

# TP2: YuGiOOP

## Detalles del trabajo

### Objetivos

- Analizar el problema planteado correctamente.
- Diseñar un modelo general de solución utilizando correctamente los conceptos de **programación orientada a objetos**.
- Demostrar el conocimiento de principios de diseño y de las buenas prácticas de programación.

### Indicaciones del trabajo

#### Grupos de trabajo

El trabajo se desarrollará de forma grupal, en grupos de 4 alumnos. El trabajo práctico debe realizarse utilizando Git y Github como herramienta de colaboración.

Todos los alumnos integrantes del grupo **deben** participar activamente del desarrollo del trabajo de forma equitativa. Mientras la participación sea equitativa, queda a criterio de cada grupo la forma de trabajo y organización.

#### Tecnología

El trabajo debe realizarse en **Java**, utilizando cualquier distribución de **Java 21**. El proyecto debe ser un proyecto Maven. Debe poder ejecutarse utilizando un comando de Maven.

#### Tiempo de trabajo y entregas parciales

Se contarán con aproximadamente 5 semanas completas para llevar a cabo el desarrollo del TP.

El trabajo cuenta con una entrega parcial no obligatoria en la semana 2 del trabajo, para hacer junto al corrector asignado una evaluación prematura del diseño planteado, y la separación de incumbencia entre las distintas partes del programa.

A final de la fecha establecida, debe realizarse la entrega final del trabajo.

## Modalidad de entrega

Para realizar la entrega se debe:

- Github:
  - El repositorio tiene que ser privado y debe estar su corrector invitado al mismo.
  - Para realizar la entrega se debe crear un branch llamado **tp-2**, con el código correspondiente a la entrega. No se debe modificar más luego de la fecha de entrega.
  - Se debe contar con un **README.md** que explique cómo correr el programa “desde cero”, cómo instalar las dependencias y otras explicaciones pertinentes. El código debe poder correrse con un comando que compile el código y lo ejecute, **no es válido subir un ejecutable y que el comando simplemente lo corra**.
- Mail: Mandar un mail a [algoritmos3.fiuba@gmail.com](mailto:algoritmos3.fiuba@gmail.com) indicando número de grupo y sus integrantes (nombre, apellido y padrón), aclarando cuál es la branch y el repositorio de la entrega. El mail debe llevar adjunto el informe en formato PDF y asunto debe **TP2 - <nombre del grupo>**.

## Fecha de entrega

La entrega definitiva, que debe alinearse a lo especificado en la [Modalidad de entrega](#), debe realizarse con anterioridad a las 23:59hs del **miércoles 11/06/2025**.

# Enunciado: Desarrollando el YuGiOOP

## Dominio

### Introducción

*El destino del mundo está en juego... y solo las cartas decidirán el resultado.*

*En este trabajo práctico, te proponemos un desafío que va más allá del código: **crear tu propio duelo de monstruos**. Inspirado en el universo de Yu-Gi-Oh!, este juego de cartas no es solo una batalla entre jugadores, sino un enfrentamiento entre la astucia, la estrategia y el poder de tu imaginación.*

*Tu misión es clara: construir un sistema donde las cartas cobren vida, los hechizos alteren el curso del combate y las criaturas más temibles se enfrenten en un duelo sin igual. No basta con programar: **vas a tener que pensar como un duelist**a. ¿Qué carta jugar? ¿Cuándo atacar? ¿Cómo anticiparse al enemigo?*

*Cada línea que escribas será una jugada. Cada decisión de diseño, una trampa oportuna o un ataque sorpresa. Y al final, cuando tu juego funcione y se enfrenten los mazos... solo uno quedará en pie.*

*El duelo ha comenzado. ¿Estás listo para demostrar que tenés el corazón de las cartas?*

### Descripción juego general

Duelo entre dos jugadores que usan cartas para invocar monstruos, lanzar hechizos y activar trampas con el objetivo de reducir los Life Points (LP) del oponente de 8000 a 0. El mazo de cada jugador está compuesto de 40 cartas.

