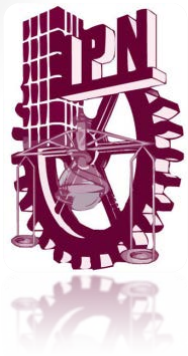


Instituto Politécnico Nacional

Escuela Superior de Cómputo

Estructuras de Datos

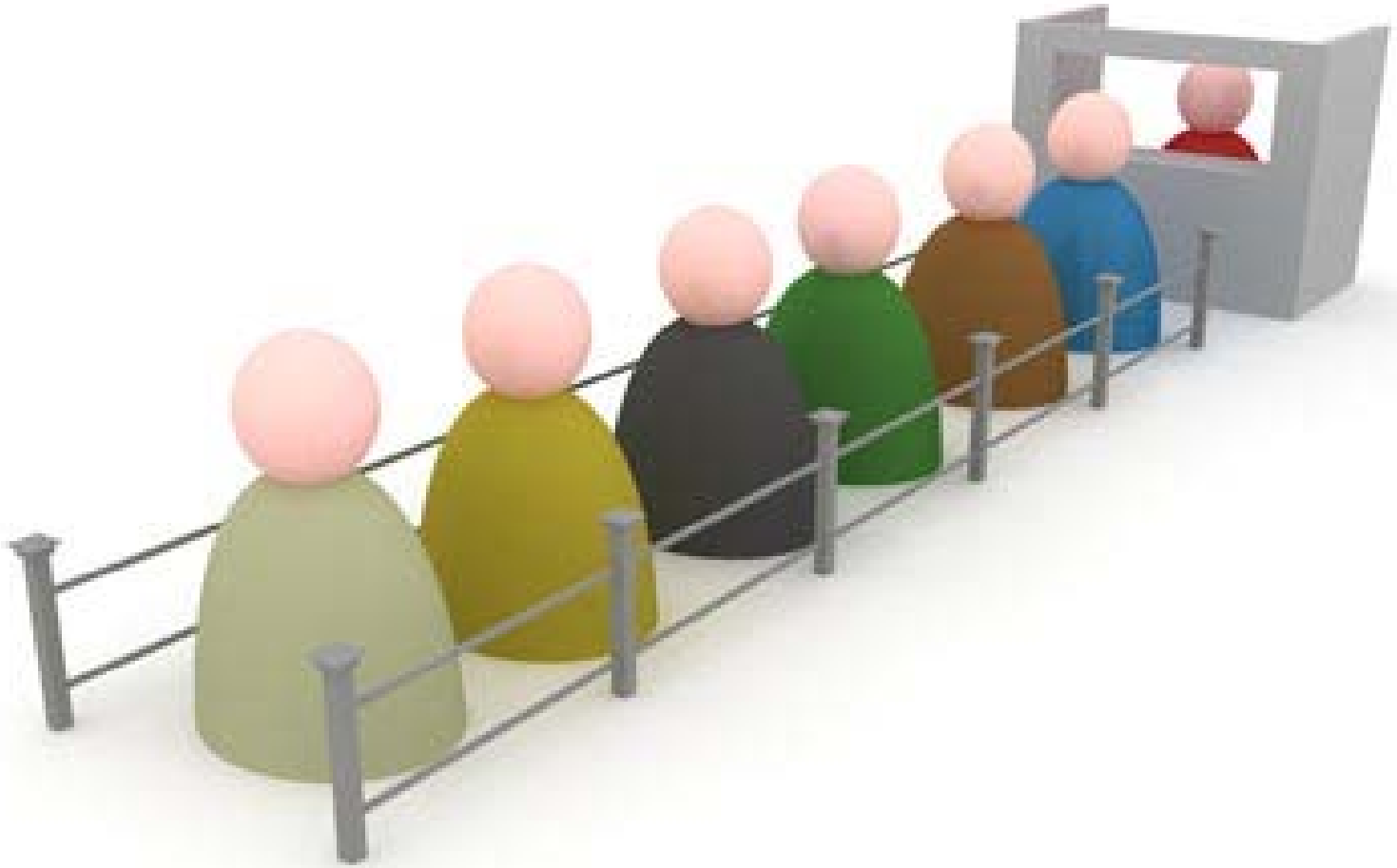
Tema 03: TAD Cola

M. en C. Edgardo Adrián Franco Martínez
http://www.eafranco.com
edfrancom@ipn.mx
@edfrancom *edgardoadrianfrancom*

Contenido

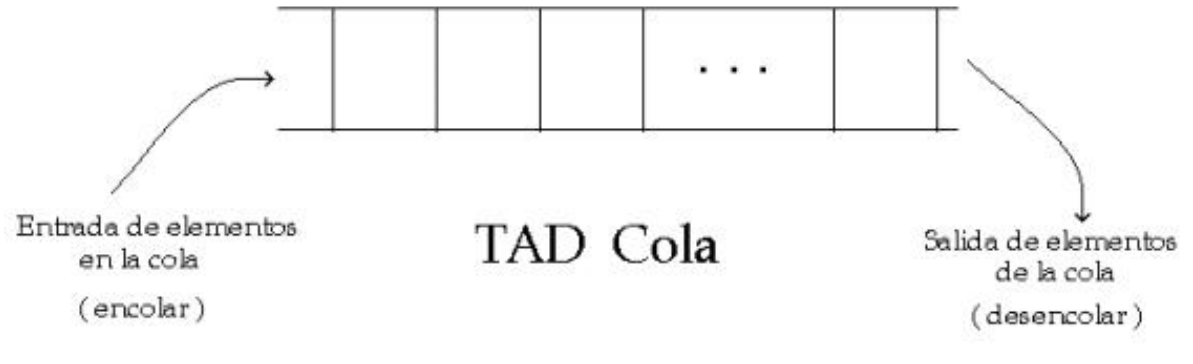
- Descripción del TAD Cola
- Especificación del TAD Cola
  - Cabecera
  - Definición
  - Operaciones
  - Observaciones
- Implementación del TAD Cola
  - Implementación estática
    - Cola circular
  - Implementación dinámica
- Aplicaciones del TAD Cola
- Cola de prioridades
  - Gestión de procesos del sistema operativo

Descripción del TAD Cola



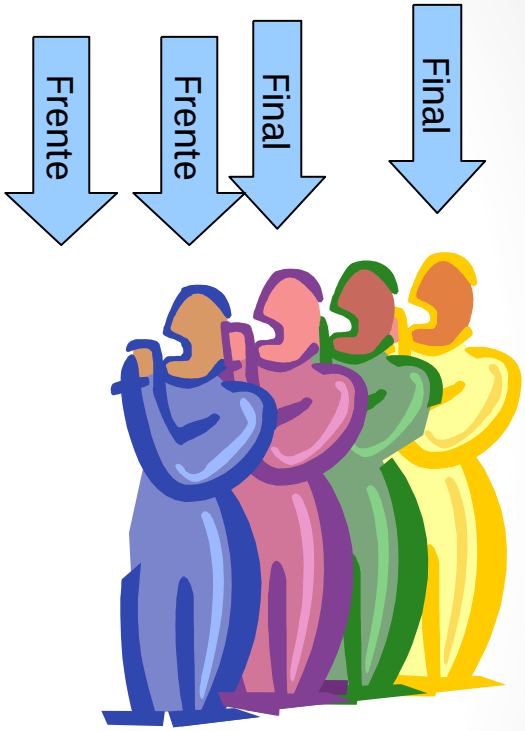
Descripción del TAD Cola

- Uno de los conceptos más abundantes en la vida cotidiana es la cola.
- Una COLA es otro tipo especial de Lista en el cual:
  - Los elementos se insertan en un extremo (el posterior) y la supresiones tienen lugar en el otro extremo (anterior o frente).
  - A las Colas se les llama también **Listas FIFO *(first in first out)*** o Listas (primero en entrar, primero en salir).



Descripción del TAD Cola (Ejemplo)

- Abunda este concepto, en la vida cotidiana
  - Cuando vamos al cine, para comprar las entradas
  - Cuando estamos en el supermercado, en el banco, etc.



