Programación II
Colas con Prioridad

2C 2023 TN – Ing. Elizabeth Barrera

Temas

- *TDA*
- *Cola con prioridad*
- *Especificación*
- *Ejemplos*
- *Implementación estática*

TDA

- *Es una **abstracción**, ignoramos algunos detalles y nos concentramos en los que nos interesan.*
- *A la definición del TDA la llamamos **especificación** y a la forma de llevar a cabo lo definido lo denominamos **implementación**.*

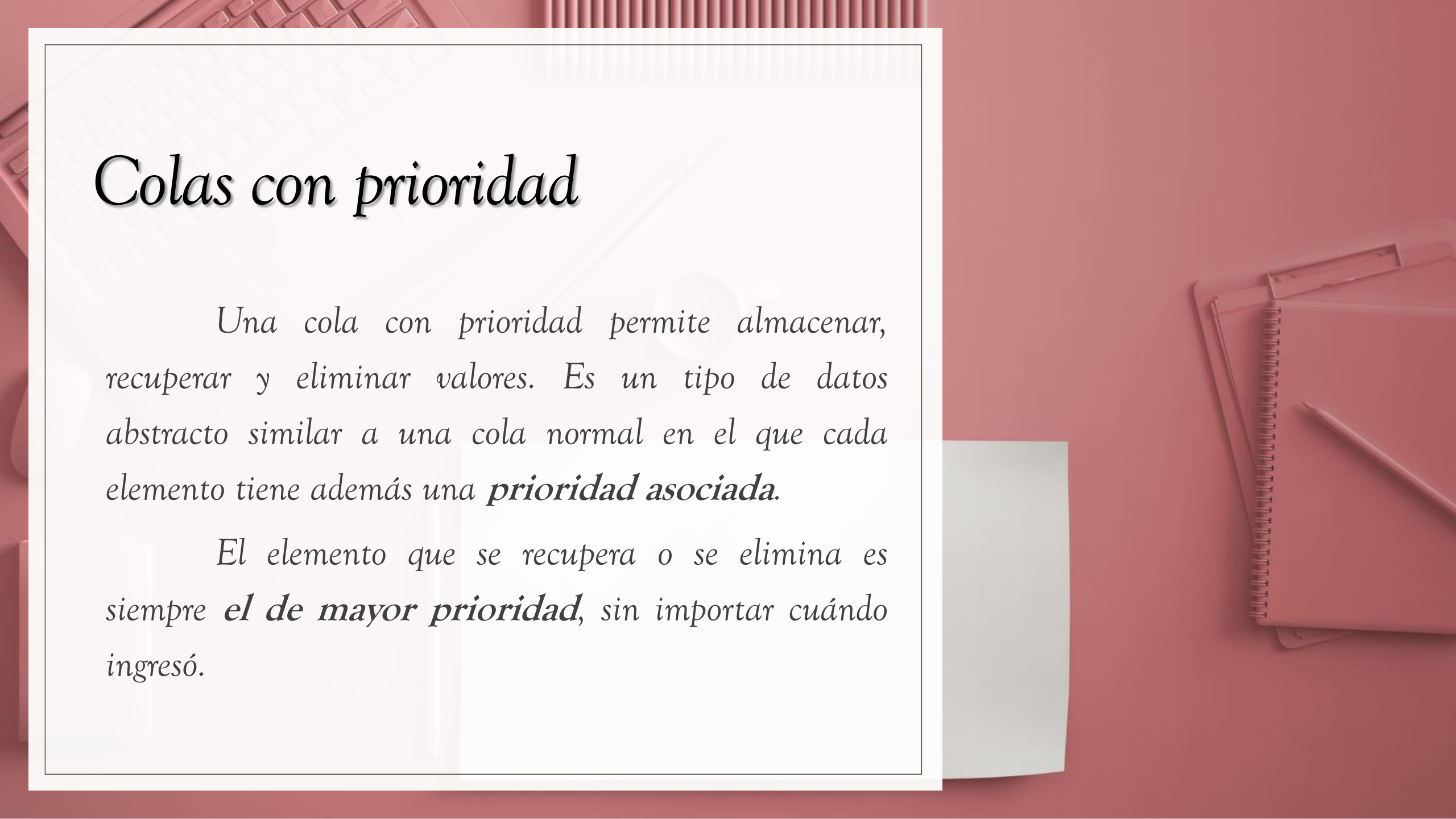
Recordar que:
*Existen siempre **2 visiones** diferentes en el TDA: usuario e implementador.*
Son separadas, y una oculta a la otra.


Colas con
Prioridad

Colas con prioridad

Una cola con prioridad permite almacenar, recuperar y eliminar valores. Es un tipo de datos abstracto similar a una cola normal en el que cada elemento tiene además una ***prioridad asociada****.*

El elemento que se recupera o se elimina es siempre ***el de mayor prioridad****, sin importar cuándo ingresó.*



Especificación

*Las operaciones que necesitaremos son: agregar y eliminar datos de la cola (que llamaremos posteriormente **acolarPrioridad** y **desacolar**), consultar el valor y la prioridad del elemento más prioritario (que llamaremos **primero** y **prioridad** respectivamente) y consultar si la cola está o no vacía (que llamaremos **colaVacia**). A estas operaciones agregaremos la inicialización de una cola con prioridad (que llamaremos **inicializarCola**).*

Especificación - Operaciones

- ***inicializarCola***: *permite inicializar la estructura de la cola.*
- ***acolarPrioridad***: *permite agregar un elemento a la cola con una cierta prioridad dada (se supone que la cola está inicializada).*
- ***desacolar***: *permite eliminar el elemento con mayor prioridad en la cola, en caso de tener dos elementos con la misma prioridad sale el primero ingresado (se supone que la cola está inicializada y no está vacía).*

