Estructuras de Datos

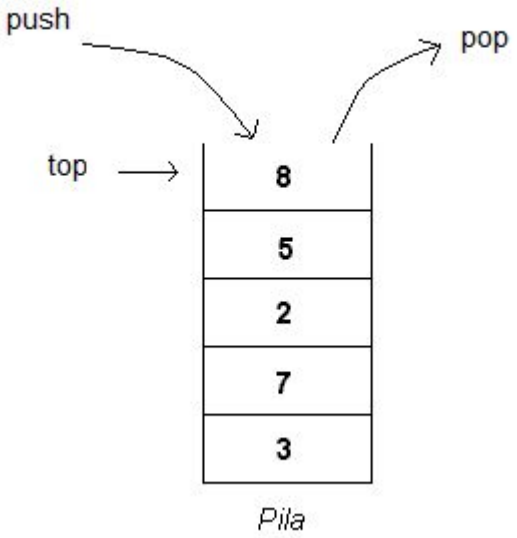
Pilas y Colas

Pila

Una Pila (*stack*) es una lista de elementos de la cual sólo se puede extraer el último elemento insertado. La posición en donde se encuentra dicho elemento se denomina *tope* de la Pila .

A partir de dicho comportamiento, se conoce a las pilas como una estructuras de datos *LIFO* (LAST IN - FIRST OUT, es decir, el último que entra es el primero que sale).

Gráficamente podemos representar a la Pila como



Pila

A diferencia de las listas y los arreglos, una Pila sólo tiene una interfaz que puede ser implementada de diferentes formas. Las operaciones con las que debe contar una implementación de esta estructura de datos son:

- **apilar(push)**: inserta un elemento en el tope de la Pila.
- **desapilar(pop)**: elimina el elemento que se encuentra en el tope de la Pila.
- **tope(top)**: retorna el elemento que se encuentre en el tope de la Pila, pero sin eliminarlo de ésta.
- **es_vacia(isEmpty)**: retorna true si la Pila no contiene elementos, false en caso contrario.

Pila

En este código podemos ver que se está declarando una Pila usando un arreglo de tamaño MAX_PILA (una constante previamente definida).

```
typedef struct _Pila {
  int datos[MAX_PILA];
  int ultimo;
} *Pila;
```

La variable ultimo representa el índice donde se encuentra el tope de la Pila .

Lo que tenemos que tener en cuenta es que en esta implementación la Pila puede estar llena ya que tiene un tamaño acotado.

Una variante, para que esto no suceda, es definir la Pila de esta forma:

```
typedef struct _Pila {
  SNodo* inicio;
} *Pila;
```

Ahora bien, deberíamos analizar qué tipo de lista nos conviene usar. Pensemos que push y pop agregan y remueven datos a la Pila por el mismo lado. Por lo tanto, sólo precisamos un puntero al inicio y, usar SNodo* es una buena idea.

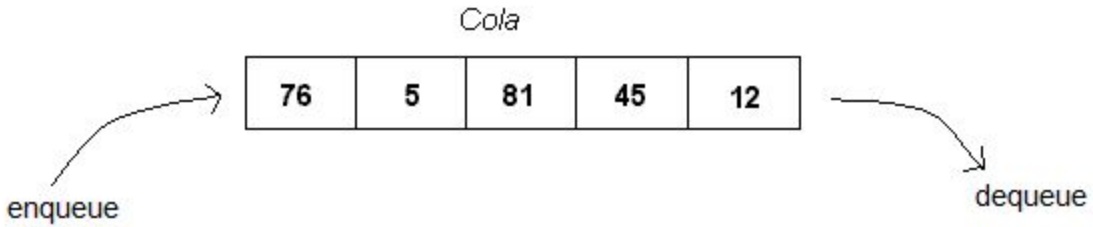
Cola

Una Cola (queue en inglés) es una lista de elementos en donde siempre se insertan nuevos elementos al final de la misma y, se extraen elementos desde el inicio.

Este comportamiento se conoce como FIFO (FIRST IN - FIRST OUT: el primero que entra es el primero que sale).

Cola

Podemos representar a la Cola como:



Cola

Al igual que la Pila, la Cola sólo tiene una interfaz que puede ser implementada de diferentes formas. Las operaciones con las que debe contar una implementación de esta estructura de datos son:

- **encolar(enqueue)**: inserta un elemento al final de la Cola.
- **desencolar(dequeue)**: elimina el elemento que se encuentra al inicio de la Cola.
- **es_vacia(isEmpty)**: retorna true si la Cola está vacía, false en caso contrario.