- **Como funciona**
  - Se puede decir que la cola tiene 2 extremos
    - FRENTE Y FINAL
  - Todo el que llega se ubica al final de la cola
  - La cola es por turno
    - El primero en llegar, tiene la seguridad de que será el primero en salir:
      - FIRST IN FIRST OUT -> FIFO

Especificación del TAD Cola

Cabecera

- ***Nombre:*** COLA *(QUEUE)*
- ***Lista de operaciones:***
  - ***Inicializar cola (Initialize):*** Recibe una cola y la inicializa para su trabajo normal.
  - ***Encolar (Queue):*** Recibe una cola y un elemento y agrega el elemento al final de ella.
  - ***Desencolar (Dequeue):*** Recibe una cola y remueve el elemento del frente retornándolo.
  - ***Es vacía (Empty):*** Recibe la cola y devuelve verdadero si esta esta vacía.
  - ***Frente (Front):*** Recibe una cola y retorna el elemento del frente.
  - ***Final (Final):*** Recibe una cola y retorna el elemento del final.
  - ***Elemento (Element*)**: Recibe una cola y un número de elemento de 1 al tamaño de la cola y retorna el elemento de esa posición.
  - ***Eliminar cola (Destroy):*** Recibe una cola y la libera completamente.
  - ***Tamaño (Size):*** *Recibe una cola y devuelve el tamaño de esta*

Definición

- En un TDA Cola (*Queue*), podemos realizar al menos dos operaciones básicas:
  - **Encolar (Queue):** Insertar un elemento nuevo a la cola, al final de la cola
    - El final aumenta
  - **Desencolar (Dequeue):** Remover un elemento del frente de la cola
    - Remueve el elemento del **frente**
    - Retorna el elemento removido
    - No se puede ejecutar si la cola **esta vacía.**

- *Así como en la pila*
  - Cualquier intento de acceder a elementos en una Cola Vacía causa:
    - Subdesbordamiento de la cola
  - Cualquier intento de introducir a elementos en una Cola llena causa:
    - Desbordamiento de la cola

Operaciones

- ***Inicializar cola (Initialize):*** ***recibe<- cola (C);***
  - ***Initialize (C)***
  - Efecto: Recibe una cola, la inicializa para su trabajo normal.
- ***Encolar (Queue):*** ***recibe<- cola (C) ; recibe<- elemento(e)***
  - ***Queue(C,e);***
  - Efecto: Recibe una cola y un elemento el cuál se introduce al final de la cola.
- ***Desencolar (Dequeue):*** ***recibe<- cola (C); retorna -> elemento***
  - ***e=Dequeue (C);***
  - Efecto: Recibe una cola y devuelve el elemento que se encuentra al frente de esta, quitándolo de la cola.
  - Excepción: Si la cola esta vacía produce **error.**
- ***Es vacía (Empty):*** ***recibe<- cola (C); retorna -> booleano***
  - ***Empty (C)***
  - Efecto: Recibe la cola y verifica si esta tiene elementos, devuelve **TRUE** si esta vacía y **FALSE** en caso contrario.
- ***Frente (Front):*** ***recibe<- cola (C) ; retorna -> elemento***
  - ***e=Front (C);***
  - Efecto: Recibe una cola y devuelve el elemento que se encuentra al frente de esta.
  - Excepción: Si la cola esta vacía produce **error.**

- ***Final (Final):*** ***recibe<- cola (C); retorna -> elemento***
  - ***e=Final(C);***
  - Efecto: Recibe una cola y devuelve el elemento que se encuentra al final de esta.
  - Excepción: Si la cola esta vacía produce **error.**
- ***Elemento(Element):*** ***recibe<- cola (C); recibe<- índice(n); retorna -> elemento***
  - ***e=Element (C,n);***
  - Efecto: Recibe una cola y un índice (entre 1 y el tamaño de la cola) y devuelve el elemento que se encuentra en la cola en ese índice partiendo del frente de esta =1.
  - Excepción: Si la cola esta vacía o el índice se encuentra fuera del tamaño de la cola se produce **error.**
- ***Eliminar cola (Destroy):*** ***recibe<- cola (C);***
  - ***Destroy(C)***
  - Efecto: Recibe una cola y la libera completamente.
- ***Tamaño (Size):*** ***recibe<- cola (C); retorna -> numero***
  - ***n=Size (C);***
  - Efecto: Recibe una cola y devuelve el tamaño de la cola.

Observaciones

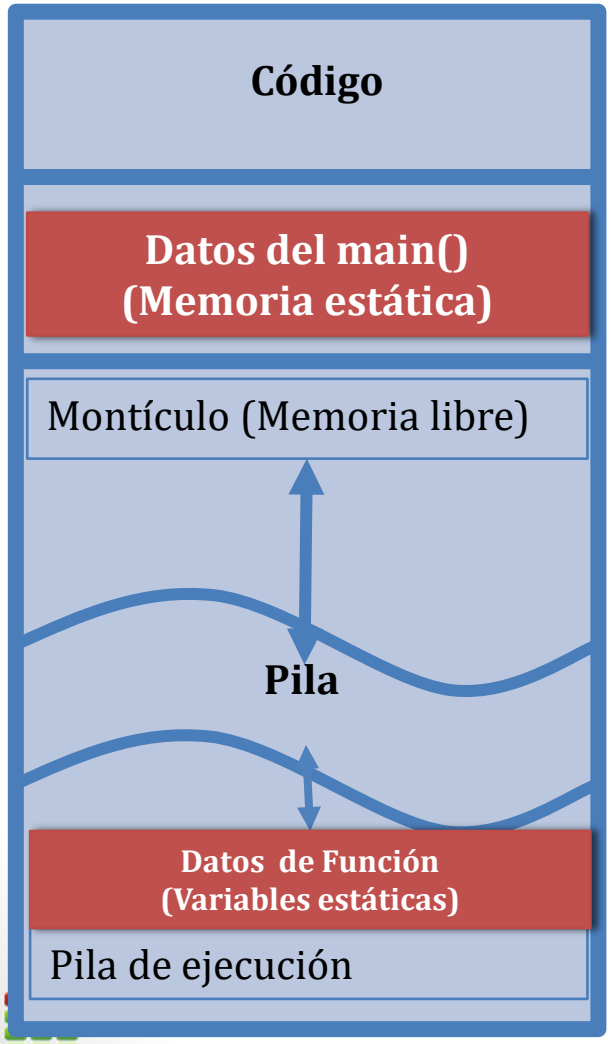
- Al remover el ultimo elemento de una cola esta queda vacía
  - Una vez vacía, no se pueden “ver o desencolar” elementos de la cola.
- Antes de sacar un elemento de la cola
  - Debemos saber si la cola **Esta Vacía**
- Al tratar de desencolar o acceder a elementos de una cola vacía se le llama:
  - **Subdesbordamiento** de la cola
- Al tratar de encolar a elementos de una cola que ha llegado a su MAXIMO_TAMAÑO *(Se supone que no deberá de tener fin, aunque en la realidad no es posible)* se le llama:
  - **Desbordamiento** de la cola

Implementación del TAD Cola

- Hay varias formas…
  - La Cola es una Lista… pero limitada
    - En la lista los nuevos nodos se pueden insertar y remover
      - Al/Del Inicio, al final, dada una posición, etc.
    - En la Cola los elementos solo se pueden **insertar al final** y solo se pueden **remover del frente** de la Cola.

- Más adelante en el curso se notará que implementaciones de la **lista simplemente enlazada** podrían usarse para implementar una Cola estática o dinámica.