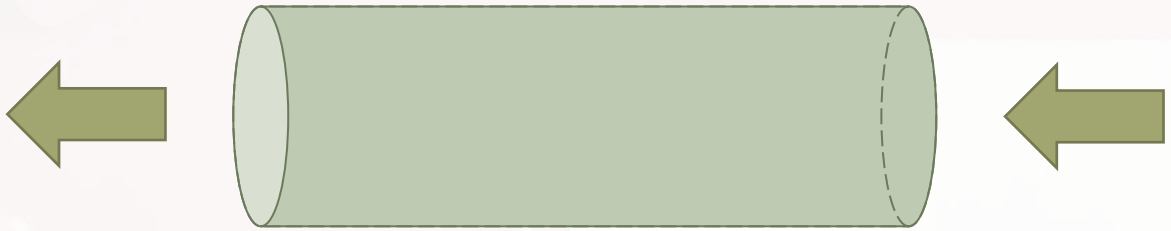
Recordar que:
*Las **precondiciones**, son condiciones que deben cumplirse antes de la ejecución de la operación.*

Especificación - Operaciones

- ***primero***: *permite conocer cuál es el elemento de mayor prioridad ingresado a la cola, en caso de tener dos elementos con la misma prioridad devuelve el primero ingresado (se supone que la cola está inicializada y no está vacía).*
- ***prioridad***: *permite conocer la prioridad del elemento con mayor prioridad de la cola (se supone que la cola está inicializada y no está vacía).*
- ***colaVacia***: *indica si la cola contiene elementos o no (se supone que la cola está inicializada).*

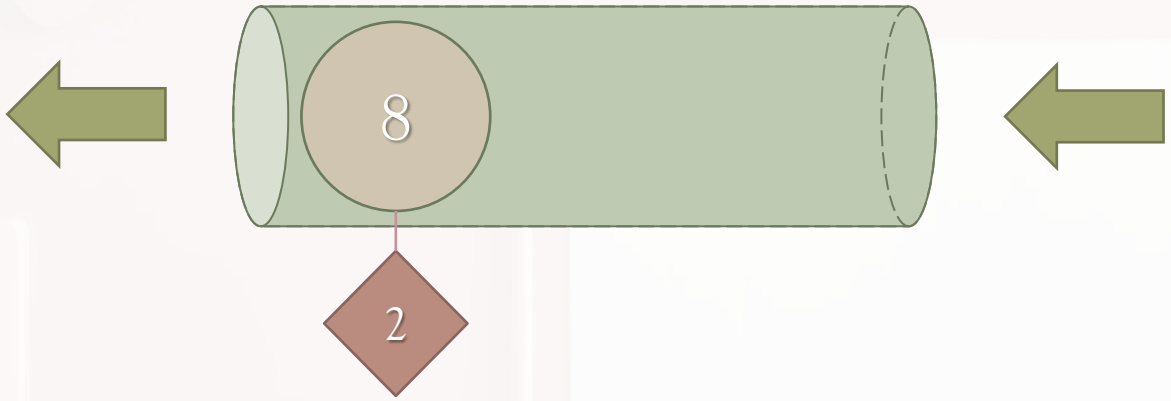
Recordar que:
*Las **precondiciones**, son condiciones que deben cumplirse antes de la ejecución de la operación.*

Colas con prioridad - Especificación



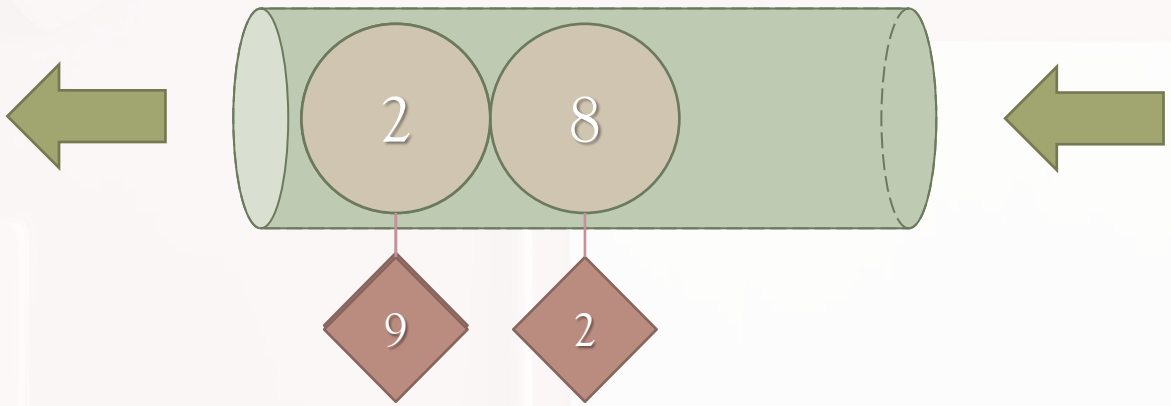
colaVacia () = true

Colas con prioridad - Especificación



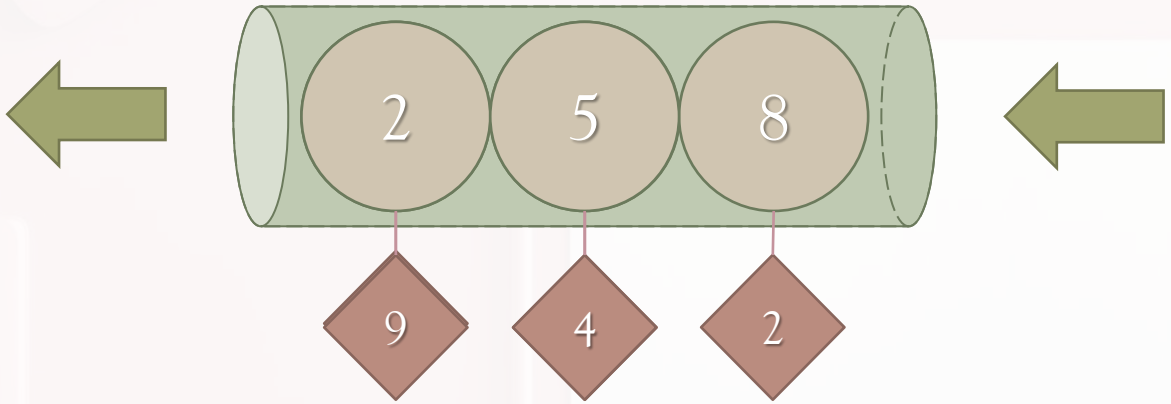
acolarPrioridad (8, 2)
colaVacia () = false
primero () = 8
prioridad () = 2