El objetivo es el de reducir los 8000 puntos de vida del oponente a 0. También un jugador puede ganar si:

1. El oponente no puede robar carta al inicio de su turno, el caso ocurre cuando se quedó sin cartas.
2. El oponente se rinde.

### Turnos

Cada jugador puede realizar acciones durante su turno, que se divide en distintas fases.

1. Fase de robo: El jugador roba una carta de su mazo.
2. Fase Standby: Se resuelven efectos que ocurren al inicio del turno
3. Fase principal 1: En esta se puede:
  - a. Invocar un monstruo: normal o especial, modo ataque o defensa. Las cartas se pueden invocar en el tablero boca abajo o boca arriba

- i. Al estar boca abajo el adversario no puede ver cual carta es pero tampoco se puede utilizar.
- b. Activar cartas mágicas o de trampa
- c. Colocar cartas: hechizo y/o trampa limitado por los slots del tablero
4. Fase de batalla (opcional): El jugador puede atacar con sus monstruos al oponente. Solo pueden atacar los monstruos que estén en posición de ataque, se pueden cambiar de posición en esta fase pero luego no pueden volver a ser defensivos hasta el turno siguiente. Cada monstruo puede atacar una sola vez por turno. Esta fase no está disponible el primer turno del jugador que empieza la partida.
5. Fase cambios: Se pueden alternar las cartas en el campo de juego entre ataque y defensa.
6. Fase final: finaliza el turno y se resuelven los efectos que ocurren al final del turno.

## Cartas

### Monstruos

- ★ Sus características principales son el Nivel, su ATK (ataque) y su DEF (defensa). Además cuentan con un Elemento (Fuego, Viento, Agua, Tierra, Oscuridad, Luz, etc) y un Atributo (Guerrero, Dragón, Zombie, Bestia, Hada, Máquina, etc).
  - ★ Se puede invocar 1 monstruo normal por turno.
  - ★ Dependiendo del nivel del monstruo, hay que hacer sacrificios/tributos para invocarlos. Por ejemplo, invocar un monstruo nivel 5 o 6 requiere tener otro monstruo en campo al cual sacrificar (enviar al cementerio).
- Niveles de sacrificios:
- Niveles 1-4: no requieren sacrificios
  - Niveles 5-6: requieren un sacrificio
  - Niveles 7 o mayor: requieren dos sacrificios



Invocación normal con sacrificio (nivel 9) || Invocación normal (nivel 3)

## Mágicas (Hechizos)

- Solo se pueden jugar en el turno de jugador.
- Hay distintos tipos de hechizos y deben implementarse todos.
- Los tipos de cartas de hechizos que pueden haber son:
  - **! Segunda oportunidad:** El jugador puede seleccionar una carta de su cementerio y colocarla en su mano. Luego de activada va al cementerio.
    - Ejemplo: *Olla de la codicia* permite al jugador robar dos cartas del mazo.
  - **! Equipamiento:** Se equipan a un monstruo (del jugador que la invoca o del rival). Le dan poder extra o lo debilitan, modificando alguna de sus características circunstancialmente por N turnos. La carta se mantiene en el campo los N turnos antes de irse al cementerio.
    - Ejemplo: *Hacha de la Desesperación*, el monstruo equipado gana 1000 ATK durante 2 turnos.
  - **Robo de cartas:** El jugador roba N cartas de su mazo. Luego de activada va al cementerio.
    - Ejemplo: *Olla de la codicia* permite al jugador robar 2 cartas del mazo.
  - **Defensivos:** Un monstruo del oponente no puede atacar por N turnos.
    - Ejemplo: *Espadas de Luz Reveladora*, el oponente no puede atacar durante 3 turnos. La carta se mantiene en el campo 3 turnos antes de irse al cementerio.
  - **Campo:** Se colocan en una zona especial del campo, común a ambos jugadores. Afectan a todas las cartas, cambiando las condiciones del juego. Solo puede haber una activa a la vez (si se coloca otra, reemplaza a la anterior).
    - Ejemplo: *Yami*, aumenta el ATK y DEF de monstruos de tipo Mago y Demonio. Reduce los de Hada. Afecta a todos los monstruos en el campo, de ambos jugadores.