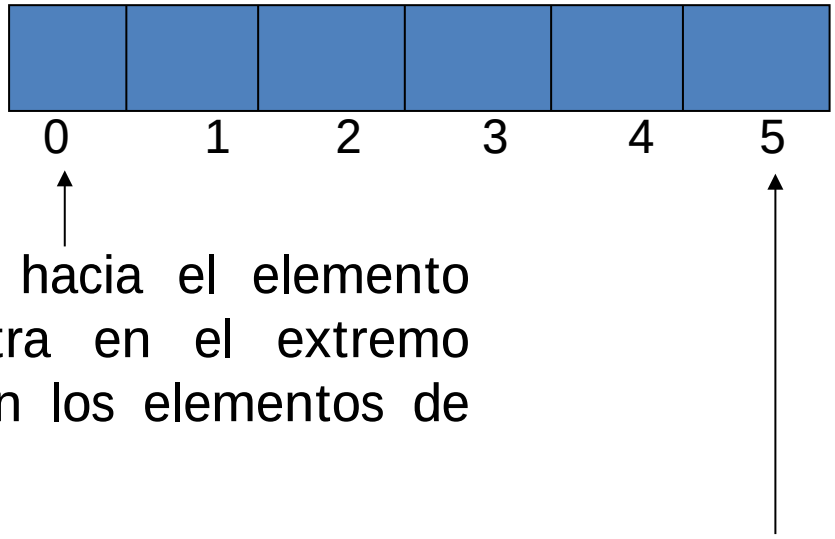
Implementación estática



- Hablamos de una **Estructura de datos estática** cuando se le asigna una cantidad fija de memoria para esa estructura antes de la ejecución del programa.
  - La cantidad de espacio asignado para la memoria estática **se determina durante la compilación y no varía a la hora de ejecutarse el programa que la implemente**.

Implementación estática del TAD Cola

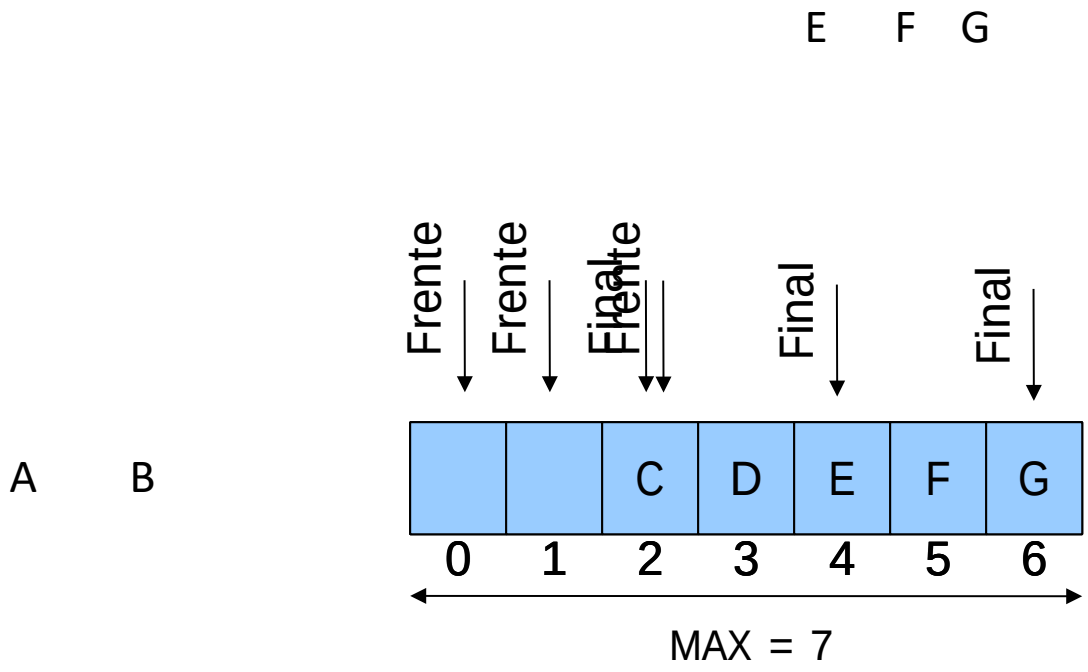
- **Usando un arreglo:** Definir un arreglo de una dimensión (vector) donde se almacenan los elementos.



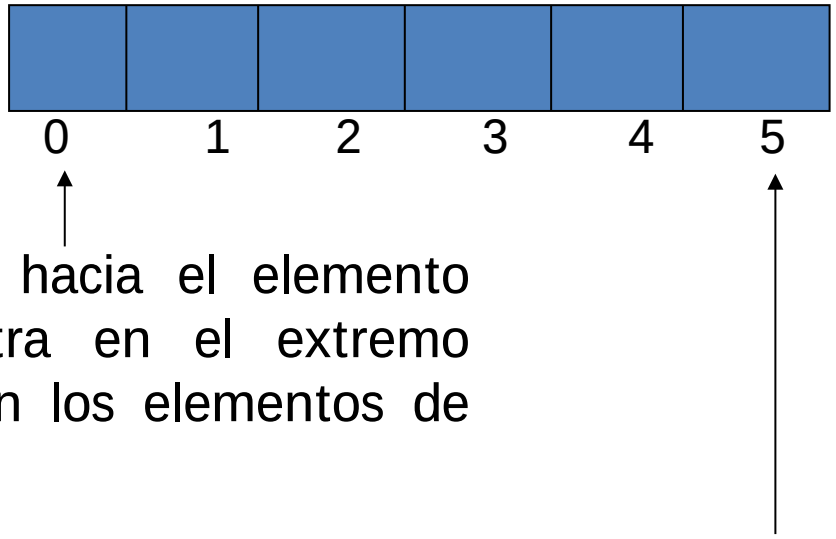
Frente: Apunta hacia el elemento que se encuentra en el extremo donde se extraen los elementos de la cola

Final: Apunta hacia el elemento que se encuentra en el extremo donde se introducen los elementos a la cola.

La implementación estática se puede modelar con un arreglo estático de tamaño máximo MAX, donde se jugara con las variables que indican el Frente y Final de la cola



- **Desencolar usando un arreglo:** Existen dos posibilidades al utilizar un arreglo estático:
  1. Al desencolar un elemento, recorrer al resto de los elementos hacia la posición 0 del arreglo, para poder dejar los espacios al final.
  2. Al desencolar la posición del frente cambiará y deberá de existir un control de los espacios para aprovechar el arreglo completamente.

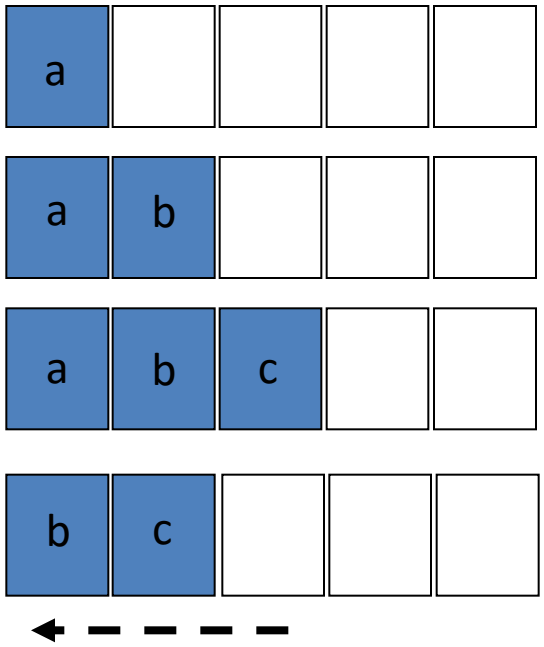


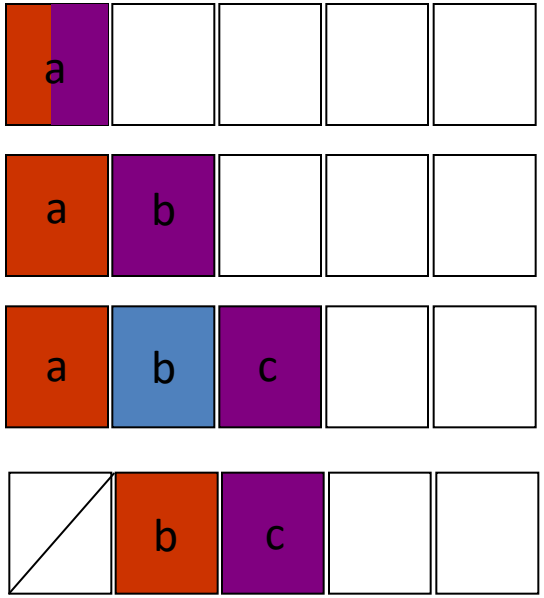
Frente: Apunta hacia el elemento que se encuentra en el extremo donde se extraen los elementos de la cola

Final: Apunta hacia el elemento que se encuentra en el extremo donde se introducen los elementos a la cola.

