

# Programación de Algoritmos

PGY1121

Escuela de Informática y Telecomunicaciones

**DuocUC**



ESCUELA DE  
INFORMÁTICA Y  
TELECOMUNICACIONES





# Experiencia de Aprendizaje N° 4

## Clase N° 2

Arreglos

## Objetivos de la sesión

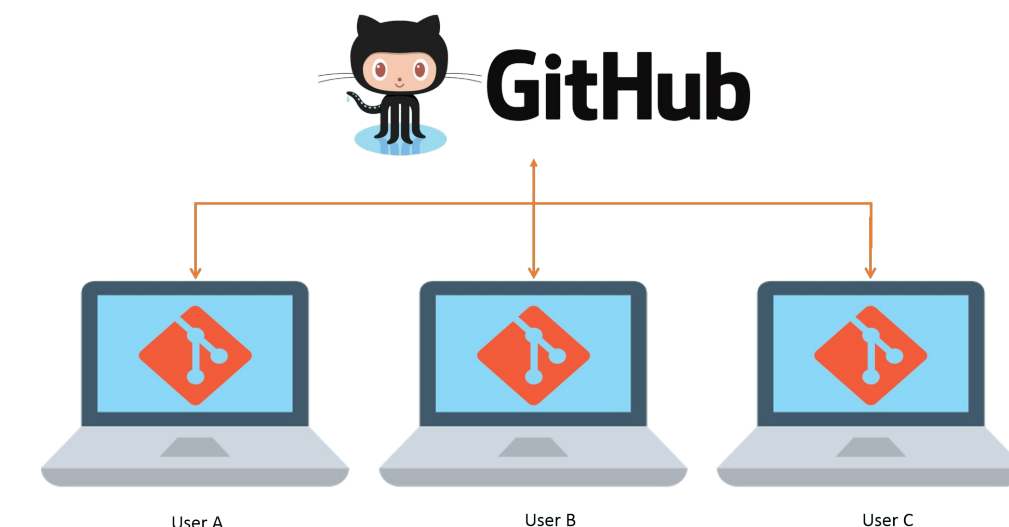
- Utilizar GitHub para el respaldo de los códigos desarrollados en Python
- Utilizar diferentes tipos de arreglos bidimensionales para almacenar datos según los requerimientos solicitados



# Git y GitHub

**Git** es un proyecto de código abierto, el cual permite llevar un control de versiones de un proyecto. Es el sistema base de versiones. Se usa localmente.

**GitHub** es una herramienta que trabaja con Git, y agrega la centralización de los proyectos en nube, interfaces muy sencillas de ocupar, junto con el control de versiones centralizado, permitiendo a varios desarrolladores unificar sus códigos en la nube de GitHub, mejorando la comunicación, el desarrollo, los respaldos y la unificación de códigos (todo en nube).





Revisa la carpeta **Instructivo GitHub** de la semana 2, para:

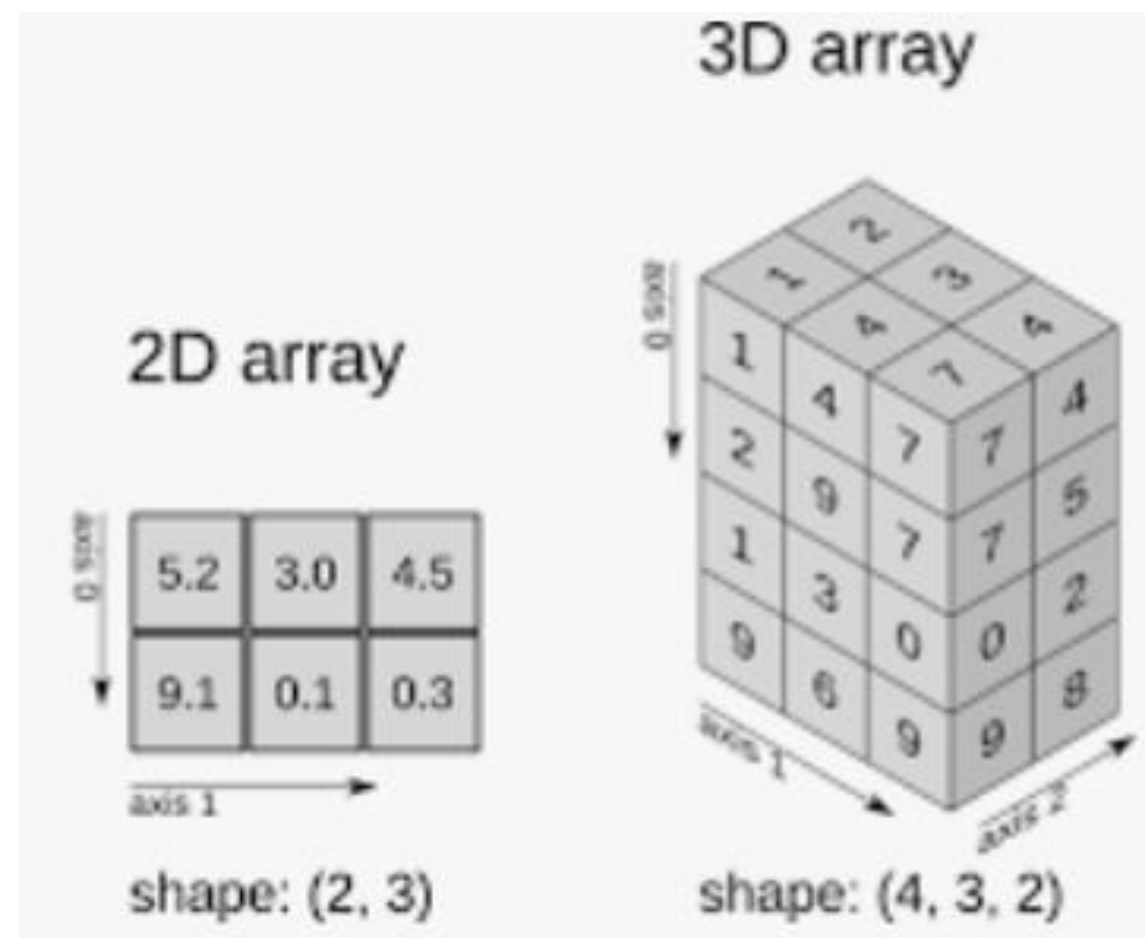
- Crear tu cuenta.
- Subir los archivos de Python.
- Compartir información con tus pares y docente.



Link apoyo a más información de GitHub:

<https://programarfacil.com/blog/arduino-blog/git-y-github/>

# Arreglo Bidimensional



En la experiencia 4 semana 1, se proporcionó la descripción de los arreglos.

