Tema 6: Pilas

Héctor Xavier Limón Riaño

April 3, 2024

Contents

1 Fundamentos **1**

2 Implementación **2**

3 Implementación con clases **3**

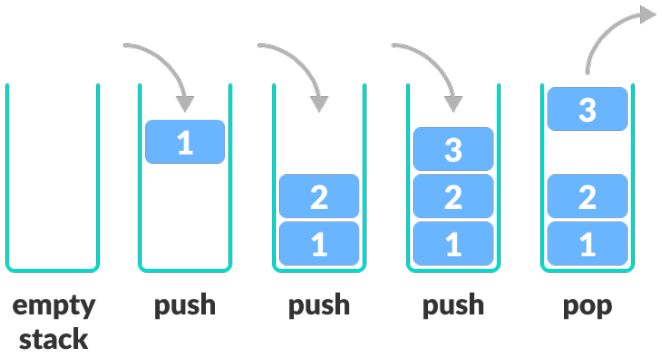
4 Ejemplos **4**

1 Fundamentos

- Las pilas son una estructura de datos similar a las listas
- Son estructuras lineales (aunque no indexables), dinámicas pero no mutables
- A diferencia de una lista, el acceso a elementos está restringido, sólo se puede tener acceso a un elemento a la vez (el tope de la pila)
- En esta estructura, el último elemento que se agrega es el primero que puede salir
- Es una estructura LIFO (last-in-First-out)
- En computación las pilas se utilizan para muchas cosas:
  - Analizadores sintácticos en compiladores
  - Orden de llamadas a funciones (se verá más al respecto en el tema de recursividad)
  - Evaluación de expresiones

  - Operaciones de undo/redo (deshacer/rehacer)
  - Historial de navegación

- Esta estructura tiene tres operaciones básicas:
  - Push: agrega un elemento al tope
  - Pop: saca y regresa el elemento al tope
  - Peek: sólo regresa el valor del elemento del tope sin sacarlo



2 Implementación

- Python no define esta estructura en su biblioteca estándar
- Sin embargo es fácil de simular usando una lista
- Incluso el API de lista tiene métodos de pilas

```
pila = []
pila.append(1) # equivalente de push
pila.append(2)
pila.append(3)

tope = pila[-1] # equivalente de peek
print(tope)

tope = pila.pop()
print(tope)
```

```
print(pila)
```

```
3
3
[1, 2]
```

3 Implementación con clases

- Para tener las funcionalidades con las restricciones establecidas

```
class Pila():
    def __init__(self):
        self.interna = []

    def push(self, valor):
        self.interna.append(valor)

    def pop(self):
        if not self.interna:
            return None
        return self.interna.pop()

    def peek(self):
        if not self.interna:
            return None
        return self.interna[-1]

    def __repr__(self):
        return str(self.interna)

pila = Pila()
pila.push(1)
pila.push(2)
pila.push(3)
print(pila)
tope = pila.peek()
print(tope)
```

```
tope = pila.pop()
print(tope)
print(pila)
```

```
[1, 2, 3]
3
3
[1, 2]
```

4 Ejemplos

- Las pilas facilitan hacer cosas en orden inverso
- Por ejemplo, obtener una cadena al revés:

```
def voltear(cadena):
    pila = Pila()
    for c in cadena:
        pila.push(c)
    inversa = ''
    while pila.peek(): # si está vacía peek regresa None
        inversa += pila.pop()
    return inversa

s = voltear('hola mundo')
print(s)
```

```
odnum aloh
```