Cola Circular

- Al implementar el TAD Cola de manera estática mediante arreglos se puede notar que:

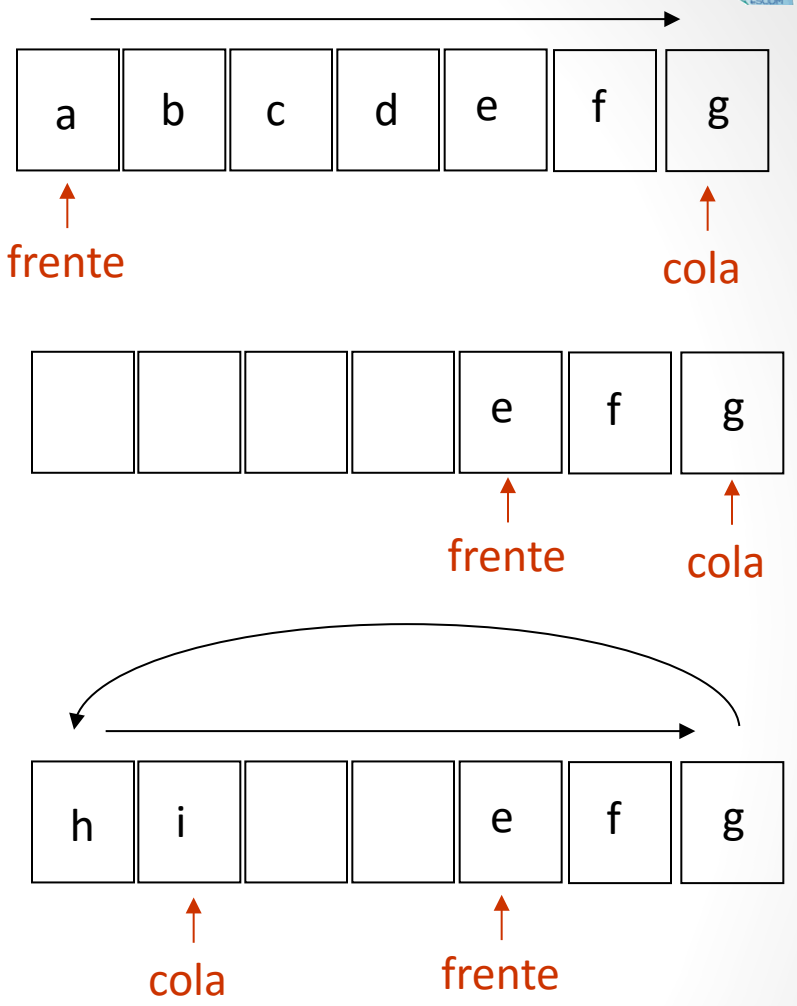




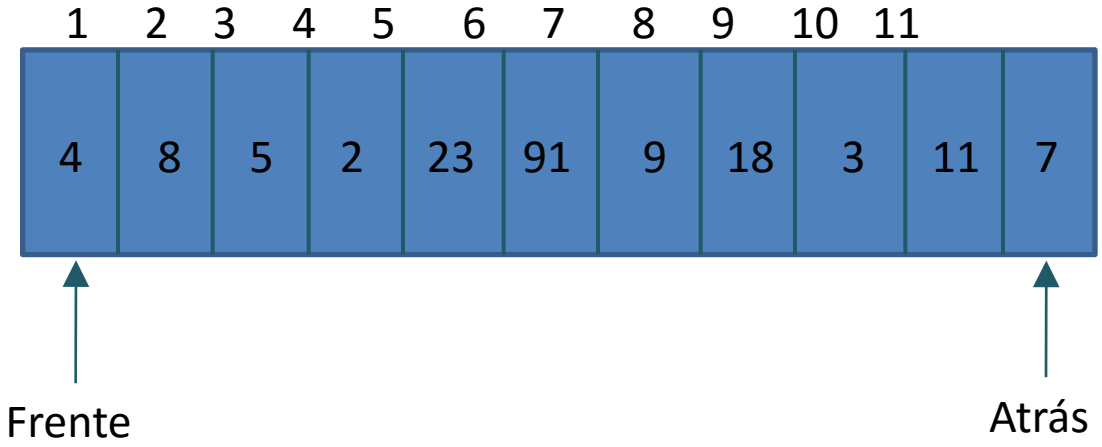
No es necesario mover todos los elementos

Basta controlar adecuadamente a los apuntadores al frente y al final de la cola.

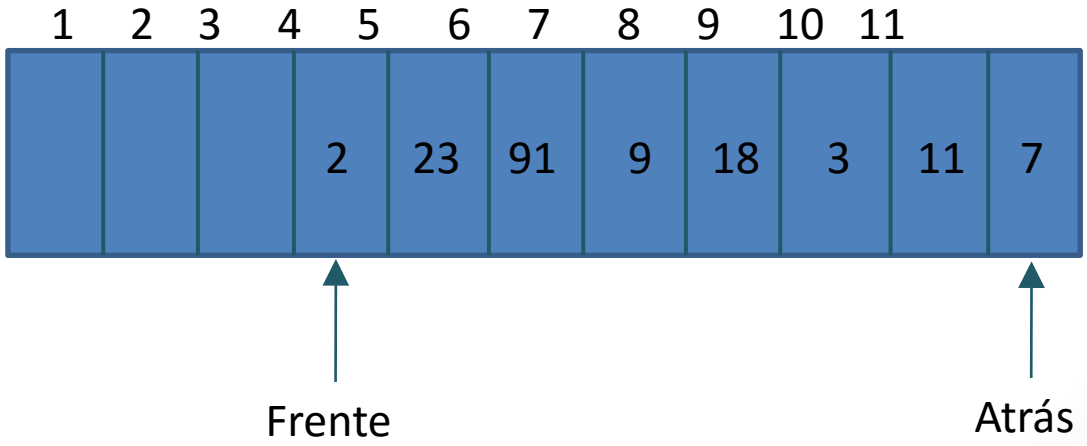
- El objetivo de una cola circular es aprovechar al máximo el espacio del arreglo.
- La idea es insertar elementos en las localidades previamente desocupadas.
- La implementación tradicional considera dejar un espacio entre el frente y la cola.



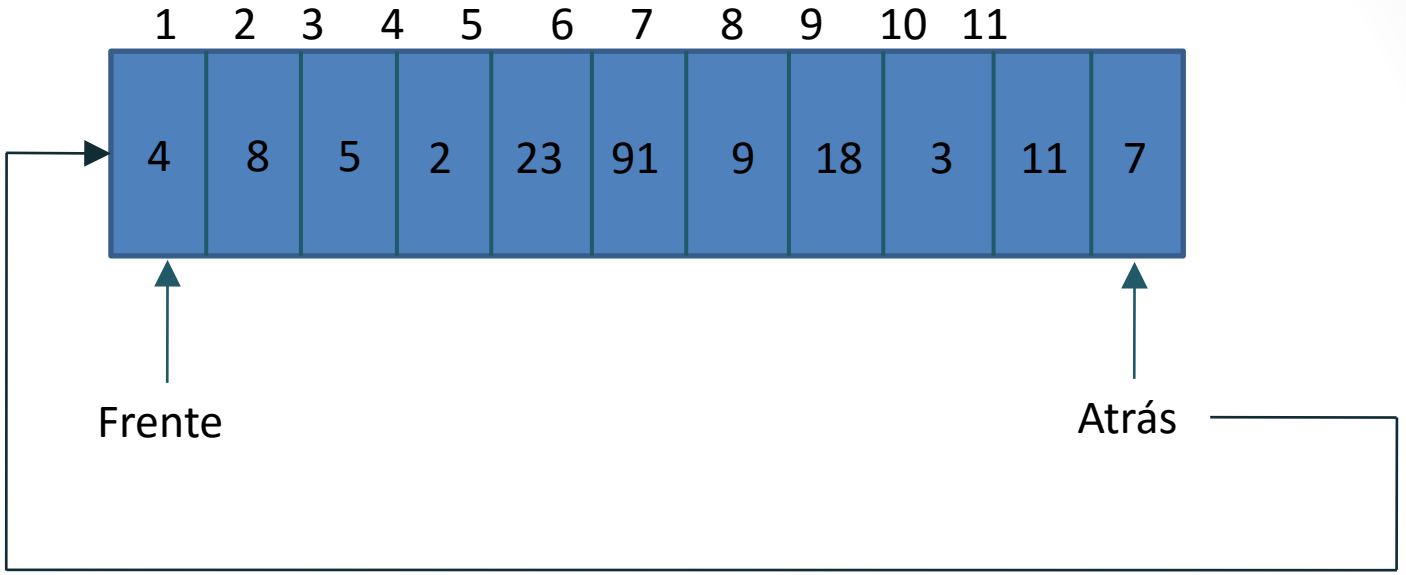
- Suponga que se tiene la siguiente cola:



- Donde se tiene que la cola esta llena, una vez que se realizan varias operaciones de desencolar ***Dequeue(C)***, quedaría:

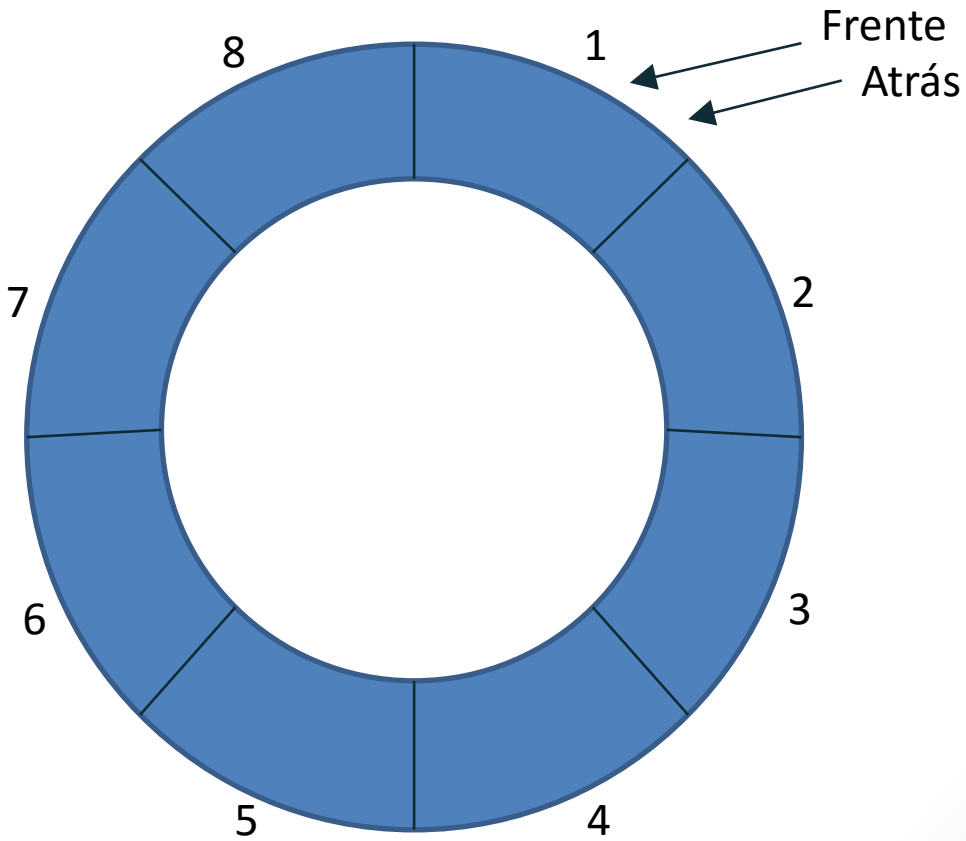


- En este caso, lo que se tiene es que al llegar ***atrás*** al final de la cola, se reinicie el mismo.



Cola Circular: Representación

- Por lo que se puede representar:

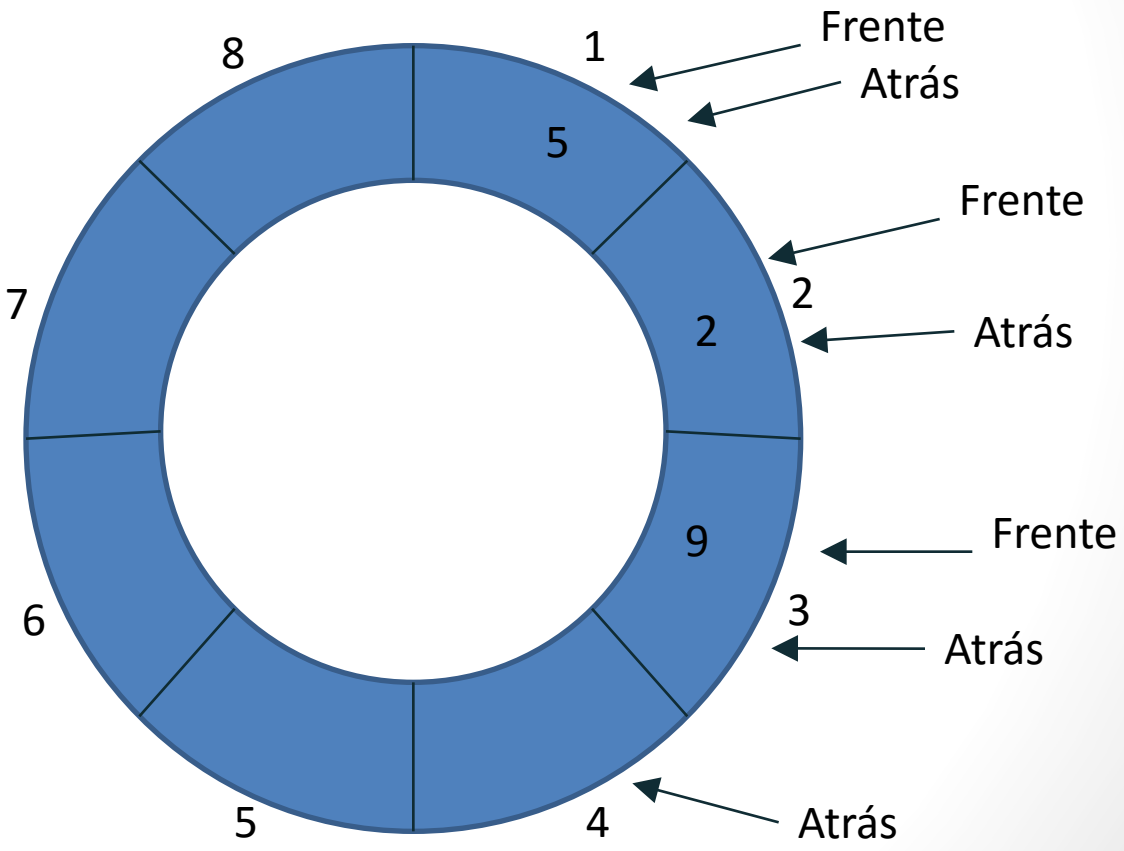


Cola Circular: Operaciones

- Se tienen las mismas operaciones que en una cola lineal.