Colas con prioridad - Especificación



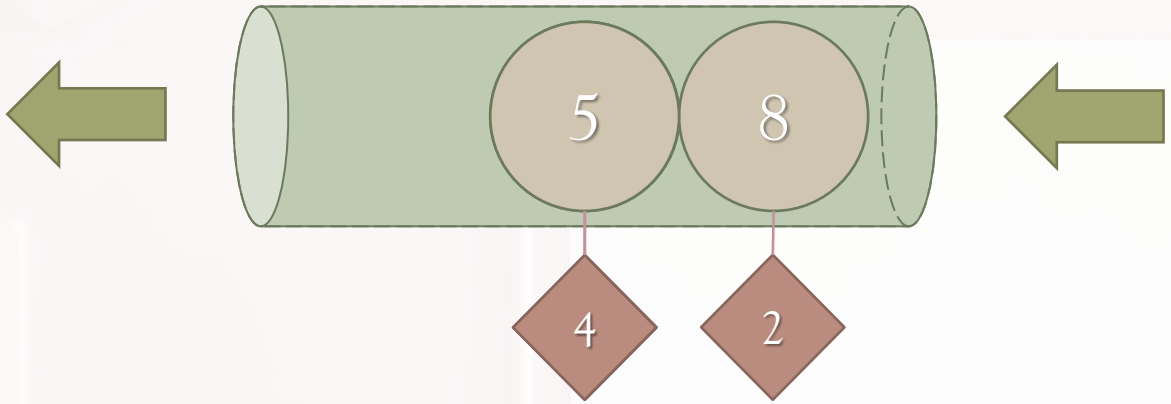
acolarPrioridad (2, 9)
colaVacia () = false
primero () = 2
prioridad () = 9

Colas con prioridad - Especificación



acolarPrioridad (5, 4)
colaVacia () = false
primero () = 2
prioridad () = 9

Colas con prioridad - Especificación



desacolar ()
colaVacia () = false
primero () = 5
prioridad () = 4

Especificación - Aclaraciones

- *En las estructuras vistas previamente los elementos se recuperaban y eliminan de acuerdo al orden en que habían ingresado a la estructura, en la cola con prioridad* ***los elementos se van ordenando según un criterio****.*
- *Si acolamos todos los elementos con la misma prioridad, la cola con prioridad se comporta como una cola común.*
- *Si acolamos todos los elementos con la prioridad igual a su valor, la cola con prioridad puede ordenar automáticamente los elementos.*

Especificación - Interfaz

```
public interface ColaPrioridadTDA {
  void inicializarCola( );
  void acolarPrioridad(int x, int
       prioridad); //cola inicializada
  void desacolar( ); //cola inicializada y no vacía
  int primero( ); //cola inicializada y no vacía
  int prioridad( ); //cola inicializada y no vacía
  boolean colaVacia( ); //cola inicializada
}
```

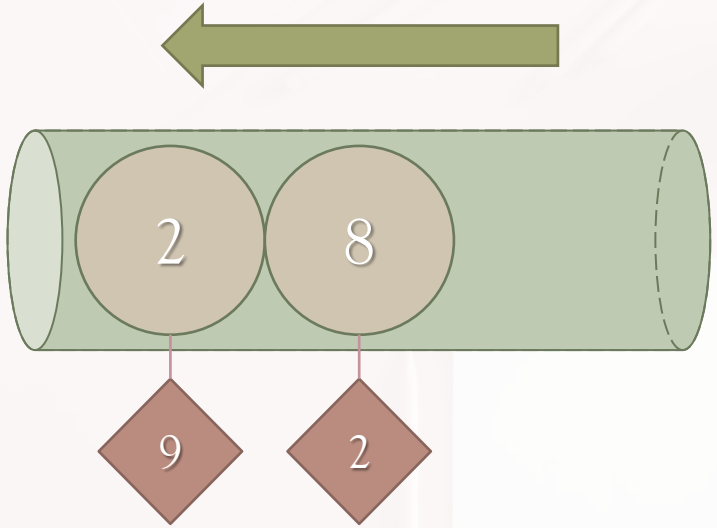
Uso - Ejemplos

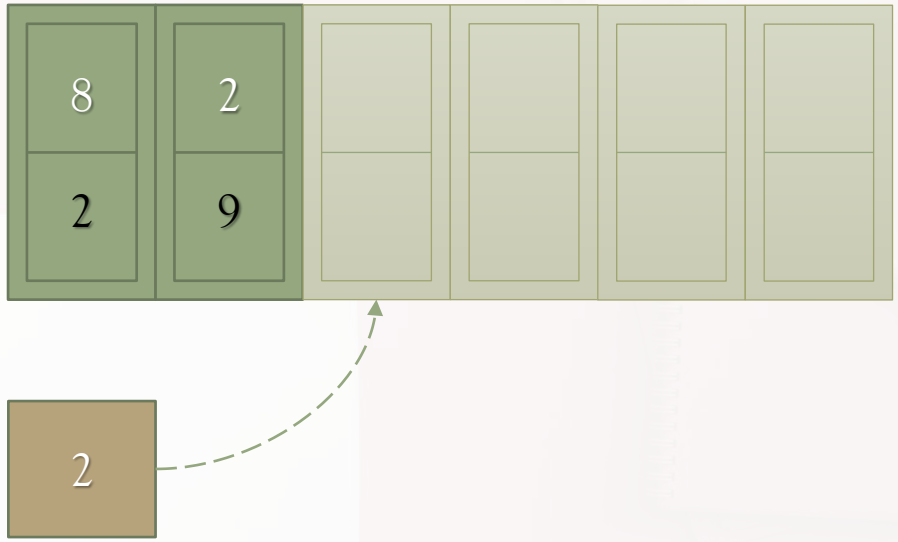
- *Vamos a escribir un método que nos permita pasar los valores de una cola con prioridad Origen a una cola normal Valores, y las prioridades correspondientes a una cola normal Prioridades.*

