# Tarea integradora 1: UNO Implementación del método de la ingeniería

## Fase 1: Identificación del problema:

El siguiente proyecto consiste en realizar la elaboración en código del juego UNO en consola utilizando métodos, algoritmos y estructuras de datos de tal manera que el funcionamiento sea eficiente. Haremos uso de las estructuras que hemos venido trabajando las cuales son pilas, colas, tablas hash y colas de prioridad.

* Gestión eficiente de cartas: Es necesario diseñar un sistema que maneje la distribución, el intercambio y el juego de cartas de manera rápida y sin problemas.
* Optimización del flujo de juego: Se necesita un flujo de juego claro y eficiente que incluya la distribución inicial de cartas, el turno de cada jugador y la determinación de ganadores de manera rápida y sin problemas.
* Manejo eficaz de estructuras de datos: Se requiere utilizar las estructuras de datos adecuadas (pilas, colas, tablas hash y colas de prioridad) de manera eficiente para almacenar y gestionar las cartas, las manos de los jugadores y otros elementos del juego.
* Implementación de reglas especiales: Se deben implementar las reglas especiales del juego UNO (cartas especiales como Cambio de Color, Roba 2, Revertir, Salto) de manera precisa y eficiente, asegurando que afecten adecuadamente al flujo del juego y al estado de los jugadores.
* Manejo de decisiones de juego: Se necesita un sistema que maneje de manera eficiente las decisiones de juego de los jugadores, como jugar una carta válida, robar cartas o pasar el turno, garantizando una experiencia de juego fluida.
* Determinación del ganador: Es esencial tener un mecanismo claro y eficiente para determinar cuándo se ha alcanzado el objetivo del juego (quedarse sin cartas en la mano) y cómo se identificará al ganador de manera justa y precisa.

## Requerimientos funcionales:

**Inicio del juego:** El sistema debe permitir al usuario iniciar el juego después de ingresar la cantidad de jugadores y los nombres de los jugadores.

**Reparto de cartas:** El sistema debe ser capaz de repartir siete cartas a cada jugador al comienzo del juego desde la baraja.

**Jugabilidad:** El sistema debe permitir a cada jugador realizar las siguientes acciones durante su turno:

1. Jugar una carta válida en la mesa.
2. Robar una carta de la baraja si no tiene una carta válida para jugar.

**Reglas del juego:** El sistema debe aplicar las reglas del juego, como permitir que los jugadores jueguen una carta que coincida en color o número con la carta en la parte superior de la mesa, o que jueguen cartas especiales (como "Roba 4", "Cambio de color", etc.).

**Ganar el juego:** El sistema debe detectar cuando un jugador se queda sin cartas en su mano y declararlo como el ganador del juego.

**Interacción de usuario:** El sistema debe proporcionar mensajes claros y concisos para guiar al usuario a través del juego, incluyendo mensajes de error cuando se ingresa una entrada incorrecta.

**Manejo de errores:** El sistema debe manejar situaciones de error, como intentos de jugar una carta inválida o robar una carta cuando no hay cartas restantes en la baraja.

## Fase 2: Recopilación de información necesaria:

**Características del juego:**

UNO es un juego de cartas para 2-10 jugadores, diseñado para ser sencillo y rápido de jugar.

El objetivo del juego es ser el primero en deshacerse de todas las cartas de tu mano gritando "¡UNO!" cuando solo te quede una carta.

El juego se juega con una baraja especial de 108 cartas, que incluye:

Cartas de números (0-9) en 4 colores diferentes (rojo, azul, verde y amarillo)

Cartas de acción como "Saltar", "Reversa" y "Tomar 2"

Cartas "Comodín" y "Cambio de color"

**Mecánica del juego:**

El juego comienza con una carta boca arriba en el centro de la mesa.

En tu turno, debes jugar una carta del mismo color, mismo número o misma acción que la carta anterior.

Si no tienes una carta que coincida, debes tomar una carta del mazo.

Las cartas de acción como "Saltar", "Reversa" y "Tomar 2" afectan al siguiente jugador.

Las cartas "Comodín" y "Cambio de color" permiten al jugador que las juega cambiar el color del juego.

Cuando solo te queda 1 carta, debes gritar "¡UNO!" para advertir a los demás.

El primer jugador en deshacerse de todas sus cartas gana la ronda.

**Variantes y reglas opcionales:**

Existen varias reglas y variantes opcionales que se pueden aplicar, como la "Regla de Robar", "Robar hasta que coincida", etc.

Estas reglas pueden variar según los grupos de jugadores y sus preferencias.

**Las características principales del juego de UNO en cuanto al número de jugadores son:**

Número mínimo de jugadores: 2

Número máximo de jugadores: 10

Esto significa que una partida de UNO puede ser jugada por entre 2 y 10 personas como máximo.

**Algunas aclaraciones adicionales sobre el número de jugadores en UNO:**

La baraja de UNO consta de 108 cartas, lo que permite que el juego se pueda disfrutar con un amplio rango de participantes.

