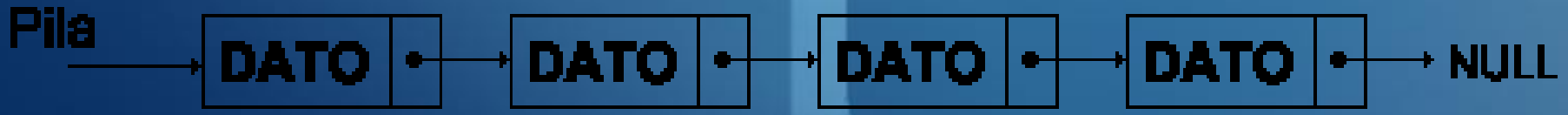
Pilas

- Una pila es un tipo de datos donde sólo se pueden insertar y eliminar nodos en uno de los extremos de la lista.
- Estas operaciones se conocen como "push" y "pop", respectivamente.
- Además, las escrituras de datos siempre son inserciones de nodos, y las lecturas siempre eliminan el nodo leído.
- Estas características implican un comportamiento de lista LIFO (Last In First Out), el último en entrar es el primero en salir.
- Es similar a una pila de platos. Sólo es posible añadir platos en la parte superior de la pila, y sólo pueden tomarse del mismo extremo.



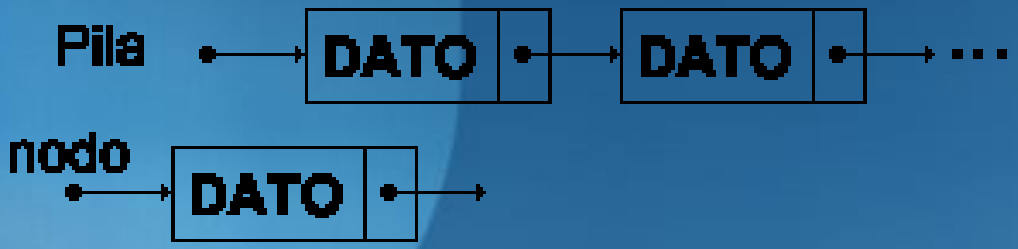
Push

- En una pila vacía:

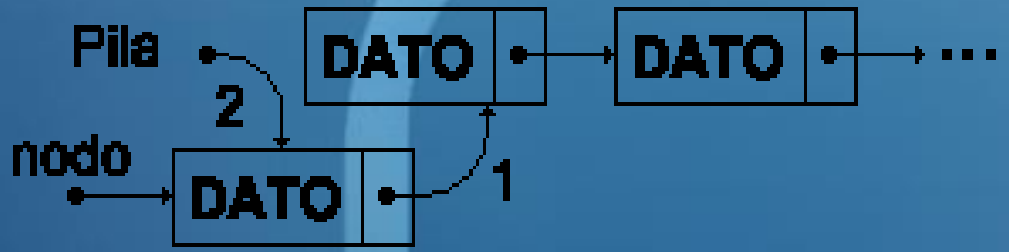
  nodo->siguiente apunta a NULL.
- Pila apunta a nodo.

Push

- En una pila no vacía
- 1. Se crea un nodo vacío

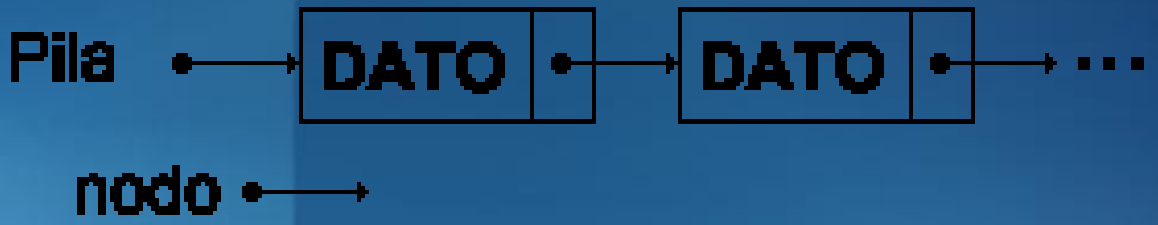


- 2. Se enlaza a la pila

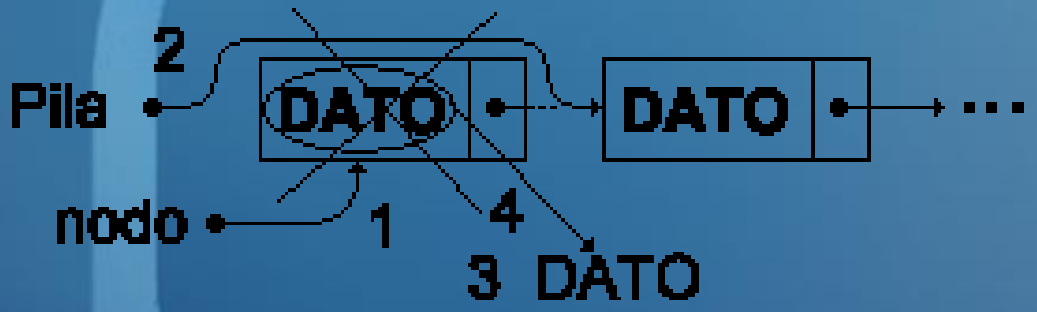


Pop

- Partimos de una pila con uno o más nodos, y usamos un nodo auxiliar:



1. Hacemos que nodo apunte al primer elemento de la pila, es decir a Pila.
2. Asignamos a Pila la dirección del segundo nodo de la pila: Pila->siguiente.
3. Guardamos el contenido del nodo para devolverlo como retorno: la operación pop equivale a leer y borrar.
4. Liberamos la memoria asignada al primer nodo, el que queremos eliminar.



```
#include<stdlib.h>
#include<stdio.h>

typedef struct nodo{
	int valor;
	struct nodo *siguiente;
}tipoNodo;

tipoNodo* pila;

//funciones

void push(tipoNodo** , int );
int pop(tipoNodo** );
void cargar();
void listar();

int main(){

	pila = NULL;
	cargar();
	listar();

	return 0;
}
```

```
void cargar(){

	int n = 1;
	while(n != 0){
		printf("Ingrese Nro (0 = fin):");
		scanf("%i", &n);
		if(n != 0)
			push(&pila, n);
	}
}

void listar(){
	tipoNodo *z = pila;
	printf("\nContenido de la Pila\n");
	while (z != NULL)
		printf("%i\n", pop(&z));

}
```

```
void push(tipoNodo**pila, int v){
    tipoNodo* nuevo;

    /*crear el nuevo nodo*/
    nuevo = (tipoNodo*) malloc(sizeof(tipoNodo));
    nuevo->valor = v;

    /*Agregamos la pila a continuacion del nuevo nodo*/
    nuevo->siguiente = *pila;
    /*Ahora ponemos el comienzo en el nuevo*/
    *pila = nuevo;
}
```

```
int pop(tipoNodo **pila){
    tipoNodo* nodo; //variable auxiliar
    int v; //variable de retorno

/*nodo apunta a primer elemeto que sacaremos*/
    nodo = *pila;
/*asignamos pila, toda la pila menos el primer elemento*/
    *pila = nodo->siguiente;
/*guardamos el valor de retorno*/
    v = nodo->valor;
    /*liberamos la memoria*/

    free(nodo);
    return v;
}
```