Implementación estática Estrategia 1

- Se utilizará un **arreglo** de una **estructura** (una clase en Java) que contiene el valor y la prioridad asociada a ese valor.
- Una variable entera indicará la primera posición libre.
- Cuando un elemento **ingresa** a la cola, deben reacomodarse los elementos según el orden de las prioridades. Al nuevo elemento se lo debe acomodar en la posición que corresponde a su prioridad, desplazando hacia la derecha los de mayor prioridad.
- Los elementos con la **máxima prioridad** quedarán en el máximo índice ocupado. Por lo tanto, para **desacolar** bastará con decrementar la variable entera.

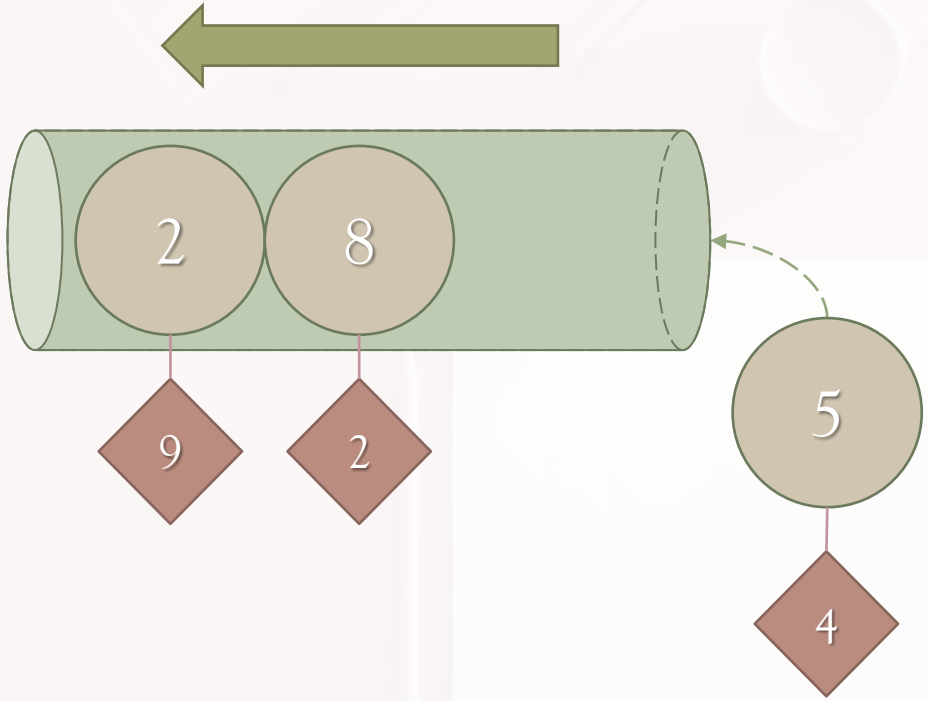
Colas con prioridad - Implementación estática