Como podemos ver, la diferencia entre Cola y Pila está en que agregamos dato de un extremo y, sacamos del otro.

Para la implementación con listas, deberíamos:

- usar una lista circular doblemente enlazada o,
- la opción de un puntero al inicio y uno al final

para evitar recorrer la lista cada vez que querramos **desencolar** un elemento.

Cola

En la implementación con arreglos, a diferencia de la implementación que se hizo para Pila vamos a necesitar dos variables: primero y ultimo, además de la cantidad máxima de elementos que se puedan almacenar: MAX_COLA.

Ahora, por qué es necesario tener dos variables? Porque ambas se van a ir desplazando en la medida que se vaya usando las funciones de **encolar** y **desencolar**.

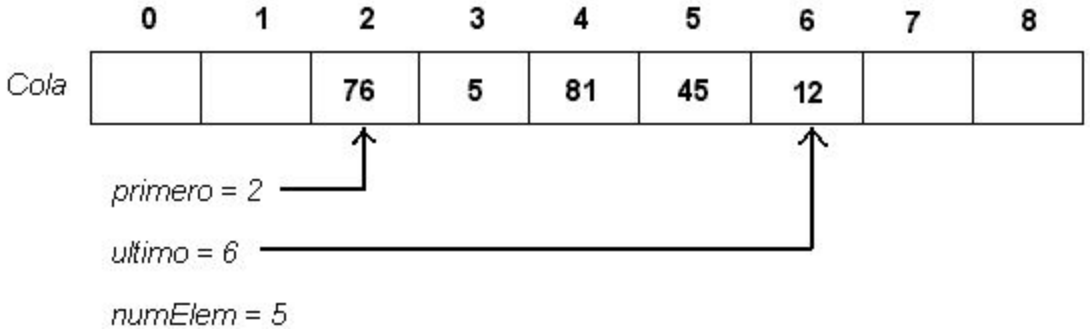
Es decir:

- primero: indica el índice de la posición del primer elemento de la cola, es decir, la posición del elemento a eliminar cuando se invoque **desencolar**.
- ultimo: indica el índice de la posición de último elemento de la cola. Si se invoca **encolar**, el elemento debe ser insertado en el casillero siguiente al que indica la variable.

Además, la cantidad de elementos que almacena la Cola, en un momento determinado, puede sacarse como consecuencia de la diferencia entre ambas variables.

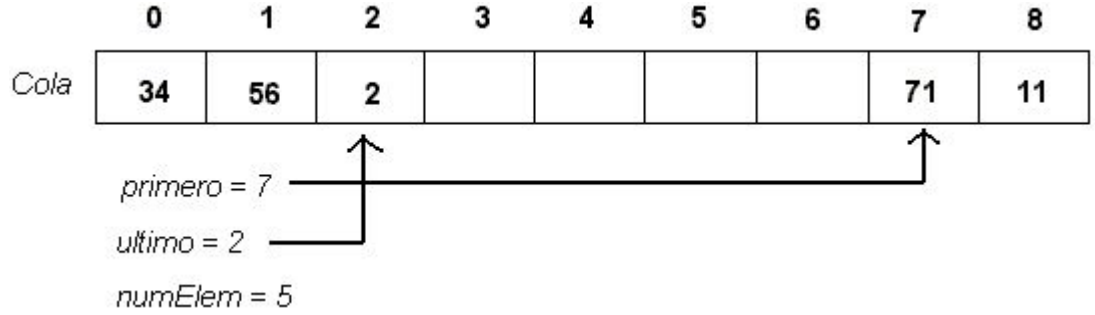
Cola

Un primer ejemplo podría ser:



Ahora bien, ¿qué pasa si la variable ultimo sobrepasa el rango de índices del arreglo?

En ese caso, podemos pensar que si después de insertar un elemento el índice ultimo == MAX_COLA, entonces se asigna ultimo = *0* , y los siguientes elementos serán insertados al comienzo del arreglo.



Podemos ver que esto no esto no produce ningún efecto en la lógica de las operaciones de la la Cola, pues siempre se saca el elemento referenciado por el índice primero.

Este enfoque es conocido como *implementación con arreglo circular*, y la forma más fácil de implementarlo es haciendo la aritmética de subíndices módulo MAX_COLA.