Encolar (Queue)

Desencolar (Dequeue)



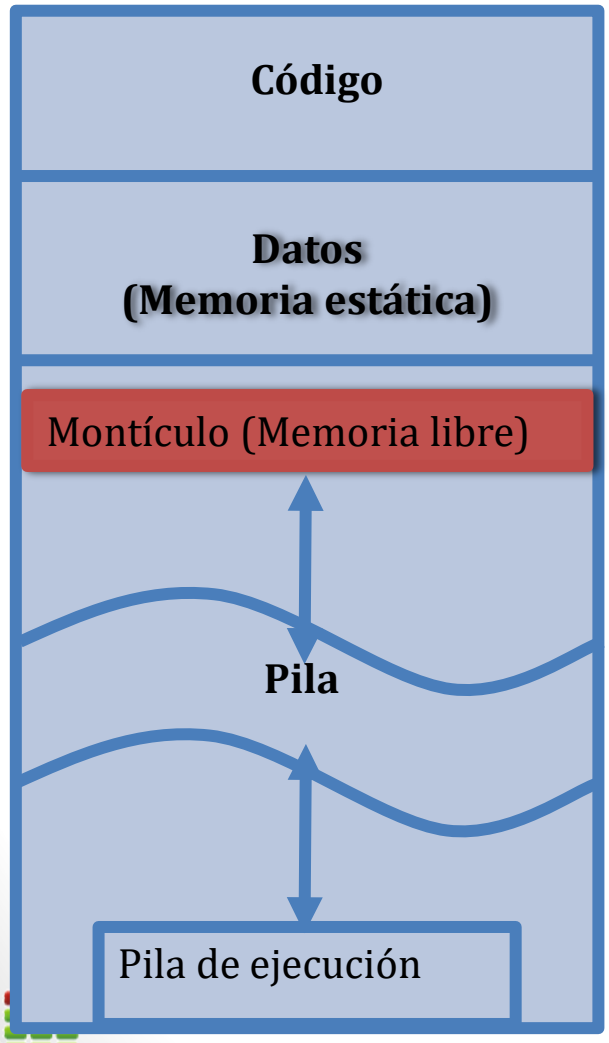
Cola Circular: Operaciones

- Ahora, para realizar las operaciones de ***Encolar(Queue)*** y ***Desencolar (Dequeue)***, es necesario saber cual fue la última operación, debido a que si: *Atrás* = *Frente*, entonces la cola o bien puede estar vacía o bien puede estar llena.

- Con la cual si:
  - *Atrás* = *Frente*, y la última operación fue eliminar entonces la cola esta **vacía**.

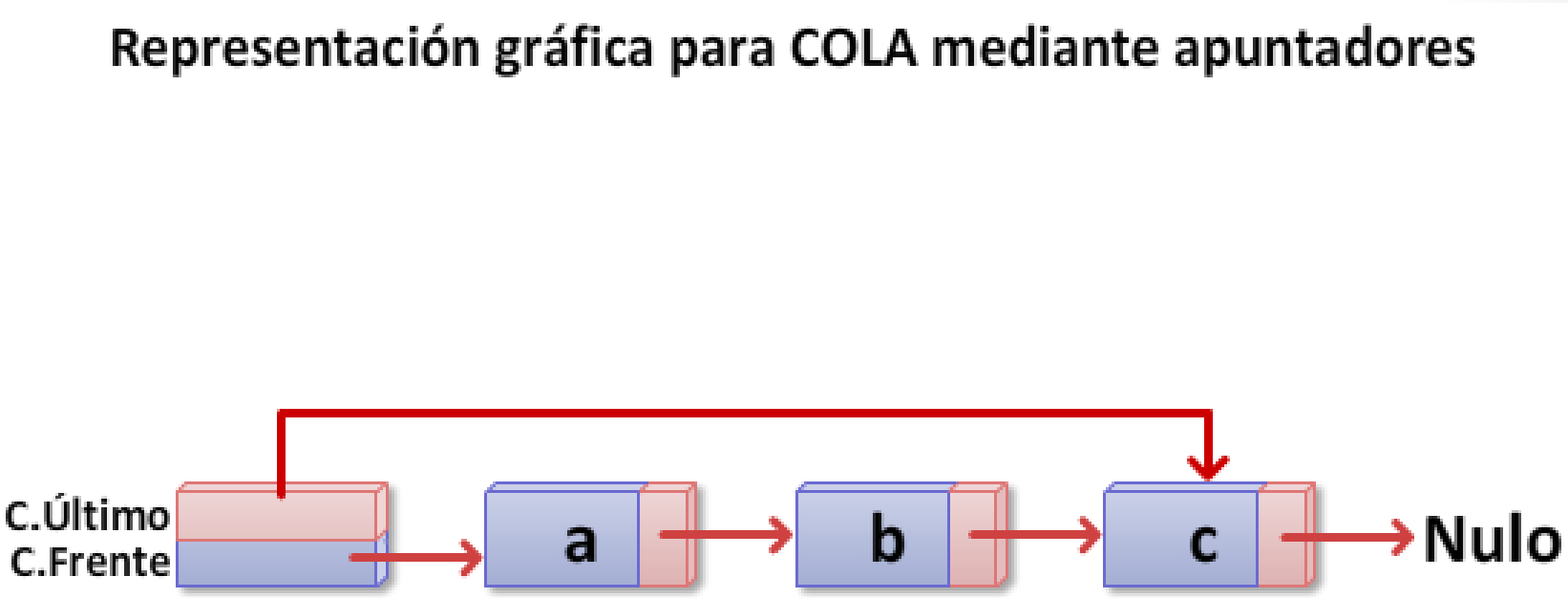
  - *Atrás* = *Frente*, y la última operación fue insertar entonces la cola esta **llena**.

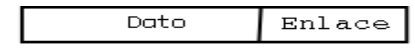
Implementación dinámica



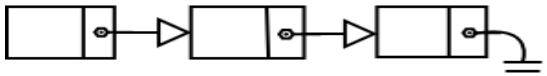
- Hablamos de una **Estructura de datos dinámica** cuando se le asigna memoria a medida que es necesitada, durante la ejecución del programa.
  - En este caso la memoria no queda fija durante la compilación.

Implementación dinámica del TAD Cola





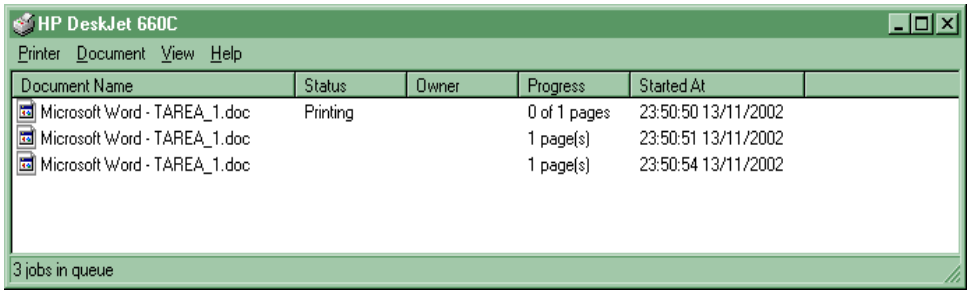
Estructura de un nodo

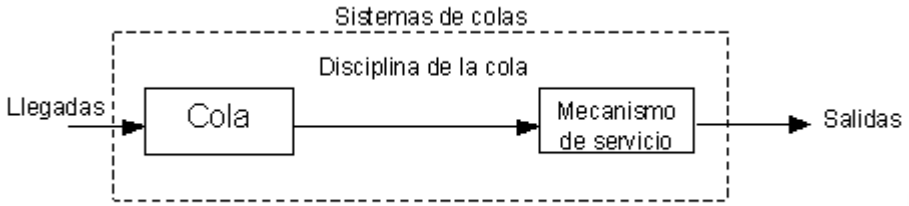


Lista enlazada

Aplicaciones del TAD Cola

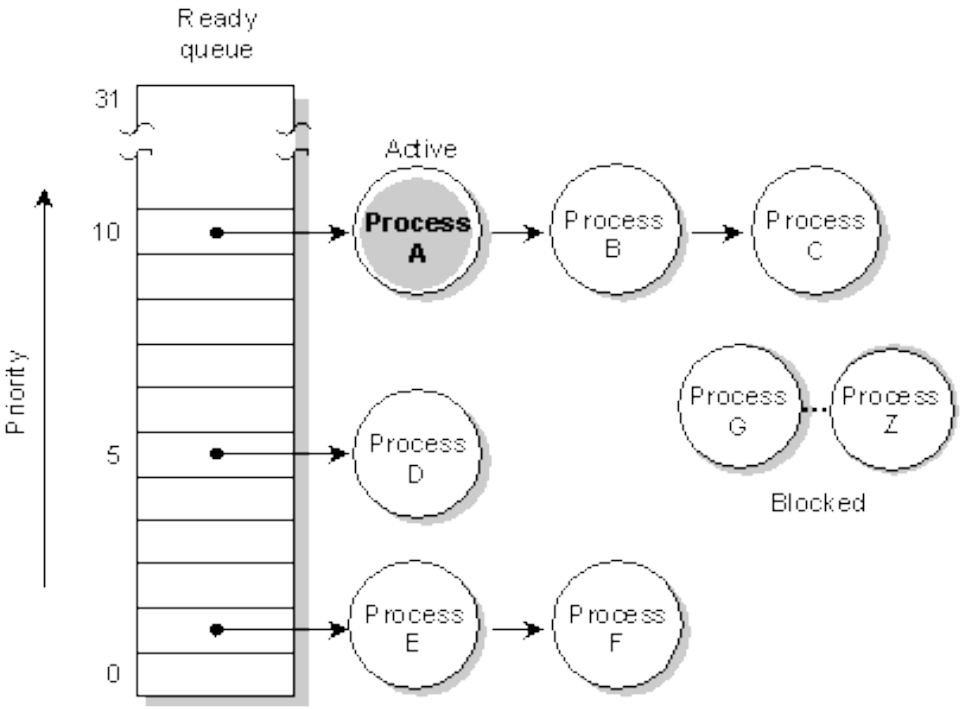
- Las colas se utilizan cuando tenemos que hacer cola para obtener un cierto “servicio” por parte de algún agente o “servidor”.
  - Cuando por ejemplo un programa debe:
    - Gestionar un servicio o recurso compartido como la impresora o el procesador.

