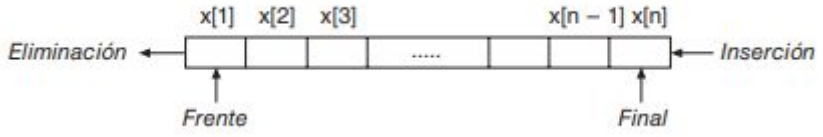
COLAS

Las colas son otro tipo de estructura lineal de datos similar a las pilas, diferenciándose de ellas en el modo de insertar/eliminar elementos. Una cola (queue) es una estructura lineal de datos

var array [1..n] de : C

en la que las eliminaciones se realizan al principio de la lista, frente (front), y las inserciones se realizan en el otro extremo, final (rear). En las colas el elemento que entró el primero sale también el primero; por ello se conoce como listas FIFO (first-in, first-out, “primero en entrar, primero en salir”). Así, pues, la diferencia con las pilas reside en el modo de entrada/salida de datos; en las colas las inserciones se realizan al final de la lista, no al principio. Por ello las colas se usan para almacenar datos que necesitan ser procesados según el orden de llegada.



En la vida real se tienen ejemplos numerosos de colas: la cola de un autobús, la cola de un cine, una caravana de coches en una calle, etc. En todas ellas el primer elemento (pasajero, coche, etc.) que llega es el primero que sale. En informática existen también numerosas aplicaciones de las colas. Por ejemplo, en un sistema de tiempo compartido suele haber un procesador central y una serie de periféricos compartidos: discos, impresoras, etc. Los recursos se comparten por los diferentes usuarios y se utiliza una cola para almacenar los programas o peticiones de los diferentes usuarios que esperan su turno de ejecución. El procesador central atiende —normalmente— por riguroso orden de llamada del usuario; por tanto, todas las llamadas se almacenan en una cola. Existe otra aplicación muy utilizada que se denomina cola de prioridades; en ella el procesador central no atiende por riguroso orden de llamada, aquí el procesador atiende por prioridades asignadas por el sistema o bien por el usuario, y sólo dentro de las peticiones de igual prioridad se producirá una cola.

Ejemplo Pseudocódigo

```
 var
punt: auxi
inicio auxi ← frente
elemento ← auxi→.elemento
frente ← frente→.sig
si frente = nulo entonces
        final ← nulo
fin_si
liberar(auxi)
fin_procedimiento
```

Como los elementos se extraen siempre por el frente, la cola estará vacía cuando frente = nulo

```
lógico función vacia(E punt: frente)
inicio
si frente = nulo entonces
        devolver(verdad)
si_no
        devolver(falso)
fin_si
fin_función
procedimiento consultarPrimero( E punt: frente; S tipo_elemento: elemento)
inicio
        si no vacia (frente) entonces
                elemento ← frente→.elemento
        fin_sin
fin_procedimiento
```