Normal

Continua

Equipo



Campo

### Cartas Trampa

- Se colocan en el turno del jugador (siempre boca abajo) y se activan en el turno del oponente cuando éste realiza alguna acción determinada.
- Son de un solo uso, y de efecto inmediato. Se activan en respuesta a algo (ataques, invocaciones, efectos).
- Cada trampa cuenta con el efecto que posee y una condición de activación a la cual responden.



- Los efectos de las cartas son los siguientes:
  - ★ **! RobarCartas:**
    - *Descripción: El jugador roba N cartas de su mazo.*
    - *Ejemplo: Esperanza de Escape – El jugador roba 3 cartas cuando el oponente juega una carta mágica.*
  - ★ **Descartar:**
    - *Descripción: El oponente debe descartar N cartas de su mano.*
    - *Ejemplo: Trampa Psíquica – Si el oponente ataca, se descartan 2 cartas aleatorias de su mano.*
  - ★ **AumentarDefensa:**

- *Descripción: Aumenta la DEF de monstruos en juego del jugador en N puntos, por M turnos.*
- *Ejemplo: Fuerza Oculta – Al recibir un ataque, los monstruos defensores ganan 500 DEF durante el turno del oponente.*

★ **ReducirAtaque:**

- *Descripción: Aumenta la DEF de monstruos en juego del jugador en N puntos, por M turnos.*
- *Ejemplo: Sello del mal – Al atacar, el monstruo atacante pierde 1000 ATK.*

★ **Contraataque:**

- *Descripción: Causa N de daño al atacante.*
- *Ejemplo: Reflejo de Daño – El atacante recibe 500 de daño cuando coloca una carta en el tablero.*

★ **RecuperarVida:**

- *Descripción: Recuperás N puntos de vida.*
- *Ejemplo: Barrera Espiritual – Ganas 500 LP cuando el oponente juega un hechizo.*

★ **DestruirCarta:**

- *Descripción: Destruye una o más cartas seleccionadas del campo.*
- *Ejemplo: Fuerza del Espejo – Cuando un monstruo del adversario ataca, destruye todos los monstruos en Posición de Ataque de tu adversario.*

★ **CambiarPosicion:**

- *Descripción: Cambia la posición de un monstruo (de ataque a defensa, o viceversa).*
- *Ejemplo: Kunai con Cadena – Cuando tu oponente termina su turno, pasa todos sus monstruos a Posición de Ataque.*

- Las **condiciones de activación** de las trampas quedan a criterio de cada grupo, y bien podrían estar atados a monstruos o jugadores específicos, o no. En los ejemplos listados arriba se mencionan algunas ideas.

## Campo de juego/Tablero

Cada jugador tiene:

- ★ 5 casilleros para los monstruos.
- ★ 5 casilleros para hechizos/trampas.
- ★ 1 zona de campo (para hechizos de campo).
- ★ 1 zona de cementerio (ahí van las cartas descartadas, monstruos destruidos, hechizos/trampas finalizados, etc)
- ★ 1 zona de mazo (40-60 cartas).