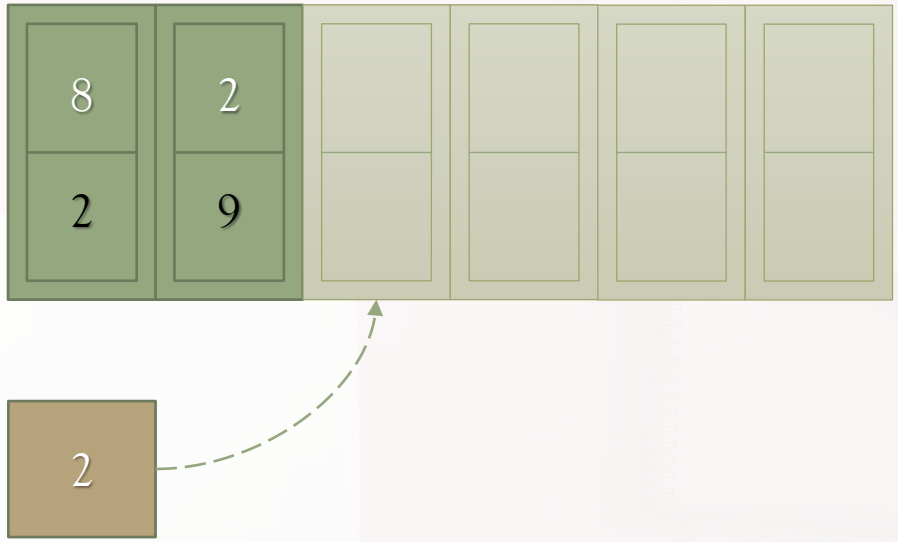




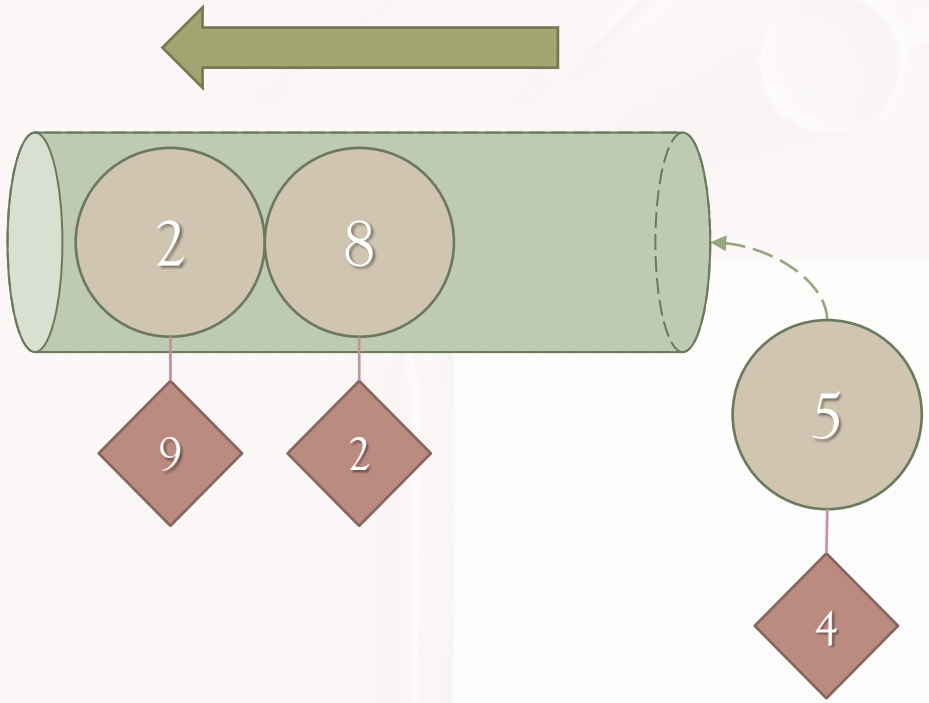
Colas con prioridad - Implementación estática

acolarPrioridad (5, 4)

