

**Universidad Tecnológica de Panamá**  
**Facultad de Sistemas Computacionales**  
**Asignatura: Desarrollo Lógico y Algoritmo**  
**Taller Práctico1**

Profesor: Napoleón Ibarra

Valor: 100 puntos

Estudiante: Luigie Caballero, Jonathan Morales

Cédula:  
4-835-1868, 4-840-21

**Fecha Inicio: 27/10/2025 --> 4:10 PM**

**Fecha Entrega: 28/10/2025 --> 3:20 PM**

**Procedimiento:**

1. De manera individual o en grupo de trabajo de 2 personas, realizar la asignación. Utilizando la herramienta Internet, investigue, desarrolle los conceptos solicitados.
2. Entregar el trabajo en formato digital(Parte I, II, III, IV) en PDF en la plataforma.

**Criterios de Evaluación:**

Criterios	Puntos (Mínimo 1, Máximo 5)	Porcentaje
Sustentación	1 - 5	15 %
Puntualidad	1 - 5	15 %
Desarrollo	1 - 5	70 %

# PROCEDIMIENTO DE BÚSQUEDA Y ORDENAMIENTO DE UN ARREGLO

- **Búsqueda secuencial (lineal):** recorrer el arreglo elemento por elemento hasta encontrar el valor o terminar. Es simple,  $O(n)$  en peor caso.
- **Push down (si en la rúbrica se refiere a push down en estructuras):** puede referirse a la operación en pila (stack) donde se "empuja" (push) un elemento y se "apila" en el tope — LIFO. Si se referían a ordenamiento, puede que el profesor quisiera push-down en estructuras de heap durante el algoritmo heapify (heapsort): empujar hacia abajo un nodo para mantener propiedad de heap.

## Búsqueda Secuencial

# Búsqueda secuencial en una lista

# Ingresar tamaño del arreglo

```
n = int(input("Ingrese la cantidad de elementos del arreglo: "))
```

# Crear e ingresar los elementos

```
arreglo = []
```

```
for i in range(n):
```

```
    valor = int(input(f"Ingrese el elemento {i + 1}: "))
    arreglo.append(valor)
```

# Valor a buscar

```
valor_buscado = int(input("Ingrese el valor que desea buscar: "))
```

# Búsqueda secuencial

```
encontrado = False
```

```
for i in range(n):
```

```
    if arreglo[i] == valor_buscado:
        print(f"Valor encontrado en la posición {i + 1}")
        encontrado = True
        break
```

```
if not encontrado:
```

```
    print("El valor no se encuentra en el arreglo.")
```

## Sintaxis

**n = int(input(...)).**

Solicita al usuario un número entero que indica la cantidad de elementos.

**arreglo = []**

Declara una lista vacía donde se almacenarán los valores.

**for i in range(n):**

Ciclo que repite una acción n veces (desde 0 hasta n-1).

**arreglo.append(valor)**

Agrega (inserta) el valor al final de la lista.

**if arreglo[i] == valor\_buscado:**

Condición: compara el elemento actual con el valor buscado.

**break**

Interrumpe el ciclo al encontrar el valor.

**if not encontrado:**

Evalúa si la variable booleana encontrado sigue siendo False.

**print().**

Muestra información en pantalla.

## Push Down

# Implementación sencilla de una pila (Push Down)

pila = []

maximo = 5

def mostrar\_menu():

print("\n=== MENÚ DE PILA ===")

print("1. PUSH (Agregar elemento)")

print("2. POP (Eliminar elemento)")

print("3. Mostrar pila")

print("4. Salir")

while True:

mostrar\_menu()

opcion = input("Seleccione una opción: ")

if opcion == "1":

if len(pila) == maximo:

print("La pila está llena (overflow).")

else:

elemento = input("Ingrese el elemento a

agregar: ")

pila.append(elemento)

print("Elemento agregado correctamente.")

elif opcion == "2":

if len(pila) == 0:

print("La pila está vacía (underflow).")

else:

eliminado = pila.pop()

print(f"Elemento eliminado: {eliminado}")

elif opcion == "3":

if len(pila) == 0:

print("La pila está vacía.")

else:

print("Elementos en la pila (de arriba hacia abajo):")

for elem in reversed(pila):

print(elem)

elif opcion == "4":

print("Programa finalizado.")

break

else:

print("Opción no válida. Intente nuevamente.")

## Sintaxis

**pila = []**

Crea una lista vacía que actuará como la pila.

**maximo = 5**

Define el tamaño máximo de la pila.

**def mostrar\_menu():**

Define una función que muestra el menú principal.

**while True:**

Bucle infinito que se repite hasta usar break.

**input()**

Permite ingresar datos por teclado.