## Detalles de juego

- ★ Al comienzo del duelo, cada jugador roba 5 cartas de su mazo.
- ★ Siempre cada jugador obtiene una carta de su mazo por turno, al principio del mismo.
- ★ Si el adversario no tiene monstruos en juego se le puede atacar a sus puntos de vida directamente.
- ★ No hay límite de cartas para tener en la mano durante el turno: se puede tener cualquier cantidad de cartas en la mano mientras estás jugando tu turno.  
Pero al finalizar el turno, si se tiene más de 6 cartas, se deben descartar cartas hasta quedarte con 6. Las cartas descartadas van al cementerio.
- ★ Hay hechizos/trampas que te permiten sacar más cartas que la mandataria al inicio de tu turno (por ejemplo, Olla de la Codicia), y también que obligan a descartar cartas (por ejemplo, Requisamiento Forzoso). Algunas también permiten recuperar cartas del cementerio.

## Batalla

- ★ Solo se puede atacar en la fase de batalla.
- ★ Un monstruo en posición de ataque ataca a otro, que puede estar en posición de ataque o defensa.
- ★ Si el ATK del atacante > ATK del adversario → se destruye el monstruo del adversario y se le sacan puntos de vida, la diferencia entre los puntos de ataque de tu monstruo y los del monstruo del adversario.
- ★ Si el ATK del atacante < ATK del adversario → se destruye el monstruo atacante y se le sacan puntos de vida, la diferencia entre los puntos de ataque de tu monstruo y los del monstruo del adversario.
- ★ Si el ATK del atacante == ATK del adversario → ambos se destruyen, ninguno recibe daño.
- ★ Si se ataca a un monstruo en posición de defensa:
  - Si el ATK del atacante > DEF del adversario → se destruye el monstruo atacado (pero el adversario no recibe daño).
  - Si el ATK del atacante < DEF del adversario → se destruye el monstruo atacante y el jugador recibe daño por la diferencia entre sus puntos de ATK y los de DEF del adversario.

## Requerimientos

*Importante: En caso de tomar una decisión respecto a una definición ambigua del dominio o los requerimientos, volcarlo en la correspondiente sección de hipótesis del informe.*

## Modelo

En un proyecto Java, se debe volcar la representación del modelo del juego, con todas sus características enunciadas en el [Dominio](#), contemplando tanto la lógica básica como sus *corner-cases*.

## Jugabilidad

Se debe jugar una partida entre 2 jugadores en una misma computadora, ingresando las acciones que espera realizar por la terminal.

Al comenzar una partida, el programa debe pedir los nombres de ambos jugadores. Luego, a cada jugador se le asignará un mazo de N cartas, el cual debe leerse desde un archivo de configuración, el cual cada grupo podrá decidir el lenguaje y formato en que escriba. Esto permitirá versatilidad a la hora de jugar partidas más diversas (además de que facilitará las pruebas que realice cada grupo).

Apenas se da inicio a la partida, se le repartirán 5 a cada uno al comenzar. También el juego decidirá qué jugador comenzará en el primer turno.

Antes de cada turno, se debe dar una breve descripción del estado de la partida y del campo de batalla (datos vitales de cada jugador, número de cartas boca arriba y boca abajo, etc). Además, cada jugador tomará una carta.

En cada turno, el jugador debe ser capaz de realizar cualquiera de las acciones que son posibles realizar según lo especificado para la *Fase Principal* del turno y según el contexto actual de la batalla (cartas a disposición, cartas presentes). Algunas de estas acciones pueden ser invocar cierto monstruo, activar determinadas cartas, atacar, entre otros.

Al finalizar cada turno, ya sea porque no se pueden realizar más acciones o porque el jugador determina hacerlo, se deben realizar las acciones pertinentes y dar inicio al turno del siguiente jugador.

Al coronarse ganador alguno de los jugadores, ya que se da alguna de las alternativas de victoria, se debe mostrar un lenguaje correspondiente en pantallas.

#### Anexo: Estado de la partida

En todo momento debe poder verse y consultarse todo el estado de la partida. Esto es, desde lo más básico como los nombres de los jugadores y sus Life Points, y el campo de batalla actual; hasta los detalles más particulares como la cantidad de cartas restantes en el mazo, la cantidad de cartas del cementerio, etc.