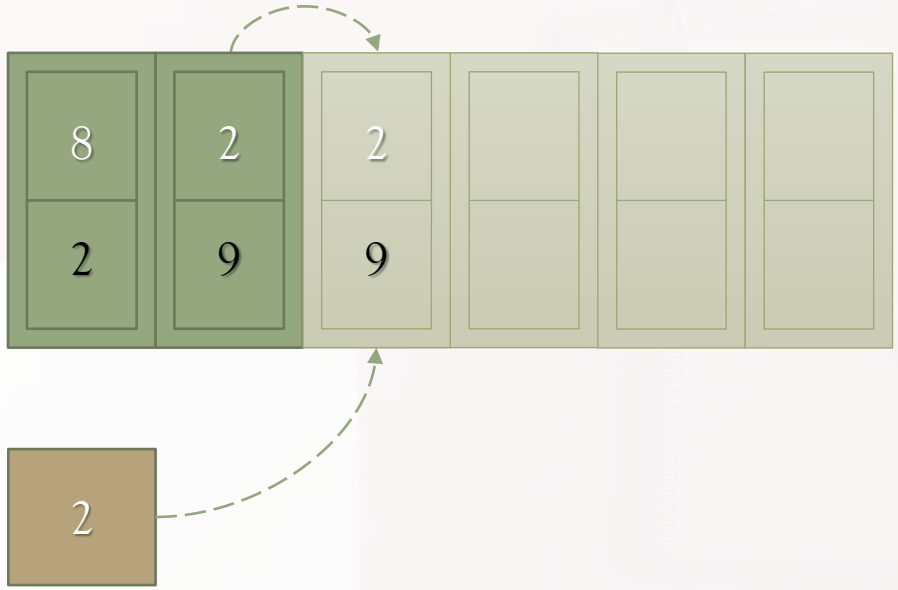




Colas con prioridad - Implementación estática

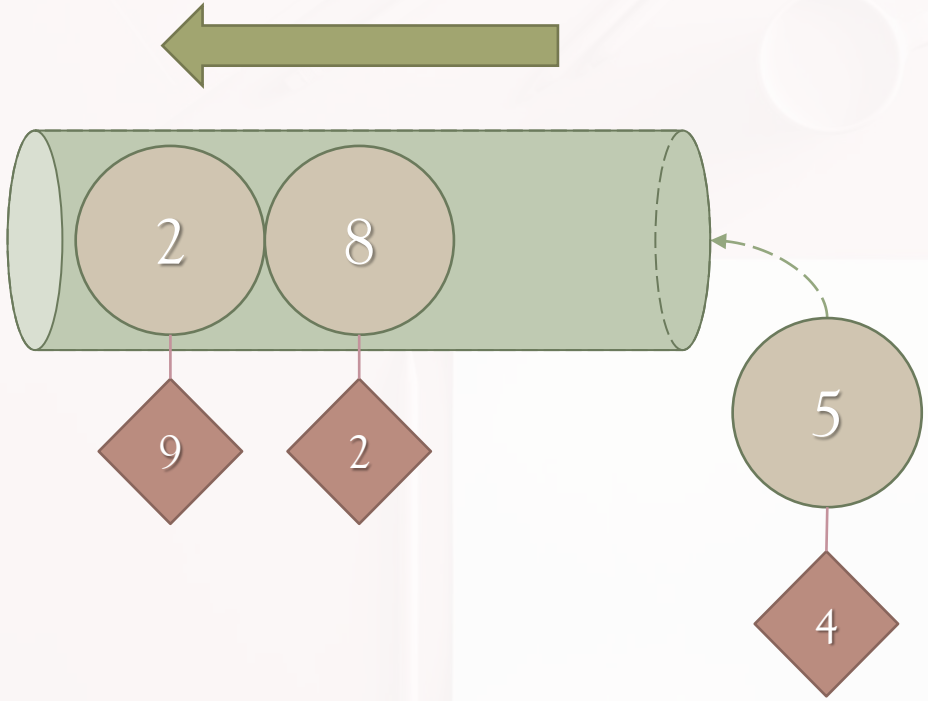


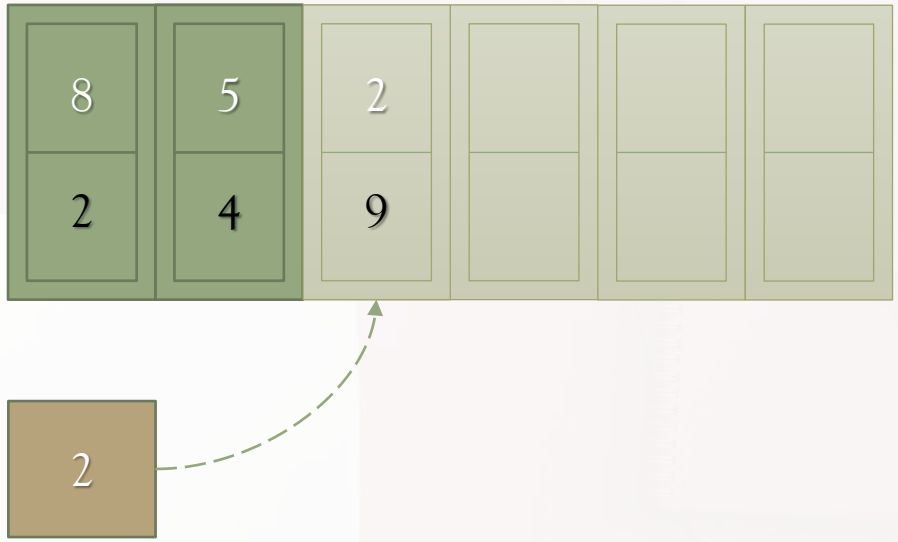
acolarPrioridad (5, 4)



Colas con prioridad - Implementación estática

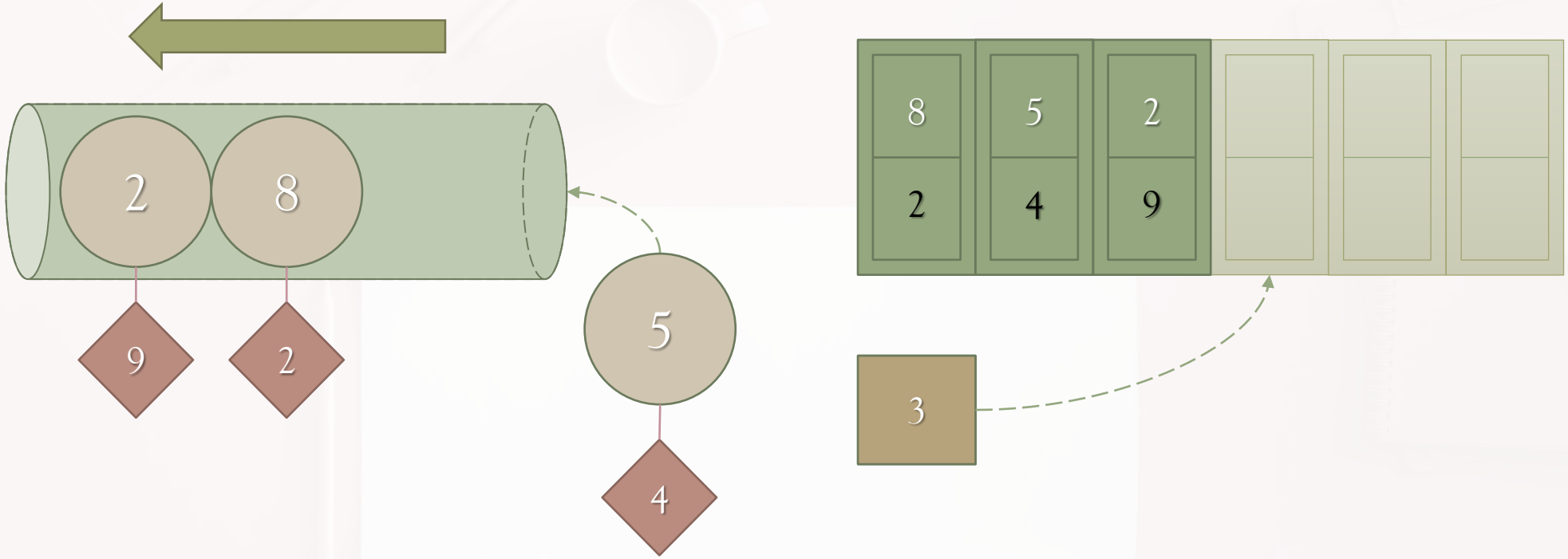
acolarPrioridad (5, 4)



