Estructura de datos:

Cola

La estructura de datos cola es otra estructura lineal sencilla como la anterior, administrada por dos eventos. Dicha estructura es clave para el funcionamiento de los sistemas operativos que la utilizan para la gestión y planificación de procesos. También es usada por otras aplicaciones que deben manejar la concurrencia del uso de recursos compartidos –por ejemplo, la impresora–, dado que estas pueden trabajar administrando prioridades. Una cola es una colección de elementos que se agregan y quitan basándose en el principio de primero en entrar-primero en salir (FIFO, First In-First Out). Esto quiere decir que el primer elemento en ser insertado es el primero en ser eliminado. Un elemento puede ser agregado en cualquier momento a una cola, esto se hace por el final, pero solo se puede acceder o eliminar el elemento que esté en el frente de la misma, como se observa en la figura 1. Es una estructura lineal dinámica de datos que no están ordenados y cuyas actividades de *inserción* y *eliminación* se realizan a través de los índices llamados **frente** y **final** respectivamente, sobre la cual además se puede operar con criterios de prioridad.

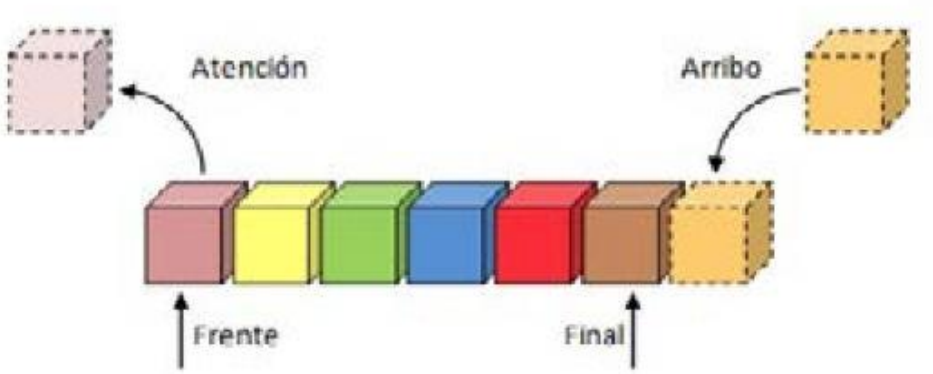

Figura 1. Funcionamiento de una cola

hay dos actividades o eventos que se encargan de administrar los elementos de una cola. Entonces igual que en el caso del TDA pila podríamos volver a afirmar que el valor del dato almacenado no influye en el manejo de la estructura –manteniéndose entonces una relevancia secundaria del valor de los datos para la estructura, más allá de la importancia de almacenarlos–, salvo los casos particulares en los que se aplica cola de prioridad donde, además de los eventos, la prioridad de los datos influye en el funcionamiento de la estructura. Estos eventos son dos: *arribo* cuando se agrega un nuevo elemento al final y *atención* cuando se saca el elemento que está en el frente.

Esto implica que solo se puede acceder al elemento que está en el frente de la cola, la forma de acceder a los demás elementos de la colección es atender cada uno de estos, de a uno a la vez.

Algunos ejemplos de la vida cotidiana donde se pueden encontrar el uso de cola de manera natural son en la parada del colectivo, en las cajas de los supermercados, venta de entradas de evento, en los cajeros automáticos, turnos en la guardia de un hospital o consultorio, etcétera. La representación virtual de este modelo también lo utilizamos de manera implícita sin darnos cuenta de esto, por ejemplos si hay una impresora en una casa u organización a la cual se envían a imprimir documentos de diferentes maquinas, se imprime el primero que llega y el resto queda en espera –es decir quedan en la cola de impresión para ser atendidos una vez finalice de imprimir el documento actual–.

Arranquemos con el diseño del TDA cola, el cual estará compuesto básicamente de dos elementos que llamaremos frente y final. Estos son punteros que se encargan de apuntar a un nodoCola, que su vez está compuesto de dos elementos, información y siguiente –como ya hemos vista anteriormente–, y las funciones que describen su comportamiento, que se enumeran a continuación –opcionalmente puede agregarse el elemento tamaño para determinar la cantidad de elementos en lacola, y evitar tener que contar los nodos con una función adicional–.

Pero estaría conformado con las siguientes funciones o métodos:

1. arribo (cola, elemento). Agrega el elemento a la final de la cola;
2. atención (cola). Elimina y devuelve el elemento almacenado en el frente de la cola;
3. cola_vacia (cola). Devuelve verdadero (true) si la cola no contiene elementos;
4. en_frente (cola). Devuelve el valor del elemento que está almacenado en el frente de la cola sin eliminarlo;
5. tamaño (cola). Devuelve la cantidad de elementos en la cola;
6. mover_al_final (cola). Elimina el elemento en el frente de la cola y lo inserta en el final de la misma;

Pasos para hacer una COLA:

1- **__init__**: constructor de la clase.

2- **arrive (***Arribo***):** Agrega el elemento a la final de la cola.

3- **attention (***Atención***)**: Elimina y devuelve el elemento almacenado en el frente de la cola.

4- **size (***Tamaño***):** Devuelve la cantidad de elementos en la cola.

5- **on_front (***en frente***):** Devuelve el valor del elemento que está almacenado en el frente de la cola sin eliminarlo;

6- **move_to_end (***Mover al final***):** Elimina el elemento en el frente de la cola y lo inserta en el final de la misma.

7- **show (***Mostrar***):** muestra toda la cola.

```python
from typing import Any, Optional

You, yesterday | 1 author (You)
class Queue:

    def __init__(self):
        self.__elements = []

    def arrive(self, value: Any) -> None:
        self.__elements.append(value)

    def attention(self) -> Optional[Any]:
        return (
            self.__elements.pop(0)
            if self.__elements
            else None
        )

    def size(self) -> int:
        return len(self.__elements)

    def on_front(self) -> Optional[Any]:
        return (
            self.__elements[0]
            if self.__elements
            else None
        )

    def move_to_end(self) -> Optional[Any]:
        if self.__elements:
            value = self.attention()
            self.arrive(value)
            return value        You, yesterday •

    def show(self):
        for i in range(len(self.__elements)):
            print(self.move_to_end())
```