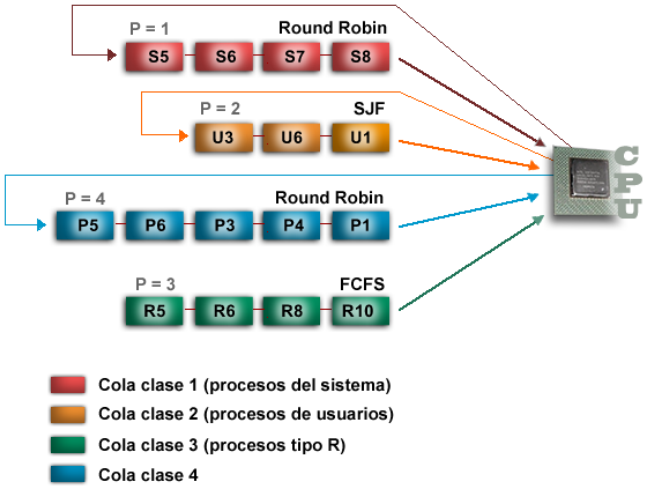


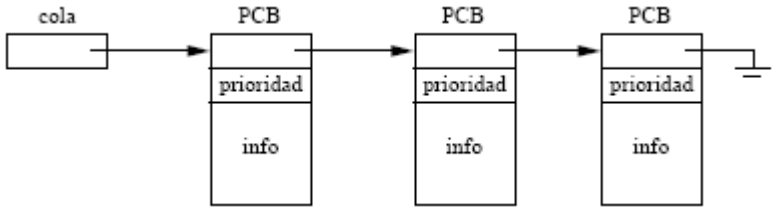


- P.g. para que un programa pueda ser ejecutado, el sistema operativo crea un nuevo proceso, y el procesador ejecuta una tras otra las instrucciones del mismo. En un entorno de multiprogramación, el procesador intercalará la ejecución de instrucciones de varios programas que se encuentran en memoria. El sistema operativo es el responsable de determinar las pautas de intercalado y asignación de recursos a cada proceso.

- El administrador de procesos del S.O. gestiona el recurso procesador a través de una cola de procesos.







Cola de prioridades

- Una cola de prioridad es una cola a cuyos elementos se les ha asignado una prioridad, de forma que el orden en que los elementos son extraídos ***(Desencolados)*** sigue las siguientes reglas:
  - El elemento con mayor prioridad es extraído primero.
  - Dos elementos con la misma prioridad son procesados según el orden en que fueron introducidos en la cola.

- Existen tres métodos básicos para la representación de colas de prioridad mediante estructuras lineales:

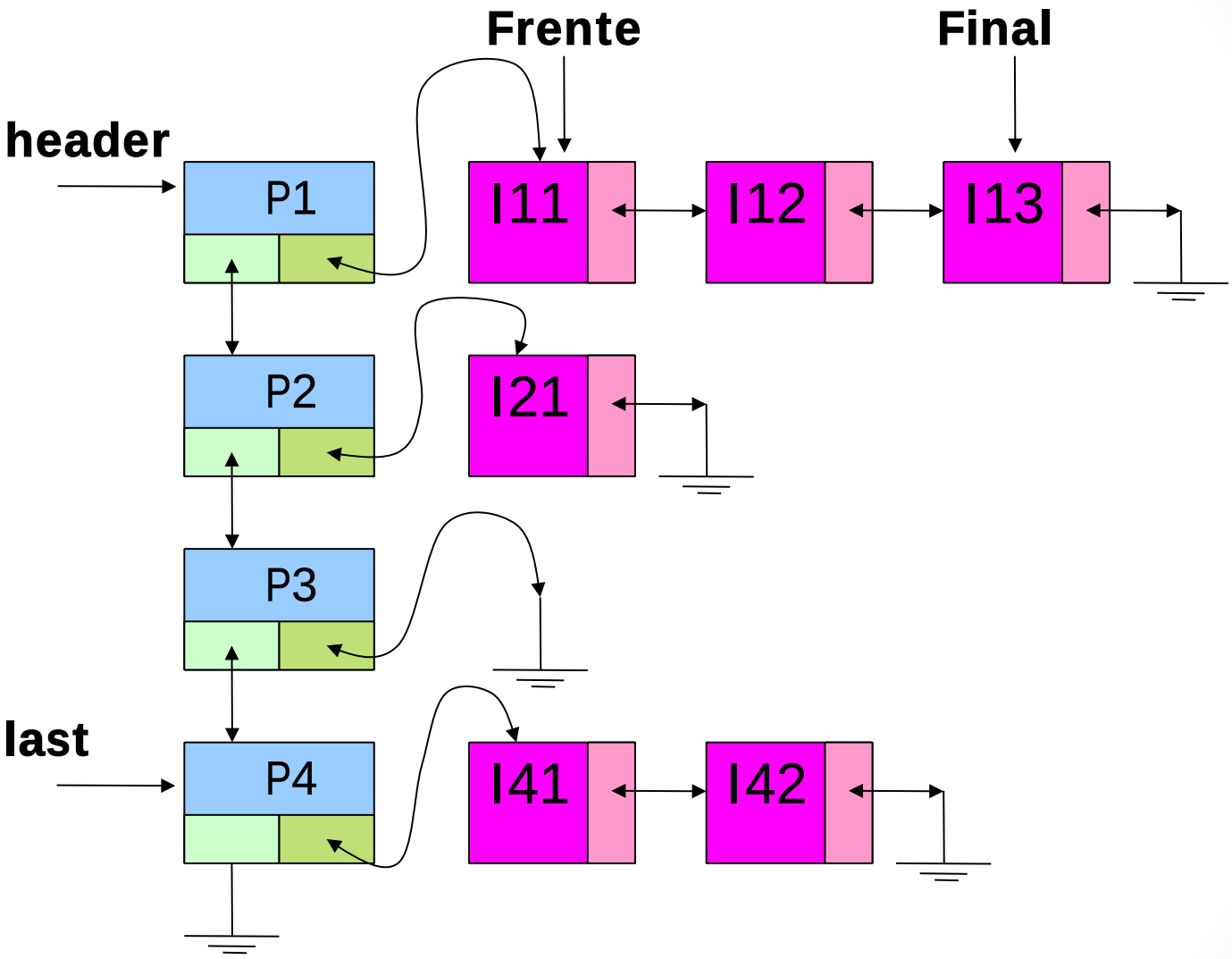
1. Tener la cola siempre ordenada de acuerdo a las prioridades de sus elementos, y sacar cada vez el primer elemento de ésta, es decir, el de mayor prioridad. En este caso, cuando se introduce un elemento en la cola, debe insertarse en el lugar correspondiente de acuerdo a su prioridad.

2. Insertar los elementos siempre al final de la cola, y cuando se va a sacar un elemento, buscar el que tiene mayor prioridad.