**len(pila)**

Retorna el número de elementos actuales en la pila.

**pila.append(elemento)**

Agrega (PUSH) un elemento al final de la pila.

**pila.pop()**

Elimina (POP) el último elemento agregado.

**reversed(pila)**

Permite recorrer la pila desde el último elemento hacia el primero.

**break**

Sale del ciclo while cuando el usuario elige salir.

**elif, else**

Se usan para controlar las diferentes opciones del menú.

## Búsqueda Secuencial

Algoritmo BusquedaSecuencial

// Definición de variables

Definir n, valor\_buscado, i Como Entero;

Definir arreglo Como Entero;

Definir encontrado Como Logico;

Dimension arreglo[100]; // Se define un tamaño máximo para el arreglo, aunque PSeInt puede manejar tamaños dinámicos

// 1. Ingresar la cantidad de elementos

Escribir "Ingrese la cantidad de elementos del arreglo:";

Leer n;

Leer n;

// Validación básica del tamaño (opcional, pero útil)

Si n <= 0 O n > 100 Entonces

Escribir "Cantidad no válida. El programa

finalizará.";

FinAlgoritmo;

FinSi

// 2. Llenar el arreglo

Para i = 1 Hasta n Con Paso 1 Hacer

Escribir "Ingrese el elemento ", i, ": ";

Leer arreglo[i];

FinPara

// 3. Ingresar el valor a buscar

Escribir "Ingrese el valor que desea buscar: ";

Leer valor\_buscado;

// 4. Búsqueda Secuencial

encontrado = Falso;

i = 1; // Inicializamos el índice para el ciclo Mientras

Mientras i <= n Y NO encontrado Hacer

Si arreglo[i] = valor\_buscado Entonces

Escribir "Valor encontrado en la posición ", i;

encontrado = Verdadero; // El 'break' se simula

cambiando la condición del Mientras

FinSi

i = i + 1; // Avanzamos al siguiente elemento

FinMientras

// 5. Mostrar resultado si no se encontró

Si NO encontrado Entonces

Escribir "El valor no se encuentra en el arreglo.";

FinSi

FinAlgoritmo

## Push Down

// Subproceso para no repetir el código del menú

SubProceso MostrarMenu

Escribir " " // Para dar un espacio

Escribir "=== MENÚ DE PILA ==="

Escribir "1. PUSH (Agregar elemento)"

Escribir "2. POP (Eliminar elemento)"

Escribir "3. Mostrar pila"

Escribir "4. Salir"

Escribir Sin Saltar "Seleccione una opción: "

FinSubProceso

## Algoritmo PilaEstatica

```
// 1. Definición de variables y constantes
Definir maximo Como Entero
Definir pila Como Caracter // Asumimos que
guardará texto (como el input de Python)
Definir tope Como Entero
Definir opcion Como Caracter
Definir elemento, eliminado Como Caracter

// 2. Inicialización
maximo <- 5
Dimensión pila[maximo] // Definimos el tamaño
del arreglo
tope <- 0 // El 'tope' en 0 indica que la pila está
vacía

// 3. Ciclo principal (equivale al 'while True')
Repetir
    MostrarMenu() // Llamamos al subproceso del
menú
    Leer opcion

// 4. Estructura 'Segun' (equivale al if/elif/else
de opciones)
Segun opcion Hacer
    '1': // PUSH
        Si tope = maximo Entonces
            Escribir "La pila está llena (overflow)."
        Sino
            Escribir Sin Saltar "Ingresa el elemento
a agregar: "
            Leer elemento
            // Apilamos: primero movemos el tope,
luego guardamos
            tope <- tope + 1
            pila[tope] <- elemento
            Escribir "Elemento agregado
correctamente."
        FinSi

    '2': // POP
        Si tope = 0 Entonces
            Escribir "La pila está vacía (underflow)."
        Sino
            // Desapilamos: obtenemos el elemento y
luego movemos el tope
            eliminado <- pila[tope]
            tope <- tope - 1
            Escribir "Elemento eliminado: ", eliminado
        FinSi

    '3': // MOSTRAR PILA
        Si tope = 0 Entonces
            Escribir "La pila está vacía."
        Sino
            Escribir "Elementos en la pila (de arriba hacia
abajo):"
            // Recorremos desde el tope (arriba) hasta 1.
            Definir i Como Entero
            Para i <- tope Hasta 1 Con Paso -1 Hacer
                Escribir " ", pila[i]
            FinPara
        FinSi

    '4':
        Escribir "Programa finalizado."

De Otro Modo:
    Escribir "Opción no válida. Intente
nuevamente."

FinSegun

Hasta Que opcion = '4' // Condición de salida del
bucle 'Repetir'

FinAlgoritmo
```