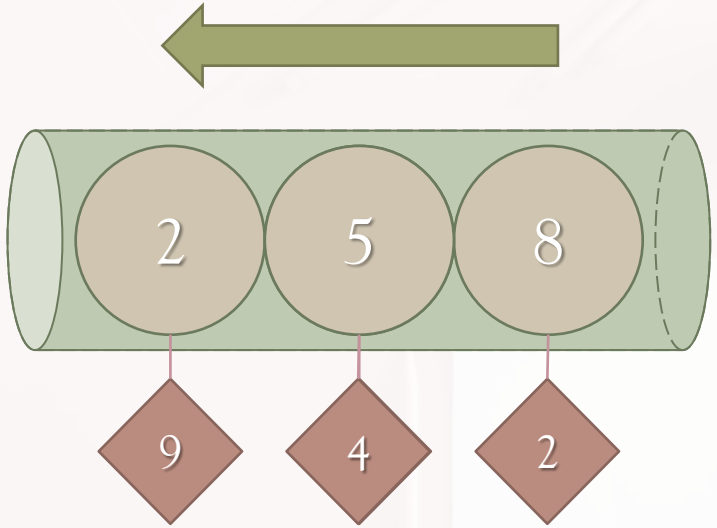


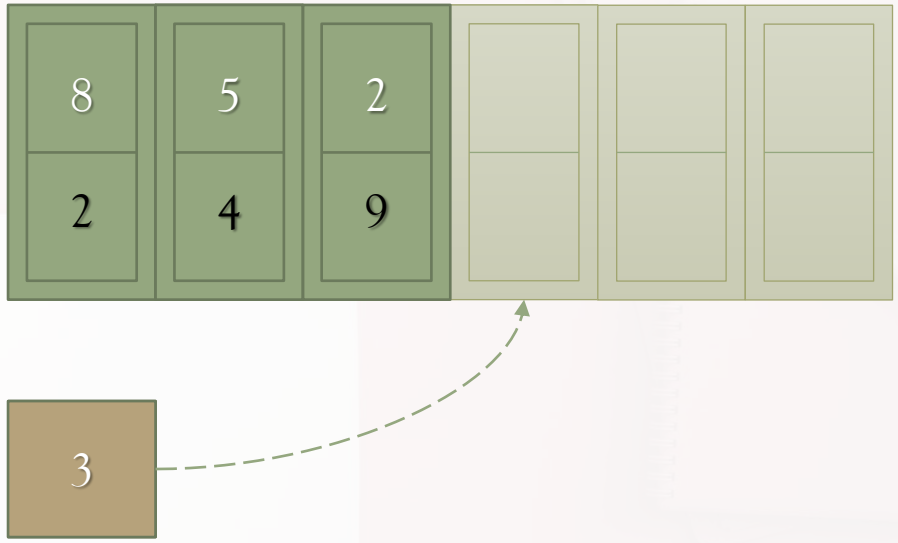
Colas con prioridad - Implementación estática

acolarPrioridad (5, 4)

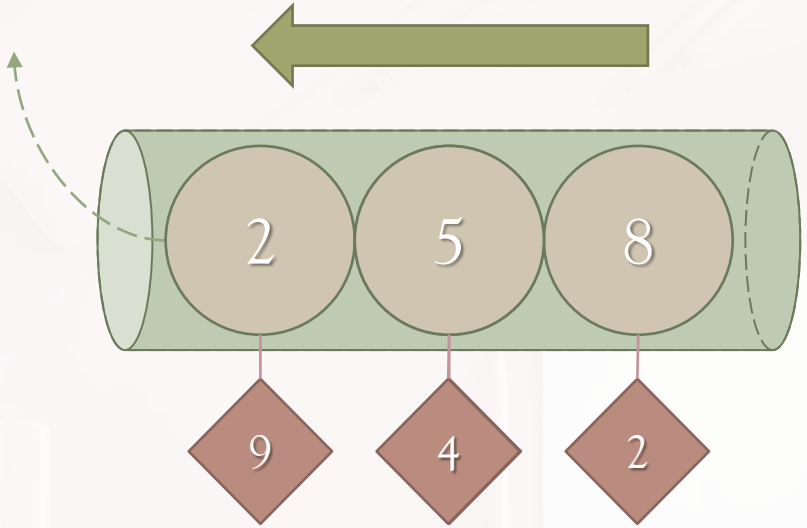


Colas con prioridad - Implementación estática

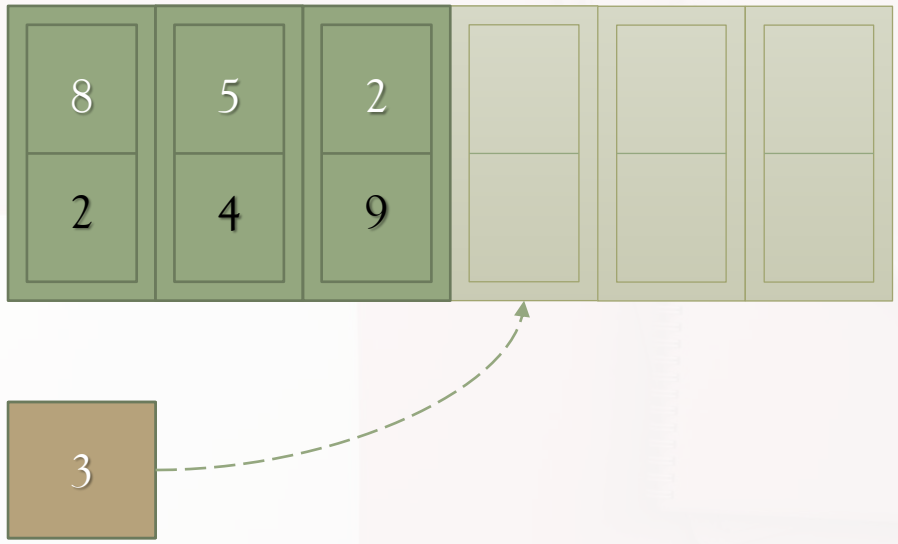




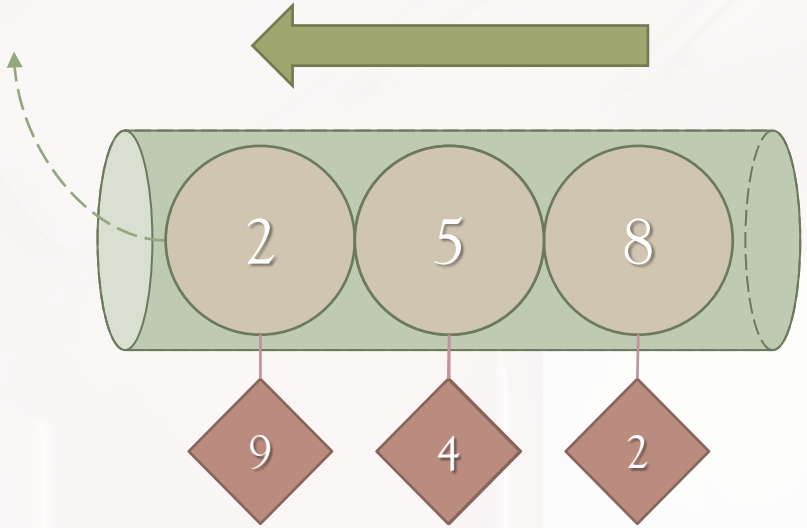
Colas con prioridad - Implementación estática



