**Que es una pila en estructura de datos?**  
  
Una pila es una estructura de datos lineal que sigue el principio LIFO (*Last In, First Out*), lo que significa que el último elemento en entrar es el primero en salir. Este comportamiento se puede comparar con una pila de platos: solo se puede acceder al plato que está arriba, y para llegar a los de abajo hay que quitar primero los de encima.

En una pila, la inserción y eliminación de elementos solo ocurre en un extremo llamado cima o tope (*top*), lo que impone una forma específica y ordenada de acceder a los datos.

Desde el punto de vista técnico, una pila es una estructura de datos abstracta que almacena elementos en una colección con acceso restringido. No se puede acceder libremente a cualquier elemento; solo se puede trabajar con el que está en el tope. Las pilas suelen implementarse utilizando un arreglo (array) o una lista enlazada, junto con un puntero o índice al tope, que indica la posición del último elemento agregado.

Entre las **ventajas** más destacadas de las pilas se encuentran:

* **Gestión ordenada (LIFO):** Permite un control preciso del orden de ejecución de las operaciones.
* **Fácil implementación:** Puede construirse fácilmente utilizando arreglos o listas enlazadas.
* **Alta eficiencia:** Las operaciones básicas como push (insertar), pop (eliminar) y peek (consultar el tope) tienen un tiempo de ejecución constante, es decir, **O(1)**.
* **Utilidad en programación:** Son ampliamente utilizadas en múltiples contextos, como el deshacer acciones (por ejemplo, Ctrl+Z), la gestión de llamadas a funciones (pila de ejecución), el recorrido de estructuras como árboles y grafos, y la evaluación de expresiones matemáticas.
* **Uso controlado de memoria:** Solo se almacena la información estrictamente necesaria y en el orden necesario.

Sin embargo, las pilas también presentan algunas **desventajas**:

* **Acceso limitado:** Solo es posible acceder al elemento en el tope, lo que impide trabajar directamente con elementos intermedios.
* **No apta para búsquedas:** Buscar un elemento específico dentro de una pila no es eficiente.
* **Tamaño limitado (en implementaciones estáticas):** Si se usa un arreglo de tamaño fijo, puede producirse un **desbordamiento (overflow)** al intentar insertar más elementos de los permitidos.
* **Riesgo de underflow:** Si se intenta eliminar un elemento de una pila vacía, se produce un error conocido como **underflow**.
* **No conserva el orden original de entrada:** Al funcionar bajo el principio LIFO, el primer elemento que se inserta será el último en salir.