
Tema 2.2 TAD lineales

TAD Pila

Estructura de Datos y Algoritmos (EDA)



Contenidos

2.1. ¿Qué es un TAD Lineal?

2.2. TAD Pila

2.3. TAD Cola

2.4. TAD Lista

2.4.1 Implementación con una Lista Simplemente Enlazada

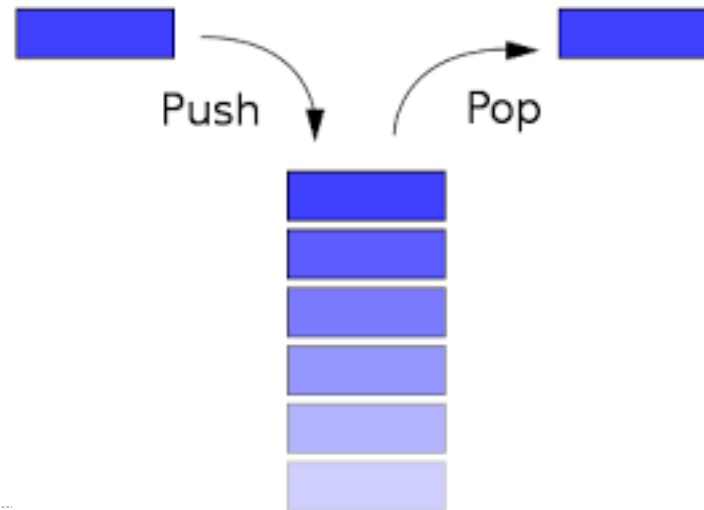
2.4.2 Implementación con una Lista Doblemente Enlazada

Objetivos

- Al final de la clase, los estudiantes deben ser capaces de:
 1. Entender el principio LIFO (last-in, first-out)
 2. Explicar el funcionamiento de una Pila
 3. Especificar formalmente una Pila
 4. Implementar una Pila utilizando una estructura dinámica

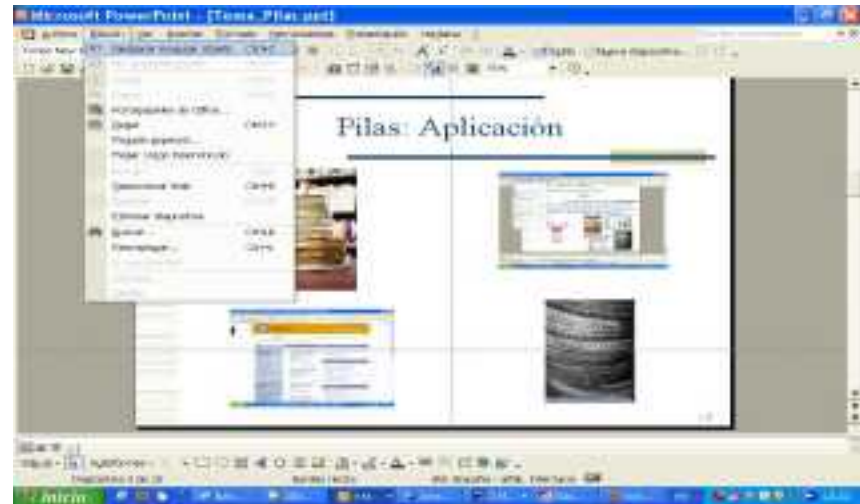
TAD Pila: Introducción

- ▶ Estructura de datos lineal basada en el principio **LIFO (last-in, first-out)**
- ▶ Las operaciones de inserción (**push**) y borrado (**pop**) se realizan en una sola posición, que es conocido como **top** o **peak**



Aplicación de Pilas

Undo operaciones



Back Navigation



Operaciones

- ▶ **push(e)**: añade un objeto *e* a la cima de la Pila
- ▶ **pop()**: borra y devuelve el elemento en la cima de la Pila
- ▶ **top()**: devuelve el elemento de la cima de la Pila
- ▶ **size()**: devuelve el número de elementos almacenados en la Pila.
- ▶ **isEmpty()**: devuelve *true* si la Pila está vacía; en otro caso, *false*.

Ejemplo

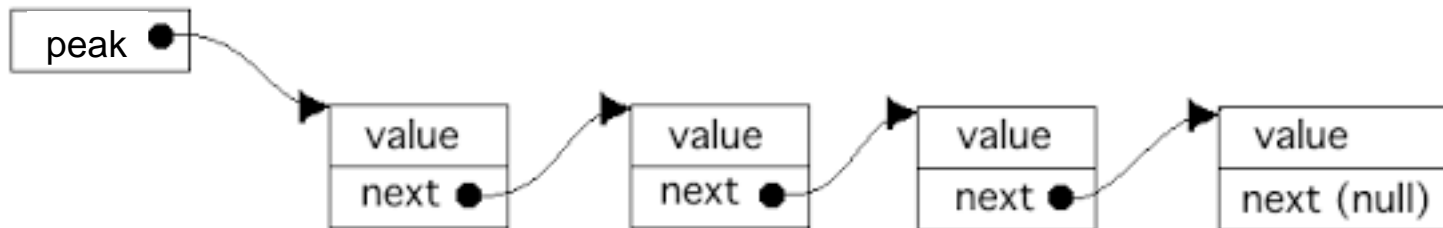
Operation	Stack	Output
push('h')	(h)	-
push('e')	(h,e)	-
top()	(h,e)	e
push('l')	(h,e,l)	-
push('l')	(h,e,l,l)	-
push('o')	(h,e,l,l,o)	-
top()	(h,e,l,l,o)	o
push('!')	(h,e,l,l,o,!)	-
top()	(h,e,l,l,o,!)	!
size()	(h,e,l,l,o,!)	6
isEmpty()	(h,e,l,l,o,!)	false
pop()	(h,e,l,l,o)	!

TAD Pila

```
public interface IStack {  
  
    public boolean isEmpty();  
    public void push(Integer elem);  
    public Integer pop();  
    public Integer top();  
    public int getSize();  
  
}
```


Implementación de un TAD Pila usando una estructura dinámica

- Una Pila se puede representar como una secuencia de nodos
- Solo permitimos insertar, borrar o leer el primer elemento (peak) de la secuencia



Class SNode

```
public class SNode {  
  
    public Integer elem;  
    public SNode next;  
  
    public SNode(Integer e) {  
        elem = e;  
    }  
  
}
```

Implementación de un TAD Pila usando una estructura dinámica

```
public class SStack implements IStack {  
    SNode peak = null;  
    int size;  
  
    public boolean isEmpty() {  
        return peak == null;  
    }  
}
```

Java proporciona un constructor por defecto que crea una Pila vacía (peak es nulo).

push() method

```
public void push(Integer newElem) {  
    SNode newNode = new SNode(newElem);  
    newNode.next = peak;  
    peak = newNode;  
    size++;  
}
```

top() method

```
public Integer top() {  
    if (isEmpty()) {  
        System.out.println("The stack is empty.");  
        return null;  
    }  
    return peak.elem;  
}
```

pop() method

```
public Integer pop() {  
    if (isEmpty()) {  
        System.out.println("The stack is empty.");  
        return null;  
    }  
    Integer elem = peak.elem;  
    peak = peak.next;  
    size--;  
    return elem;  
}
```