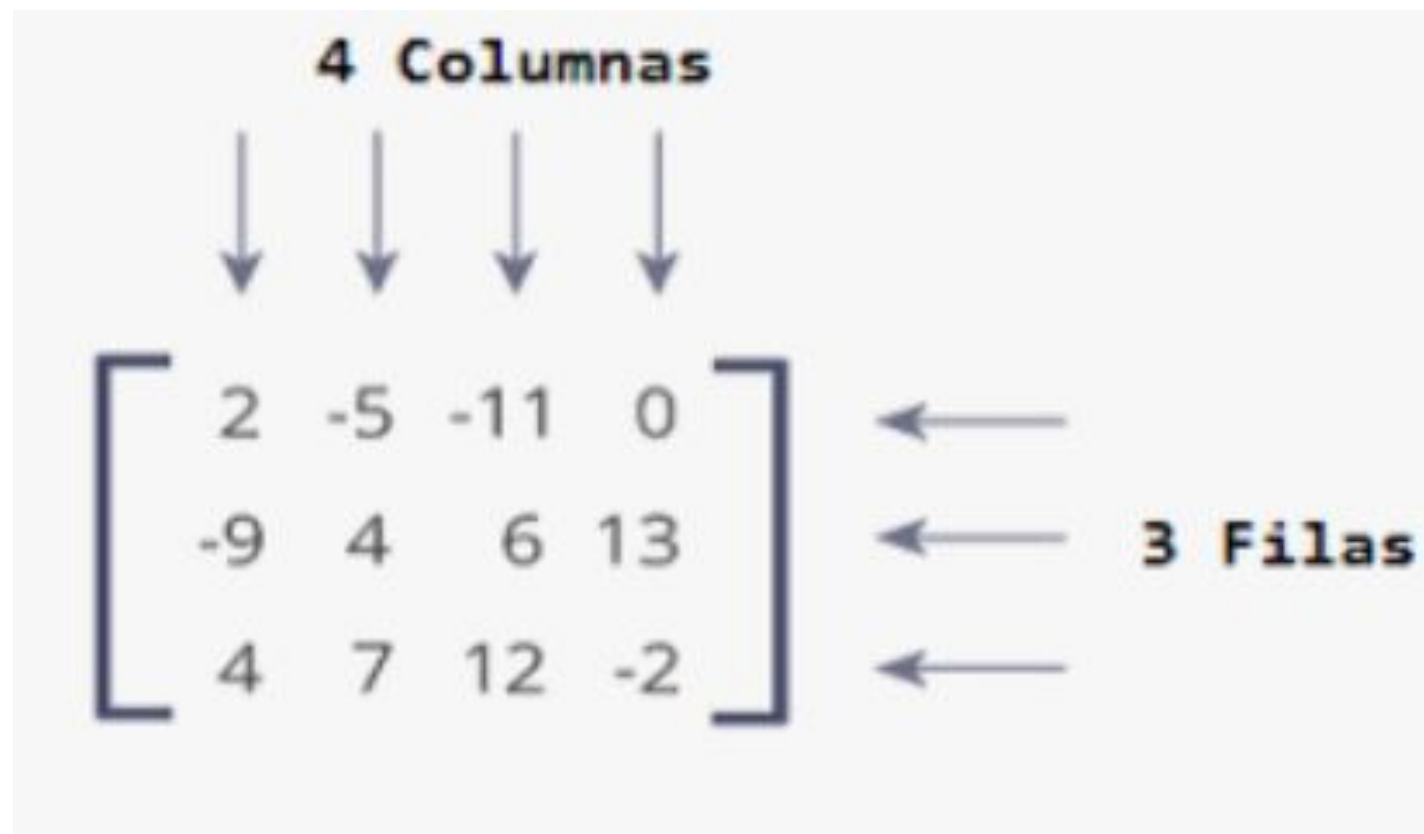
## Para recordar:

Son un conjunto de datos que se almacenan temporalmente, además, cumplen ciertas características, tales como:

- Colección finita.
- Homogénea.
- Elementos ordenados.

Hoy desarrollaremos ejercicios con arreglos con 2 o más dimensiones, las cuales son llamadas **“Matriz”**

# Arreglo Bidimensional



The diagram shows a 3x4 2D array represented as a matrix. Above the matrix, the text "4 Columns" is centered, with four downward-pointing arrows indicating the columns. To the right of the matrix, the text "3 Filas" is centered, with three leftward-pointing arrows indicating the rows. The matrix contains the following values:

2	-5	-11	0
-9	4	6	13
4	7	12	-2

Una matriz está compuesta por filas y columnas, como muestra el ejemplo.

Cada celda o casillero contiene dos índices, el primero es la fila y luego la columna, las cuales son las coordenadas para posicionado el elemento.

**Responde:**

- ¿Qué largo tiene este arreglo?
- ¿Qué tipo de datos posee el arreglo?



## Declarar un arreglo de dos dimensiones (matriz)

Python no posee estructuras propias para definir matrices, como se conocen en los lenguajes tradicionales, sino más bien son listas de listas.