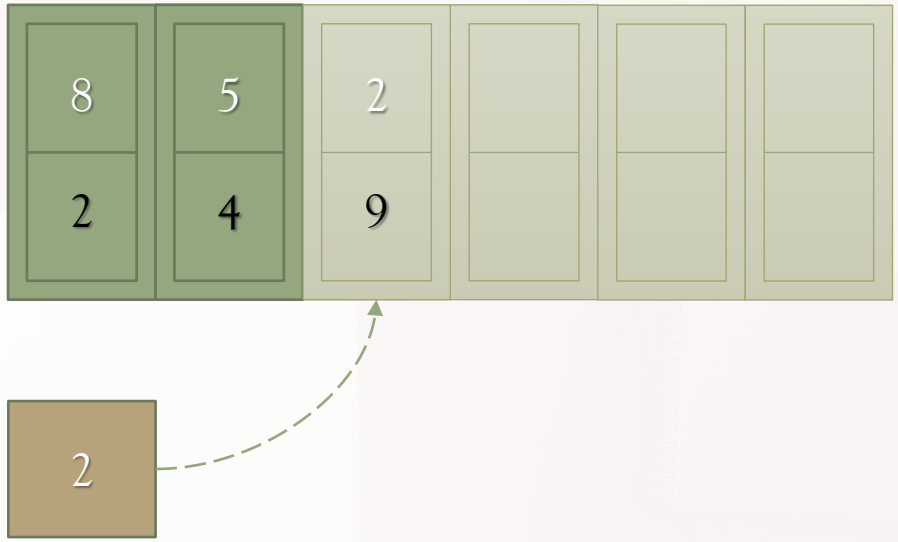
desacolar ( )



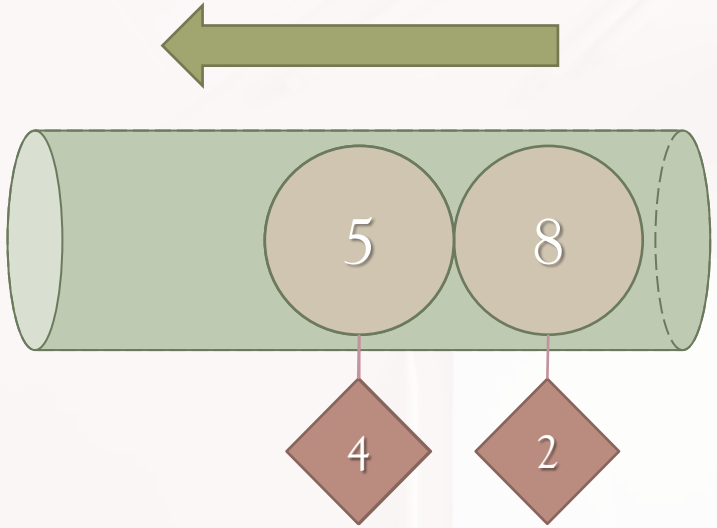
Colas con prioridad - Implementación estática

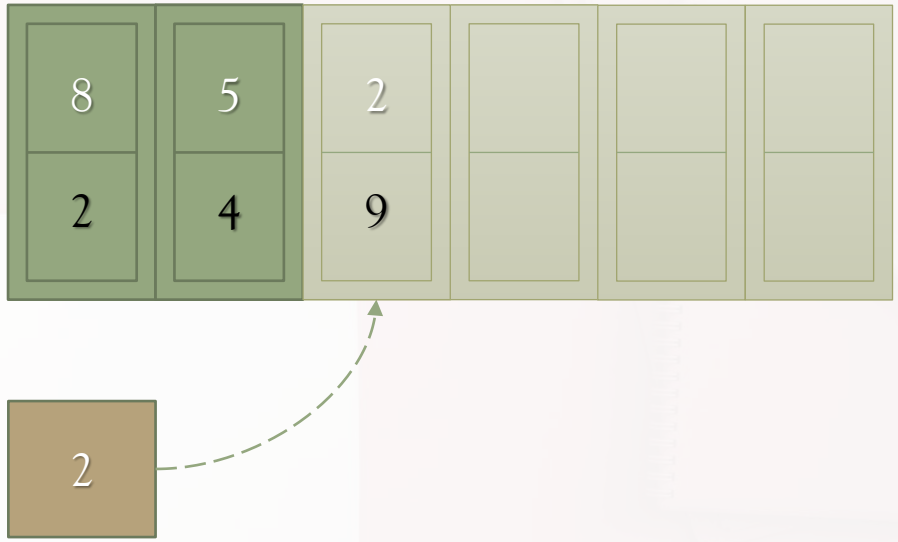


desacolar ( )



Colas con prioridad - Implementación estática





Implementación estática Aclaraciones

- Se cuenta con un objeto de tipo Elemento que tiene los parámetros Elemento.valor y Elemento.prioridad.
- La **eliminación** de un elemento del vector arr se representa dejándolo afuera de la parte del arreglo delimitada por la variable indice; a los efectos prácticos, cualquier elemento arr[i] situado en una posición i ≥ indice **no existe más en la cola**.
- Tanto el vector arr, como el entero índice **no son accesibles desde afuera de la implementación** (son privados).


¡Muchas Gracias!

Bibliografía

- *Programación II – Apuntes de Cátedra – V1.3 – Cuadrado Trutner – UADE*
- *Programación II – Apuntes de Cátedra – Wehbe – UADE*