Lógica de programación I

Juan Pablo Restrepo Uribe

Ing. Biomedico

MSc. Automatización y Control Industrial

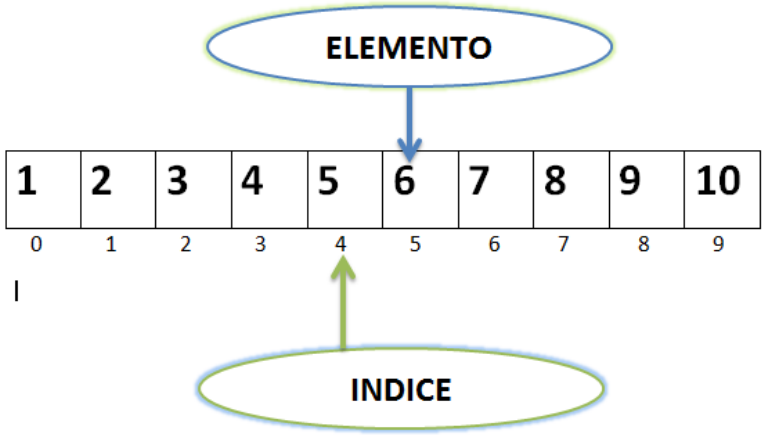
jprestrepo@correo.iue.edu.co

2023-2

Institución Universitaria de Envigado

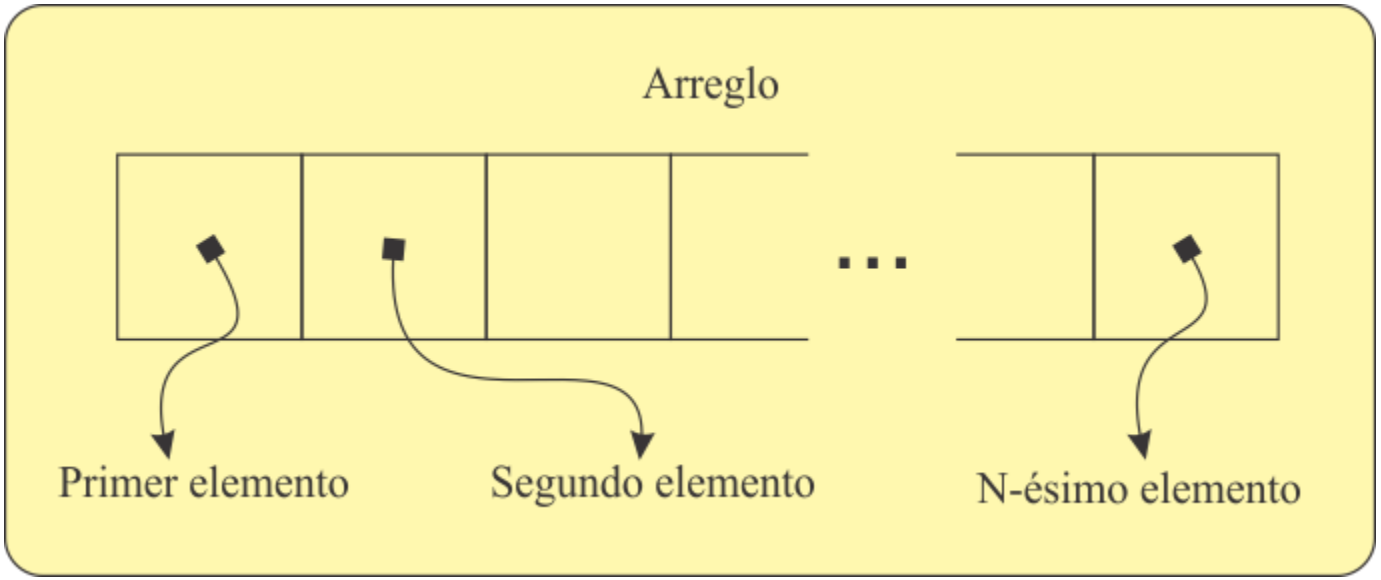
Arreglos unidimensionales

Un arreglo unidimensional es un tipo de datos estructurado que está formado por una colección finita y ordenada de **datos del mismo tipo**. Los datos que se guarden en los arreglos todos deben ser del mismo tipo.



Arreglos unidimensionales

El tipo de acceso a los arreglos unidimensionales es el acceso directo, es decir, podemos acceder a cualquier elemento del arreglo sin tener que consultar a elementos anteriores o posteriores, esto mediante el uso de un índice para cada elemento del arreglo que nos da su posición relativa.