### Caso 1: Tradicional

```
import numpy as np
matriz = np.array([[0, 1, 2], [3, 4, 5]])
for f in range(2):
    for c in range(3):
        print(matriz[f][c])
```

```
0
1
2
3
4
5
```

### Caso 2: Usando listas

```
import numpy as np
lista = [[1,2,3],
         [4,5,6]]

matriz = np.array(lista)
matriz

array([[1, 2, 3],
       [4, 5, 6]])
```

## Operaciones con un arreglo de dos dimensiones (matriz)

### Mostrar un elemento de la matriz

#### Caso 1:

```
import numpy as np
matriz = np.array([[0, 1, 2], [3, 4, 5]])
for f in range(2):
    for c in range(3):
        print(matriz[f][c])
print("")
print(matriz[1,1])
```

0  
1  
2  
3  
4  
5  
  
4

#### Caso 2:

```
import numpy as np
lista = [[1,2,3],
        [4,5,6]]

matriz = np.array(lista)
print(matriz[0][1])
```

2



## Operaciones con un arreglo de dos dimensiones (matriz)

Mostrar un elemento de la matriz

```
import numpy as np
lista = [[1,2,3],
         [4,5,6]]

matriz = np.array(lista)
print(matriz[1,1])

print(matriz[:,2])

print(matriz[0,:])

print(matriz[0,::-1])
```



5



[3 6]



[1 2 3]



[3 2 1]

## Operaciones con un arreglo de dos dimensiones (matriz)

### Generar matriz con ceros

```
import numpy as np
matriz = np.zeros((3, 3))
print(matriz)
```

```
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
```

### Generar matriz con unos

```
import numpy as np
matriz = np.ones((3, 3))
print(matriz)
```

```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

### Generar matriz con Diagonal principal con 1

```
import numpy as np
matriz = np.diag([1,1,1])
print(matriz)
```

```
[[1 0 0]
 [0 1 0]
 [0 0 1]]
```



## Operaciones con un arreglo de dos dimensiones (matriz)

### Sumar todos los elementos

```
import numpy as np
lista = [[1,2,3],
         [4,5,6]]

matriz = np.array(lista)

print(matriz.sum())
```

21

### Sumar elementos por fila

```
import numpy as np
lista = [[1,2,3],
         [4,5,6]]

matriz = np.array(lista)

print("Suma elementos fila 0: ", matriz[0,:].sum())
print("Suma elementos fila 1: ", matriz[1,:].sum())
```

Suma elementos fila 0: 6  
Suma elementos fila 1: 15

## Operaciones con un arreglo de dos dimensiones (matriz)

Responder, ¿qué realizan las siguientes funciones?

```
import numpy as np  
matriz = np.array([[0, 1, 2], [3, 4, 5]])  
  
matriz.ndim
```



2

```
import numpy as np  
matriz = np.array([[0, 1, 2], [3, 4, 5]])  
  
matriz.shape
```



(2, 3)

```
import numpy as np  
matriz = np.array([[0, 1, 2], [3, 4, 5]])  
  
matriz.size
```



6



## Operaciones con un arreglo de dos dimensiones (matriz)

### Concatenar dos matrices

```
import numpy as np
m1 = np.array([[1,2,3],
               [4,5,6]])
m2 = np.array([[10,20,30],[40,50,60]])
np.concatenate((m1,m2), axis=0)
```

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [10, 20, 30],
       [40, 50, 60]])
```

**PGY1121**

**Ver video**

<https://www.youtube.com/watch?v=97o5ckUHDmo>





Comenta con tus compañeros y docente sobre las operaciones con arreglos.

Busca y expone otras funciones y operaciones con arreglos de dos o más dimensiones.

1. Crear un arreglo de dos dimensiones de tamaño 3 x 3, con elementos aleatorios de números enteros del 0 al 100.
2. Utilice las siguientes funciones en el arreglo creado en el punto 1
  - Promedio de los elementos.
  - Suma de los elementos.
  - Mostrar el elemento mayor.
  - Mostrar el elemento menor.
  - Mostrar sólo los elementos de la diagonal principal.
3. Crear un arreglo de dos dimensiones de 3 x 3 con números ceros, excepto la diagonal principal que debe contener en el mismo orden los siguientes elementos 1, 2 y 3.

**Recuerda subir los archivos a GitHub**

**PGY1121**

# **Guía de Ejercicios**

Revisa los ejercicios de la Experiencia 4 Semana 2.



# Programación de Algoritmos

PGY1121

Escuela de Informática y Telecomunicaciones

**DuocUC**



ESCUELA DE  
INFORMÁTICA Y  
TELECOMUNICACIONES