Pilas

Una pila es un estructura de datos que sigue un patrón llamado “LIFO” (Last IN, First OUT) el último que entra es el primero que sale, Para hacer una analogía con el mundo real imagina una pila de platos en donde pones uno encima de otro… el único plato al cual puedes acceder es el último que colocaste (A menos que te arriesgues a hacer un desastre.) Observa el siguiente gráfico:

FIG 00: representación de LIFO.

LIFO es igual que una pila de platos, la única manera de acceder a un plato en concreto es quitando los que hay encima de él.

Por qué existen las pilas:

- Control de la recursión: cada que una llamada se llama a sí misma se almacena en una pila.
- Reversas: son ideales para acciones como el control + z.
- Control de recorrido de grafos.

Advertencia: Se espera que el lector entienda que al igual que el mundo real cuando un plato se quita de la pila este desaparece (Se va de la pila).

Una pila no es más que una colección de Nodos, los cuales contienen datos y un enlace a otro nodo. Cada nodo contiene 2 elementos:

- Dato.
- Nodo siguiente.

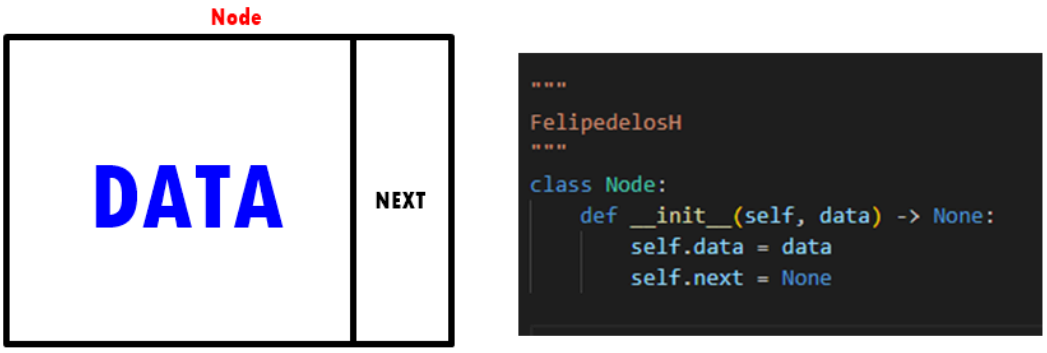


FIG 00: representación de un nodo al lado de su respectivo código.

Para poder utilizar esto necesitamos crear algo que se llama "Pila"

Una pila es una colección de nodos que nos va a permitir guardar la información, dicha clase contiene un apuntador y varios datos… si observa la siguiente figura usted podrá notar cómo funciona:

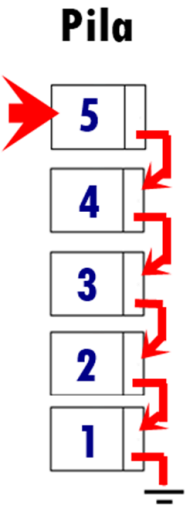


FIG 01: Ejemplo visual de una pila

Las pilas necesitan de los siguientes atributos y métodos para poder funcionar:

- Se necesita un pivote el cual es el encargado de moverse para leer/escribir la información.
- Se necesita una variable para obtener el total de elementos en la pila.
- Se necesita un método para agregar un elemento a la pila.
- Se necesita un método para quitar el elemento de la cima de la pila.

Que es el pivote

El pivote no es más que una instancia del nodo, pero no podemos empezar la pila instanciando un nodo sin una data (o al menos no es recomendable) el pivote es la cabeza "un nodo" y se encarga de moverse para recorrer la pila y poder agregar elementos.

El pivote/cabezal es el encargado de ser el punto de acceso a los datos de la pila sin él no existiría una forma de recorrerla.

```
class Stack:
    def __init__(self) -> None:
        self.pivot = None
        self.size = 0
```

Advertencia: el pivote inicia en None dado a que cuando la cola es creada no posee datos.

Como se agregan elementos a la pila

Yo he decidido hacerlo de manera iterativa y siempre se cumple la misma lógica en caso de que la pila esté o no vacía.

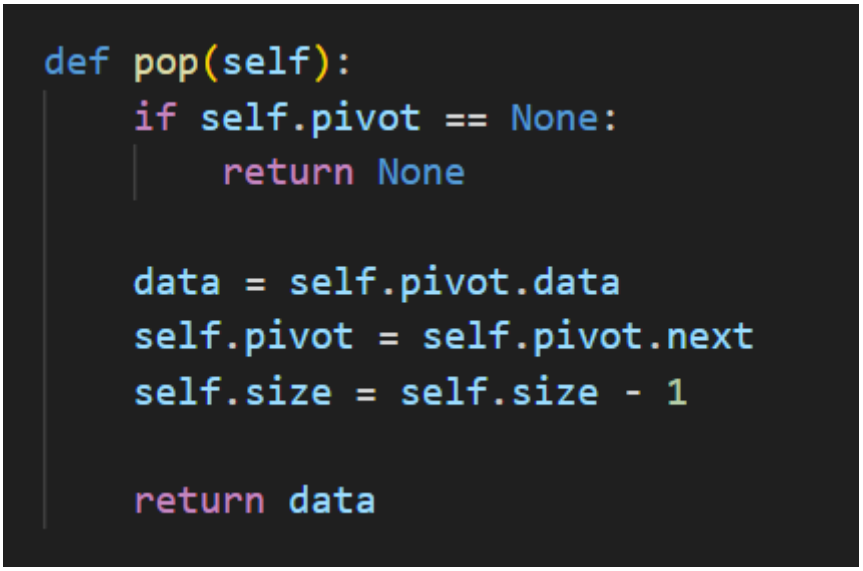
Se crea un nuevo nodo, luego se pone como siguiente del nuevo nodo al pivote, luego establecemos ese nuevo nodo como el nuevo pivote y por último se incrementa el contador de tamaño de la pila.

```
def push(self, data):
    new_node = Node(data)
    new_node.next = self.pivot
    self.pivot = new_node
    self.size = self.size + 1
```

Como se sacan datos de la pila

La lógica es que cuando el elemento de la pila sea quitado este desaparece de la pila:

Lo único que hacemos es declarar una variable temporal con la información que contiene la pila en la cima “TOP” y luego de ello procederemos a establecer el pivote al siguiente además de incrementar la variable que nos ayuda a saber el tamaño de la pila.



```
def pop(self):
    if self.pivot == None:
        return None

    data = self.pivot.data
    self.pivot = self.pivot.next
    self.size = self.size - 1

    return data
```