Arreglos unidimensionales

Python no proporciona formas naturales para crear arreglos, estos deben ser creados mediante listas. Sin embargo, existen librerías que permiten la creación de arreglos

```
n = int(input(u"Ingrese el tamaño del arreglo"))
m = int(input(u"Ingrese el número de múltiplos"))
A = []
for i in range (0,n):
 A.append(i*m)
print A
```

Recorrido (Acceso secuencial)

Se puede acceder a cada elemento de un vector para introducir datos (leer) en él o bien para visualizar su contenido (escribir). A la operación de efectuar una acción general sobre todos los elementos de un vector se le denomina recorrido del vector.

```
Haga para i= 1 hasta n
        escribir("Introduzca el elemento" ,i, "del vector F:")
        leer(F[i])
Fin de Para
```

Lectura/Escritura

La lectura/escritura de datos en arreglos normalmente se realiza con estructuras repetitivas (usando un recorrido secuencial).

```
import numpy as np
a = np.array([1,2,3,4,5,6])
print(a)
print(a[2])
```

Asignación

La asignación de valores a un elemento del vector se realizará con la instrucción de asignación:

```
import numpy as np
a = np.array([1,2,3,4,5,6])
print(a)
print(a[2])

a[2] = 10
print(a)
print(a[2])
```

Insertar Elemento a un Vector

Consiste en introducir un elemento en el interior de un vector ordenado. En este caso se necesita un desplazamiento previo hacia abajo para colocar el nuevo elemento en su posición relativa.

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
nuevo_valor = 10
posicion = 2
nuevo_arr = np.insert(arr, posicion, nuevo_valor)
print(nuevo_arr)
```

Insertar Elemento a un Vector

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])
nuevos_valores = [10, 11, 12]
posiciones = [2, 4, 0]

nuevo_arr = np.insert(arr, posiciones, nuevos_valores)

print(nuevo_arr)
```

Borrar un elemento de un Vector

La operación de borrar el último elemento de un vector no representa ningún problema. El borrado de un elemento del interior del vector provoca el movimiento hacia arriba de los elementos inferiores a él para reorganizar el vector.

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
posicion = 2
nuevo_arr = np.delete(arr, posicion)
print(nuevo_arr)
```

Numpy

```
import numpy as np
a = np.empty((1,10))
print(a)
```

```
import numpy as np
a = np.ones((1,10))
print(a)
```

```
import numpy as np
a = np.zeros((1,10))
print(a)
```

```
import numpy as np
a = np.random.random((1,10))
print(a)
```

```
import numpy as np
a = np.random.randint(5,10, size = (1,10))
print(a)
```

Ejercicios

Escribe una función que tome un array NumPy de números enteros y calcule la suma de todos los elementos pares en el array.

Escribe una función que cuente cuántos elementos en un array NumPy son mayores que un valor dado.

Escribe una función que tome un array NumPy y cambie todos los valores negativos por cero.

Escribe una función que calcule el promedio de los valores en un array NumPy. Si el valor es negativo, no debe ser incluido en el cálculo del promedio.

Burbuja

```
Procedimiento ordenamientoBurbuja(arr)
    n = longitud(arr)
    Para i desde 0 hasta n-1
        intercambiado = Falso
        Para j desde 0 hasta n-1-i
            Si arr[j] > arr[j+1] Entonces
                intercambiar arr[j] y arr[j+1]
                intercambiado = Verdadero
            Fin Si
        Fin Para
        Si intercambiado = Falso Entonces
            Salir del bucle
        Fin Si
    Fin Para
Fin Procedimiento
```

Inserción

```
Procedimiento ordenamientoInsercion(arr)
    n = longitud(arr)
    Para i desde 1 hasta n-1
        valor = arr[i]
        j = i - 1
        Mientras j >= 0 y arr[j] > valor
            arr[j + 1] = arr[j]
            j = j - 1
        Fin Mientras
        arr[j + 1] = valor
    Fin Para
Fin Procedimiento
```

Ordenamiento por Selección

```
Procedimiento ordenamientoSeleccion(arr)
    n = longitud(arr)
    Para i desde 0 hasta n-1
        indiceMinimo = i
        Para j desde i+1 hasta n-1
            Si arr[j] < arr[indiceMinimo] Entonces
                indiceMinimo = j
            Fin Si
        Fin Para
        Si i != indiceMinimo Entonces
            Intercambiar arr[i] y arr[indiceMinimo]
        Fin Si
    Fin Para
Fin Procedimiento
```