**UNIVERSIDAD AMERICANA**

## **Facultad de Ingeniería y Arquitectura**



# **ALGORITMOS Y ESTRUCTURAS DE DATOS - GRUPO 2**



Código #1



# **Nombre:**

* Freddy Adrian peralta
* Jose Gabriel Cano Blandón
* Marcos Alessandro Lagos Rivera
* Axell Antonio Castillo Tapia

# **Docente**

## Silvia

*Managua, Nicaragua*

*26 de mayo del 2024*

# [Nodo.py](https://github.com/JGCB-2007/Trabajo-Jose-Cano/blob/main/Trabajo%20clase%2026.05.25/Ejercicio%201/Ejercicio%201/nodo.py)

Se define una clase Nodo.

El nodo contiene dos atributos:

* valor: almacena el dato del nodo.
* siguiente: referencia al siguiente nodo en la estructura (inicialmente None).

# [Cola.py](https://github.com/JGCB-2007/Trabajo-Jose-Cano/blob/main/Trabajo%20clase%2026.05.25/Ejercicio%201/Ejercicio%201/cola.py)

#### Clases involucradas:

##### Nodo

* Atributos:  
  + valor: almacena el dato del nodo.
  + siguiente: referencia al siguiente nodo (por defecto, None).

##### Cola

* Atributos:  
  + frente: referencia al primer nodo.
  + final: referencia al último nodo.
* Métodos:  
  + encolar(valor): Inserta un nuevo nodo al final.
  + desencolar(): Elimina el nodo del frente y retorna su valor.
  + esta\_vacia(): Verifica si la cola está vacía.
  + ver\_cola(): Retorna una lista con los valores de todos los nodos en la cola.

# [Pila.py](https://github.com/JGCB-2007/Trabajo-Jose-Cano/blob/main/Trabajo%20clase%2026.05.25/Ejercicio%201/Ejercicio%201/pila.py)

#### Atributos:

* cima: referencia al nodo que está en la cima de la pila (el último elemento insertado).

#### Métodos:

* \_\_init\_\_(self)  
   Inicializa una pila vacía, con la cima apuntando a None.
* apilar(self, valor)  
   Crea un nuevo nodo con el valor dado y lo coloca en la cima de la pila.
* desapilar(self)  
   Elimina el nodo que está en la cima de la pila y retorna su valor.  
   Retorna a None si la pila está vacía.
* esta\_vacia(self)  
   Retorna True si la pila no contiene nodos (es vacía), de lo contrario False.
* ver\_pila(self)  
   Recorre la pila desde la cima y retorna una lista con todos los valores almacenados, en orden de cima hacia abajo.

[Main.py](https://github.com/JGCB-2007/Trabajo-Jose-Cano/blob/main/Trabajo%20clase%2026.05.25/Ejercicio%201/Ejercicio%201/main.py)

Ejecuta un programa que recibe una lista de elementos separados por comas, los almacena en una cola, y luego separa los elementos según su posición (índices pares e impares).

* Los elementos en posición par se vuelven a encolar en la cola original.
* Los elementos en posición impar se apilan en una pila.

Finalmente, muestra los contenidos de la cola (con elementos de posición par) y de la pila (con elementos de posición impar).

#### Flujo de la función:

1. Entrada del usuario:  
    Solicita ingresar elementos separados por comas (letras o números).
2. Creación y llenado de la cola:  
    Los elementos ingresados se limpian (se remueven espacios) y se agregan a la cola.
3. Separación de elementos:  
   * Se recorre la cola desencolando un elemento a la vez.
   * Si el índice es par, el elemento se vuelve a encolar en la cola.
   * Si el índice es impar, el elemento se apila en la pila.
4. Salida:  
    Se imprime el contenido final de la cola y la pila.

#### Variables clave:

* cola\_original: instancia de la clase Cola donde se almacenan los elementos originales y pares.
* pila: instancia de la clase Pila donde se almacenan los elementos en posiciones impares.
* elementos: lista con los elementos ingresados por el usuario, limpios de espacios.
* tamaño: número total de elementos ingresados para controlar el ciclo.