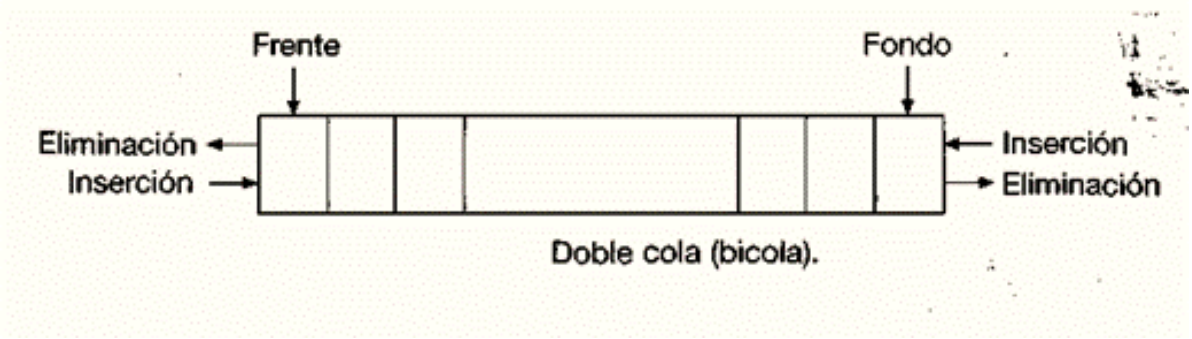
Ontiveros Lara Claudia Sarahi 17212167

Doble Cola

Una **cola doble**, también conocida como cola de doble extremo, es una colección ordenada de ítems similar a la cola. Tiene dos extremos, frente y final, y los ítems permanecen posicionados en la colección. Lo que hace a una cola doble diferente es la naturaleza no restringida de las operaciones de agregar y remover ítems. Los ítems nuevos se pueden agregar en el frente o en el final. Del mismo modo, los ítems existentes se pueden eliminar de cualquier extremo. En cierto sentido, esta estructura lineal híbrida proporciona todas las capacidades de las pilas y las colas en una única estructura de datos.

Es importante tener en cuenta que a pesar que las colas dobles pueden asumir muchas de las características de las pilas y de las colas, ellas no requieren los ordenamientos LIFO y FIFO que son respetados por esas estructuras de datos. Depende de usted hacer un uso consistente de las operaciones agregar y remover. Gráficamente se representan así:



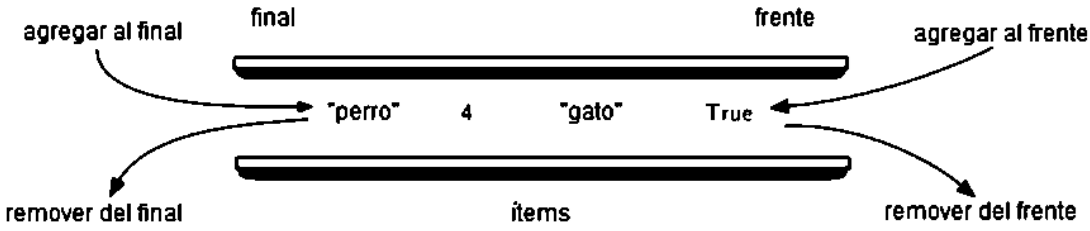


Figura 1: Una cola doble de objetos de datos de Python

Existen dos variantes de la doble cola:

- Doble cola de entrada restringida: Este tipo de doble cola acepta solamente la inserción de elementos por un extremo; mientras que puede eliminar por ambos.

- Doble cola de salida restringida: Este tipo de doble cola acepta solamente la eliminación de elementos por un extremo; mientras que puede insertar por ambos.

El tipo abstracto de datos Cola Doble se define por la siguiente estructura y las siguientes operaciones. Una cola doble está estructurada, como se describió anteriormente, como una colección ordenada de ítems en la que se añaden y se retiran ítems de cualquier extremo, ya sea por el frente o por el final. Las operaciones de la cola doble se dan a continuación.

- **ColaDoble()** Crea una cola doble nueva que está vacía. No necesita parámetros y devuelve una cola doble vacía.
- **agregarFrente(item)** Añade un nuevo ítem al frente de la cola doble. Necesita el ítem y no devuelve nada.
- **agregarFinal(item)** Añade un nuevo ítem en el final de la cola doble. Necesita el ítem y no devuelve nada.
- **removerFrente()** Elimina el ítem que está en el frente de la cola doble. No necesita parámetros y devuelve el ítem. La cola doble se modifica.
- **removerFinal()** Elimina el ítem que está al final de la cola doble. No necesita parámetros y devuelve el ítem. La cola doble se modifica.
- **estaVacia()** Comprueba si la cola doble está vacía. No necesita parámetros y devuelve un valor booleano.
- **tamano()** Devuelve el número de ítems en la cola doble. No necesita parámetros y devuelve un entero.

Ejemplo:

Operación de cola doble	Contenido de la cola doble	Valor devuelto
`d.estaVacia()`	`[]`	`True`
`d.agregarFinal(4)`	`[4]`	
`d.agregarFinal('perro')`	`['perro',4,]`	
`d.agregarFrente('gato')`	`['perro',4,'gato']`	
`d.agregarFrente(True)`	`['perro',4,'gato',True]`	
`d.tamano()`	`['perro',4,'gato',True]`	`4`
`d.estaVacia()`	`['perro',4,'gato',True]`	`False`
`d.agregarFinal(8.4)`	`[8.4,'perro',4,'gato',True]`	
`d.removerFinal()`	`['perro',4,'gato',True]`	`8.4`
`d.removerFrente()`	`['perro',4,'gato']`	`True`

Ejemplo de colas dobles en Python:

```
class ColaDoble:

    def __init__(self):

        self.items = []


    def estaVacia(self):

        return self.items == []


    def agregarFrente(self, item):

        self.items.append(item)


    def agregarFinal(self, item):

        self.items.insert(0,item)


    def removerFrente(self):

        return self.items.pop()


    def removerFinal(self):

        return self.items.pop(0)


    def tamano(self):

        return len(self.items)
```

Aplicaciones de las colas:

- En Ciencias Computacionales, Investigación de Operaciones, y muchas otras áreas donde datos son almacenados para ser procesados posteriormente, ejecutando una función de buffer (e.g., inventarios, threads, simulación, manufactura, etc).

Referencias:

Aguilar, J. Luis, Martinez, Z. Ignacio, Azuela, F. Matilde, & Garcia, S. Lucas. (1999). Estructura de datos. Libro de problemas. Madrid, España: Mc Graw Hill.

Cairó, O., & Guardati, S. (2002). Estructuras de datos (3rd ed.). México: McGraw-Hill.

Martínez, I., & Aguilar, L. (2011). Estructuras de datos en Java. España: McGraw-Hill España.