A medida que aumenta el número de jugadores, la dinámica y rapidez del juego suele aumentar, ya que hay más turnos y mayor competencia.

Algunos grupos de jugadores prefieren limitar el número a 4-6 personas para mantener un ritmo más manejable y evitar que el juego se alargue demasiado.

## Fase 3: Búsqueda de soluciones creativas:

**Representación de las cartas:**

Utilizar una tabla hash (diccionario) para representar cada carta, donde la clave sea una combinación única del color y el valor de la carta.

Almacena en la tabla hash información relevante sobre cada carta, como su color, valor, tipo (número, acción, comodín, etc.).

**Manejo de la baraja:**

Usa una pila para almacenar todas las cartas de la baraja inicial.

Cuando se requiere tomar una carta, pop de la pila para obtenerla.

Al jugar una carta, agrega esa carta a una cola que represente la pila de descartes.

**Turnos de los jugadores:**

Utiliza una cola para mantener el orden de los jugadores.

Cuando sea el turno de un jugador, saca el primer elemento de la cola, procesa su movimiento y luego lo vuelves a agregar al final de la cola.

Maneja las cartas de acción (saltar, reversa, tomar 2) actualizando la cola según corresponda.

**Lógica del juego:**

Implementa funciones para validar si una carta puede ser jugada en el momento actual (mismo color, mismo número, misma acción).

Maneja el cambio de color con las cartas "Comodín" y "Cambio de color".

Implementa la funcionalidad de gritar "¡UNO!" cuando un jugador tenga una sola carta.

Detecta cuándo un jugador gana la ronda y termina el juego.

**Interfaz de usuario:**

Utiliza la consola para mostrar el estado actual del juego (cartas en la pila de descartes, cartas en mano de cada jugador, turno actual, etc.).

Implementa funciones para que los jugadores puedan ingresar sus movimientos (jugar carta, tomar carta, etc.).

**Manejo de excepciones y validaciones:**

Verifica que los jugadores sigan las reglas del juego (no jueguen cartas inválidas, griten "¡UNO!" cuando corresponda, etc.).

Maneja errores y situaciones inesperadas que puedan ocurrir durante el juego.

**Baraja de juego:** Se puede utilizar una lista, una lista enlazada, una vez se juega una carta se elimina de la baraja de juego. Una vez come agrega a la baraja de juego y elimina de la baraja de mazo.

**Lista simple**

**Ventajas:**

* Acceso a los elementos por índice, lo que facilita la implementación de operaciones como barajar, repartir, etc.
* Ocupan menos espacio en memoria en comparación con las listas enlazadas, ya que no requieren almacenar los punteros a los siguientes nodos.
* Suelen aprovechar la memoria contigua.

**Desventajas:**

* Insertar o eliminar elementos en posiciones determinadas de la lista puede ser mas costoso, ya que implica mover todos los elementos posteriores.
* Si la baraja es muy grande, usar una lista puede consumir mas memoria en total, ya que cada elemento debe almacenar el dato.

**Lista enlazada**

**Ventajas:**

* Insertar y eliminar elementos en cualquier posición es más eficiente, ya que solo se requiere actualizar los punteros.
* Pueden crecer o disminuir de tamaño de manera dinámica, sin necesidad de pre-asignar memoria.
* Pueden ser más eficientes en memoria si la baraja es muy grande, ya que cada nodo solo almacena el dato y los punteros.

**Desventajas:**

* Acceder a un elemento en una posición específica requiere recorrer la lista hasta llegar a ese punto, lo que puede ser más lento que el acceso directo de una lista.
* Ocupan más espacio en memoria debido a los punteros que deben almacenar para enlazar los nodos.
* Pueden ser menos eficientes en operaciones de recorrido completo de la baraja, ya que la memoria no es contigua.

**Baraja de descarte:** No va a tener iteración con el usuario que juega, solo mostrará la primera carta de la torre de la pila de cartas. Se puede utilizar lista simple, lista enlazada o pila.

**Lista simple**

**Ventajas:**

* Acceso rápido a la última carta agregada, ya que se encuentra en el último índice.
* Uso eficiente de memoria, ya que no requiere almacenar punteros adicionales.
* Recorrido de la pila de descarte es más sencillo y rápido al estar en memoria contigua.

**Desventajas:**

* Agregar nuevas cartas al final de la lista es una operación eficiente.
* Si la pila de descarte crece mucho, una lista simple puede consumir más memoria que otras estructuras.
* No se aprovecha la estructura LIFO (Last In, First Out) inherente al juego de UNO.

**Lista enlazada**

**Ventajas:**

* Agregar nuevas cartas al final de la lista es una operación eficiente, ya que solo se requiere actualizar el último puntero.
* Puede crecer o disminuir de tamaño de manera dinámica, sin necesidad de pre-asignar memoria.
* Se adapta bien a la estructura LIFO del juego de UNO, ya que la última carta agregada siempre se encuentra en el tope de la lista.

