decisiones de diseño y reconoce limitaciones. Las reflexiones son profundas, honestas y muestran evolución cognitiva. La bitácora es analítica, no descriptiva	Hay reflexión sobre dificultades y aprendizajes, aunque sin análisis profundo. Se evidencian correcciones y decisiones razonadas.	La reflexión es superficial o centrada en el producto. La bitácora describe acciones, pero no razones		No hay evidencia de pensamiento crítico ni autorreflexión. Las decisiones parecen copiadas o poco comprendidas	No se entregó la bitacora

Rúbrica de propiedad intelectual

	1	0.8	0.6	0.4	0.2	0
Sustentación	Es evidente que el estudiante entiende el código que desarrolló lo explica con claridad y responde correctamente a las preguntas.	La sustentación es buena pero se evidenció inseguridad del estudiante para explicar algunas partes del trabajo desarrollado o para responder algunas preguntas.	La sustentación es aceptable se evidencia que el estudiante desarrolló el código pero le cuesta trabajo explicar aspectos del código.	La sustentación es regular se evidenció inseguridad del estudiante para explicar gran parte del trabajo desarrollado o para responder muchas de las preguntas. Parece que el código no hubiera sido desarrollado por el estudiante.	El estudiante demuestra que entiende partes del código, pero no tiene claro cómo se relacionan con la funcionalidad solicitada.	Se evidencia que el estudiante no entiende el código desarrollado, no es capaz de responder a las preguntas formuladas de manera correcta.
Uso ético de IA	El estudiante demuestra criterio al explicar cómo usó IA para aprender o verificar ideas, no para reemplazar el razonamiento propio. Reconoce qué fragmentos adaptó y qué comprendió		Usó IA de apoyo pero con dependencia parcial: no logra justificar todas las decisiones de diseño o el código fuente			Se evidencian fragmentos de código no comprendidos o explicaciones incoherentes con el proyecto