Siempre cuanta más información dentro del dominio y permitida en el contexto actual de la partida (la manera en que se expone queda a criterio de cada grupo) mejorará la jugabilidad.

#### Anexo: Separación de responsabilidades

Es indispensable para lograr bajo acoplamiento y flexibilidad a la hora de desarrollar el juego que los grupos sean capaces de identificar y establecer barreras de responsabilidades entre el modelo del juego y lo que implica la interacción con el usuario.

Se recomienda basarse en un diseño a lo MVC.

#### Interfaz de usuario

*Importante: Esta siguiente sección contempla una jugabilidad vía terminal.  
Si se quisiera hacer una interfaz gráfica para el juego, debe ser charlado y coordinado previamente con el corrector asignado.*

Los mensajes de información y de error quedan para ser definidos por los alumnos. Se recomienda utilizar una librería como [JLine](#) para manejar el input del usuario y mostrar listas de opciones para limitar la cantidad de errores posibles. Como por ejemplo:

- 1 . A
- 2 . B

3 . C

*Ingrese número de la opción deseada : 1*

*Seleccionado : A*

**Recomendación:** Tener un módulo que ayude con este tipo de opciones que reciba una lista de opciones y retorna la seleccionada para tener la impresión por consola en un solo lugar.

Anexo: Menús de opciones

Si bien no se obliga a utilizar un esquema de menú específico a seguir, desde el curso sugerimos trabajar con un esquema de sub-menús para facilitar la usabilidad del juego y también simplificar el armado de acciones a nivel código.

El modelo de sub-menús plantea que haya un menú principal de opciones de usuario, y que se habilite un nuevo sub-menú con opciones más específicas para la opción principal elegida.

Por ejemplo, si en un primer momento se elige la opción “Invocar monstruo”, luego se mostrará un sub-menú donde cada opción serán aquellos monstruos que el jugador posea en su mano y también otro para determinar en qué slot posicionarlo. Al elegir un determinado monstruo, si este requiriese un sacrificio, se mostraría un nuevo sub-menú condicional para seleccionar las cartas a sacrificar. Cada sub-menú además debería contar con la posibilidad de regresar al estado/sub-menú anterior.

## Informe

Se espera la realización de un informe que contenga:

- Cómo se compila y corre el programa.
- Qué funcionalidades fueron implementadas.
- Un diagrama de clases UML **completo**, dando las explicaciones que consideren pertinentes respecto a la toma de decisiones en el diseño. Puede ser dividido en múltiples diagramas si lo consideran necesario.
- Al menos un diagrama de secuencia, de algún flujo que consideren interesante. Explicar por qué eligieron mostrar ese flujo.
- Una sección de las hipótesis tomadas durante la realización del trabajo.
- Conclusiones.

## Criterios de aprobación

Para que el trabajo práctico se considere aprobado, deberá cumplir con los siguientes requisitos mínimos:

- Cumplir con los objetivos enunciados en los detalles del trabajo, y también con los lineamientos y buenas prácticas del POO.
- Debe haber una correcta división de responsabilidades entre el modelo del juego y la parte interactiva con los jugadores.
- Implementar al menos 4 hechizos y 4 trampas, incluyendo dentro de ellas las obligatorias (indicadas con !)
- Es necesario implementar al menos 1 condiciones de activación de trampas (por ejemplo, cuando el oponente ataca o cuando termina su turno).
- Debe funcionar con al menos 1 mazo (es decir, con que este precargado en el juego alcanza). Para la aprobación, no es necesario que sea configurable por archivo.
- Se pueden omitir monstruos que requieran sacrificio para ser invocados.

Si bien estos son los requisitos mínimos para la aprobación, un TP se considerará *completo* si cumple con todos los requerimientos descritos en las secciones anteriores. Un TP incompleto, puede no ser considerado para la promoción.

Para este TP, los criterios para la promoción son los mismos que para la aprobación.