**Desventajas:**

* Acceder a la última carta agregada requiere recorrer toda la lista enlazada, lo que puede ser más lento que en una lista simple.
* Ocupa más espacio en memoria debido a los punteros necesarios para enlazar los nodos.
* Recorrido completo de la pila de descarte puede ser menos eficiente que en una lista simple.

**Pila**

**Ventajas:**

* Agregar nuevas cartas (Push) y obtener la última carta agregada (Top) son operaciones eficientes.
* Mantiene perfectamente la estructura LIFO (Last In, First Out) del juego de UNO.
* Puede crecer o disminuir de tamaño de manera dinámica.

**Desventajas:**

* No permite acceder a cartas en posiciones específicas, solo a la última carta agregada.
* Puede consumir más memoria que una lista simple, dependiendo de la implementación.

**Baraja de mazo:** Se come cuando no se posee ninguna carta posible para jugar en una determinada ocasión, además, cuando se obtiene un castigo o consecuencia de una carta (+2 o +4) o acumulación de varias de estas. Se puede usar una lista simple, lista enlazada, cola o pila.

**Lista simple**

**Ventajas:**

* Acceso rápido a la primera carta del mazo (tope de la pila) para realizar las operaciones de "tomar" y "devolver" cartas.
* Uso eficiente de memoria, ya que no requiere almacenar punteros adicionales.
* Recorrido secuencial del mazo es sencillo y rápido al estar en memoria contigua.

**Desventajas:**

* Insertar o eliminar cartas en posiciones intermedias del mazo puede ser ineficiente, ya que implica mover el resto de los elementos.
* Si el mazo crece mucho, una lista simple puede consumir más memoria que otras estructuras.
* No se aprovecha la estructura LIFO (Last In, First Out) inherente a la dinámica del mazo en UNO.

**Lista enlazada**

**Ventajas:**

* Insertar y eliminar cartas en el tope del mazo (inicio o final de la lista) es eficiente, ya que solo se requiere actualizar los punteros.
* Puede crecer o disminuir de tamaño de manera dinámica, sin necesidad de pre-asignar memoria.
* Se adapta bien a la estructura LIFO del mazo en UNO, ya que la última carta agregada siempre se encuentra en el tope.

**Desventajas:**

* Acceder a la primera carta del mazo (tope de la pila) requiere recorrer toda la lista enlazada, lo que puede ser más lento que en una lista simple.
* Ocupa más espacio en memoria debido a los punteros necesarios para enlazar los nodos.
* Recorrido completo del mazo puede ser menos eficiente que en una lista simple.

**Cola**

**Ventajas:**

* Operaciones de agregar (Enqueue) y extraer (Dequeue) cartas al mazo son eficientes.
* Mantiene el orden FIFO (First-In-First-Out) del mazo, lo que se adapta a la dinámica del juego de UNO.
* Puede crecer o disminuir de tamaño de manera dinámica.

**Desventajas:**

* Acceder a una carta específica en el mazo puede requerir recorrer toda la cola.
* No permite acceso directo a las cartas por índice, lo que puede dificultar algunas operaciones.
* Puede consumir más memoria que una lista simple, dependiendo de la implementación.

**Pila**

**Ventajas:**

* Operaciones de agregar (Push) y extraer (Pop) cartas al mazo son eficientes.
* Mantiene perfectamente la estructura LIFO (Last In, First Out) del mazo en UNO.
* Puede crecer o disminuir de tamaño de manera dinámica.

**Desventajas:**

* No permite acceder a cartas en posiciones específicas, solo a la última carta agregada (tope de la pila).
* Puede consumir más memoria que una lista simple, dependiendo de la implementación.

## Fase 4: Transición de la formulación de ideas a los diseños preliminares:

Luego de evaluar las posibles soluciones y verificar su comportamiento y utilidad a la hora de desarrollar el programa o los algoritmos del programa se concluye que en el diseño y desarrollo del juego de cartas UNO, se utilizarán las siguientes estructuras de datos:

**Pilas:** Se emplearán para representar el mazo de juego y el mazo de descarte.

**Colas:** Se implementarán para representar la manera en que se distribuirán las cartas desde el mazo de cartas a los jugadores, tanto al momento de repartir como al momento de robar. Cuando un jugador necesite robar una carta, se tomará la primera carta de la cola.

**Tablas Hash:** Se usarán para almacenar información adicional sobre las cartas, como su valor y su tipo (normal o especial). Esto permitirá hacer referencia a las cartas en las colas y pilas, en lugar de tener que guardar los objetos carta completos.

Colas de Prioridad: Se utilizarán para determinar el orden de juego cuando se usen cartas especiales como "Robar 2" o "Salto". Cuando se juegue una carta especial, se necesitará cambiar el orden en el que los jugadores están jugando, lo cual se logrará mediante una cola de prioridad donde cada elemento tendrá una prioridad asociada (la posición del jugador).

## Fase 5: Evaluación y selección de la mejor solución:

Criterios para el problema

* Utilizar la menor cantidad de estructuras evitando el consumo de memoria innecesario
* Utilizar algoritmos de manera eficiente
* Utilizar estructuras de datos óptima para cada situación