

## #3.4 Pilas, Colas y Listas Enlazadas

### Definición de Pila

Una pila (stack) es una estructura de datos que sigue el principio LIFO (Last In, First Out), es decir, el último elemento en entrar es el primero en salir. Las operaciones principales de una pila son:

push: agregar un elemento al tope de la pila.

pop: eliminar y retornar el elemento del tope de la pila.

Ejemplo de Pila Implementada con Listas en Python

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

    def esta_vacia(self):
        return self.items == []

    def apilar(self, item):
        self.items.append(item)

    def desapilar(self):
        return self.items.pop()

    def ver_tope(self):
        return self.items[-1]

    def tamaño(self):
        return len(self.items)

# Ejemplo de uso
pila = Pila()
pila.apilar(1)
pila.apilar(2)
pila.apilar(3)
print(pila.desapilar()) # Salida: 3
print(pila.ver_tope())  # Salida: 2
print(pila.tamaño())    # Salida: 2
```



## Definición de Cola

Una cola (queue) es una estructura de datos que sigue el principio FIFO (First In, First Out), es decir, el primer elemento en entrar es el primero en salir. Las operaciones principales de una cola son:

enqueue: agregar un elemento al final de la cola.

dequeue: eliminar y retornar el elemento al frente de la cola.

Ejemplo de Cola Implementada con Listas en Python

```
class Cola:
    def __init__(self):
        self.items = []

    def esta_vacia(self):
        return self.items == []

    def encolar(self, item):
        self.items.insert(0, item)

    def desencolar(self):
        return self.items.pop()

    def tamano(self):
        return len(self.items)

# Ejemplo de uso
cola = Cola()
cola.encolar(1)
cola.encolar(2)
cola.encolar(3)
print(cola.desencolar()) # Salida: 1
print(cola.tamano())    # Salida: 2
```

## Definición de Lista Enlazada

Una lista enlazada es una estructura de datos lineal en la que cada elemento es un nodo, y cada nodo contiene un valor y una referencia (o enlace) al siguiente nodo en la secuencia. Las operaciones básicas incluyen la inserción, eliminación y búsqueda de elementos.

### Ejemplo de Nodo de Lista Enlazada

```
class Nodo:
    def __init__(self, dato=None):
        self.dato = dato
        self.siguiente = None
```

## Implementar una Pila utilizando Listas Enlazadas

```
class Nodo:
    def __init__(self, dato=None):
        self.dato = dato
        self.siguiente = None

class PilaEnlazada:
    def __init__(self):
        self.tope = None


    def esta_vacia(self):
        return self.tope is None

    def apilar(self, item):
        nuevo_nodo = Nodo(item)
        nuevo_nodo.siguiente = self.tope
        self.tope = nuevo_nodo

    def desapilar(self):
        if self.esta_vacia():
            raise IndexError("La pila está vacía")
        dato = self.tope.dato
        self.tope = self.tope.siguiente
        return dato

    def ver_tope(self):
        if self.esta_vacia():
            raise IndexError("La pila está vacía")
        return self.tope.dato

# Ejemplo de uso
pila = PilaEnlazada()
pila.apilar(1)
pila.apilar(2)
pila.apilar(3)
print(pila.desapilar()) # Salida: 3
print(pila.ver_tope()) # Salida: 2
```



## Implementar una Cola utilizando Listas Enlazadas

```
class Nodo:
    def __init__(self, dato=None):
        self.dato = dato
        self.siguiente = None

class ColaEnlazada:
    def __init__(self):
        self.frente = None
        self.final = None

    def esta_vacia(self):
        return self.frente is None

    def encolar(self, item):
        nuevo_nodo = Nodo(item)
        if self.esta_vacia():
            self.frente = self.final = nuevo_nodo
        else:
            self.final.siguiente = nuevo_nodo
            self.final = nuevo_nodo

    def desencolar(self):
        if self.esta_vacia():
            raise IndexError("La cola está vacía")
        dato = self.frente.dato
        self.frente = self.frente.siguiente
        if self.frente is None:
            self.final = None
        return dato

# Ejemplo de uso
cola = ColaEnlazada()
cola.encolar(1)
cola.encolar(2)
cola.encolar(3)
print(cola.desencolar()) # Salida: 1
print(cola.desencolar()) # Salida: 2
```