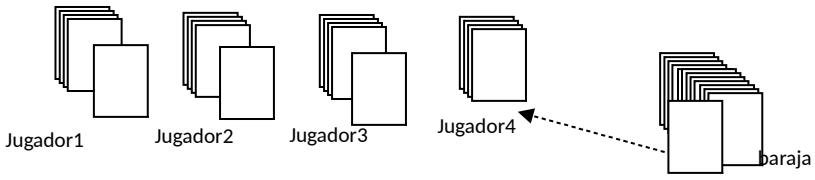
2. Partida de cartas (2,5 puntos)

Cuatro amigos van a jugar al juego de cartas de los SEISES. Las características del juego son las siguientes:

- La baraja contiene 40 cartas divididas en cuatro palos (oros, copas, espadas y bastos). Cada palo consta de 10 cartas numeradas del 1 al 10.
- Las cartas se reparten en 4 montones, es decir, 10 cartas por jugador.



- El funcionamiento de la partida es el siguiente:
    - El primer jugador juega con la primera carta de su montón,
        - Si puede la coloca en la mesa.
        - Si no puede colocar la carta en la mesa, la deja de nuevo en el fondo de su montón
    - Y pasa el turno al siguiente jugador.

- Un jugador puede colocar su carta en la mesa en 2 casos:
    - La carta es un 6.
    - La carta no es 6, pero es una carta consecutiva a otra que sí se encuentra en la mesa. Por ejemplo, si tengo el 2 de bastos y en la mesa está el 3 de bastos, ó si tengo el 8 de oros y en la mesa está el 7 de oros, etc.

- El juego acaba cuando un jugador se coloca la última carta de su montón, es decir, se queda sin cartas.

Utilizando los TADs Baraja y Bicola, **(no hay que implementar ninguna operación de los TAD's) se pide:**

a) Implementar las estructuras de datos necesarias para representar a los jugadores y la mesa con el estado del juego.

b) *Diseñar y escribir* un subprograma que simule una partida diciendo al final cuál ha sido el jugador ganador.

Baraja:

```
public class Carta {

   String palo;  // oros, copas, espadas, espadas
   int valor;    // valor entre 1 y 10
}

public class Baraja {

   prívate Carta[] cartas;

   public Baraja(); // constructora
   // postcondición: la baraja contiene 40 cartas en orden aleatorio

   public Iterator<Carta> iterador()
   // devuelve un iterador para recorrer las cartas

}
```

Bicola:

```
public class Bicola<T> {

   public Bicola(); // constructora
   // Inicializa la bicola (vacía)

   public boolean estaVacia();
   // Indicará si la bicola está o no vacía.

   public void insertarIzq(T elemento);
   // añade el elemento E por el extremo izquierdo de la bicola

   public void insertarDer(T elemento);
   // añade el elemento E por el extremo derecho de la bicola

   public void eliminarIzq();
   // borra el elemento del extremo izquierdo de la bicola

   public void eliminarDer();
   // borra el elemento del extremo derecho de la bicola

   public T obtenerIzq();
   // obtiene el elemento.del extremo izquierdo de la bicola

   public T obtenerDer();
   // obtiene el elemento.del extremo derecho de la bicola

}
```