3. **Tener una cola para cada una de las prioridades de los elementos y atender siempre a la cola con elementos de mayor prioridad y encolar a los elementos en la cola de su prioridad correspondiente.**

- En las colas normales
  - Las operaciones están definidas en función del orden de llegada de los elementos
    - Al encolar un elemento ingresa al final de la cola
    - Al desencolar, sale del frente de la cola
  - En una cola, los elementos esperan por ser atendidos
    - Es justo, porque el que llega primero, se atiende primero

- En una cola de prioridad
  - Prioridad
    - El orden de atención, no esta dado solo por el orden de llegada
    - Cada elemento, tendrá asociado una cierta prioridad
    - Cada elemento será “procesado”, según su prioridad

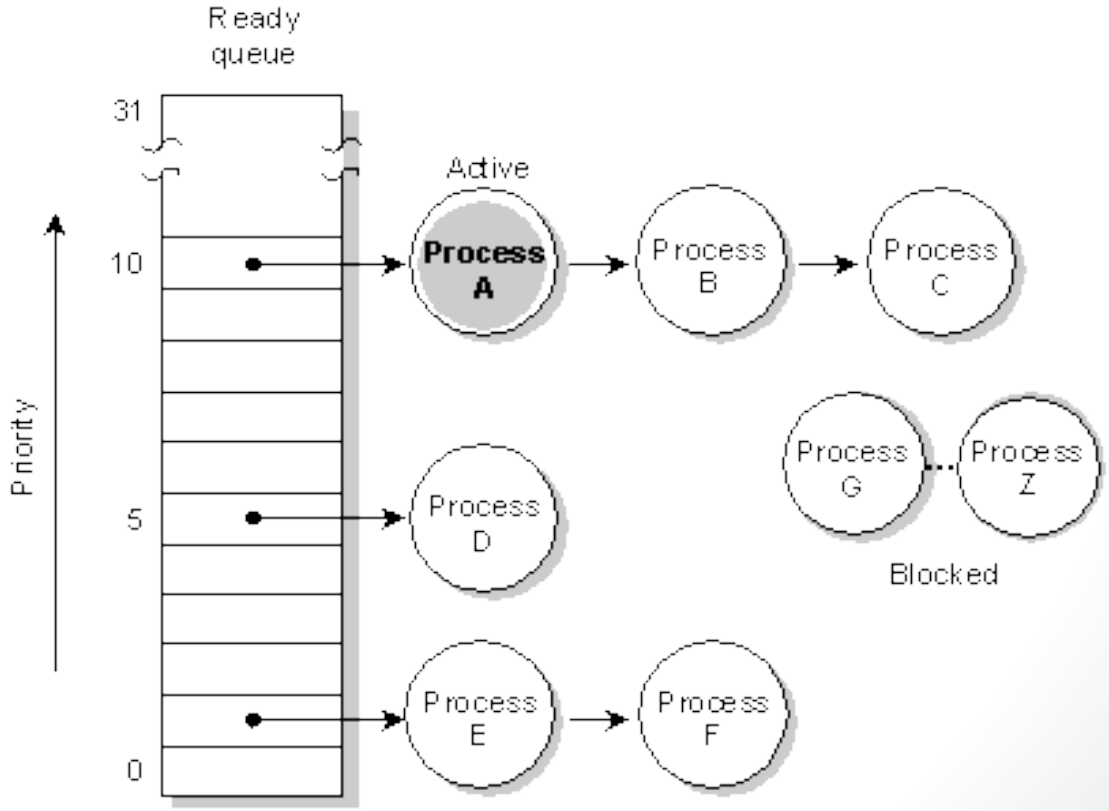

Frente
Final
header
P1
I11
I12
I13
P2
I21
P3
last
P4
I41
I42

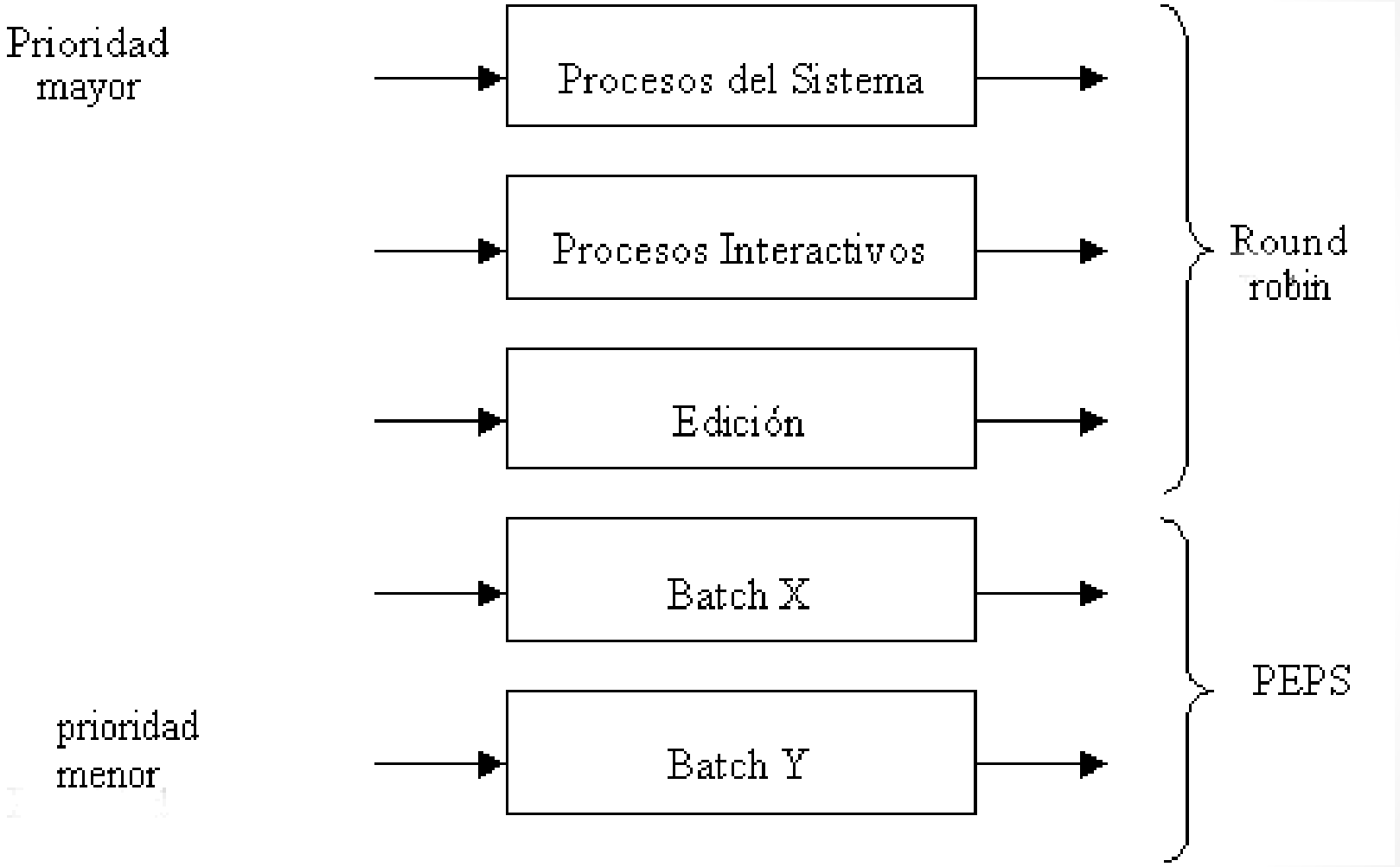
- Hay dos tipos de colas de prioridad
  - ***De Prioridad Ascendente***
    - ***Encolar (Queue):*** son encolados arbitrariamente
    - ***Desencolar (Dequeue):*** se remueve el elemento de menos prioridad de la cola
  - ***De Prioridad Descendente***
    - ***Encolar (Queue):*** son encolados arbitrariamente
    - ***Desencolar (Dequeue):*** se remueve el elemento de mayor prioridad de la cola

- Las colas de prioridad pueden contener
  - Enteros, Reales
  - Estructuras,
    - Estarían ordenadas (priorizadas) con base en uno o mas campos.

Cola de prioridades:
Administrador de procesos

- El administrador de procesos del S.O. gestiona el recurso procesador a través de una cola de procesos.




Prioridad
mayor
Procesos del Sistema
Procesos Interactivos
Edición
Round
robin
Batch X
Batch Y
PEPS
prioridad
menor