



# Estud-IA

## Introducción a la Lógica de Programación y Algoritmos con Python

Miguel Angel Carrillo - 2025

 [miguelcmo](#)

Apoya:



Organiza:



#ciudadanosdigitales



# SESIÓN 1 - VIRTUAL (4 horas): Introducción a Python y Primeros Pasos

## Objetivos:

- Configurar el entorno de desarrollo
- ¿Como se ejecutan los programas?
- ¿Qué es La Terminal?
- Lógica y Pseudocódigo
- Introducción a Python
- Comprender la sintaxis básica de Python
- Trabajar con variables y tipos de datos
- Realizar operaciones básicas



# Configurar el entorno de desarrollo – Instalación de Python


<https://www.python.org/downloads/>





## Configurar el entorno de desarrollo – Instalación de VSCode

<https://code.visualstudio.com>

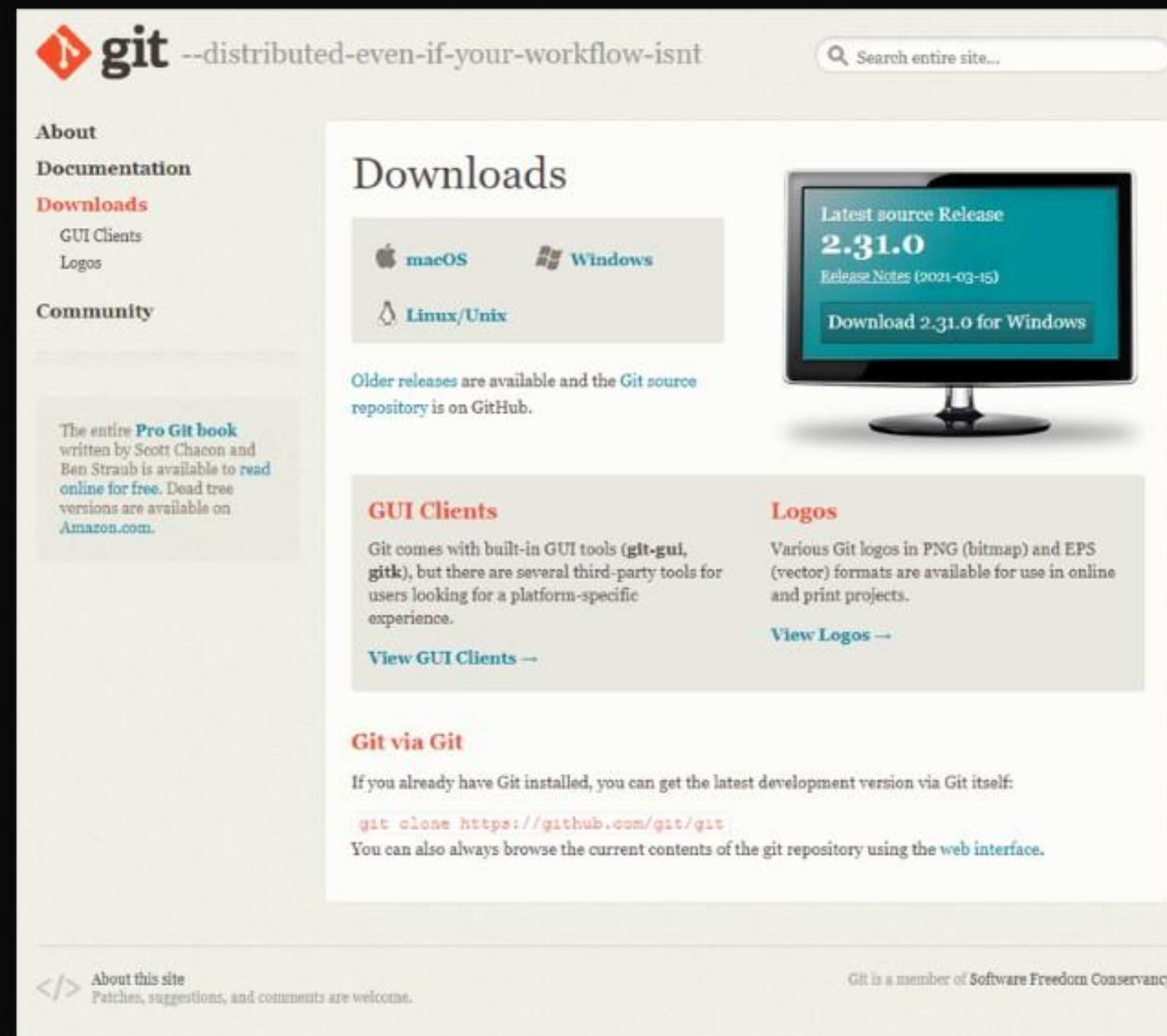


```
~/Downloads/AlertBox.js []  
  
AlertBox.js x  
  
export default function AlertBox() {  
  const props = {  
    visible: true,  
    colour: 'neutral',  
    reason: (  
      <span>  
        Warning:  
        VS Code is cool.  
      </span>  
    ),  
  };  
  return <AlertBox key="box" {...props} />;  
}
```

Ln 7, Col 25 Spaces: 4 UTF-8 LF JavaScript React Prettier

# Configurar el entorno de desarrollo – Instalación de Git

<https://git-scm.com/install/windows>



The screenshot shows the Git website's 'Downloads' page. The header includes the Git logo and the tagline '--distributed-even-if-your-workflow-isnt', along with a search bar. The left sidebar contains links for 'About', 'Documentation', 'Downloads' (highlighted), 'GUI Clients', 'Logos', and 'Community'. Below the sidebar, there is a section about the 'Pro Git book'. The main content area features a 'Downloads' section with links for macOS, Windows, and Linux/Unix. A large monitor graphic displays the 'Latest source Release 2.31.0' and a 'Download 2.31.0 for Windows' button. Below this, there are sections for 'GUI Clients' and 'Logos'. At the bottom, a 'Git via Git' section provides instructions on how to install Git using the command line or the web interface. The footer includes a link to 'About this site' and a statement that Git is a member of the Software Freedom Conservancy.

git --distributed-even-if-your-workflow-isnt

Search entire site...

About  
Documentation  
**Downloads**  
GUI Clients  
Logos  
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

## Downloads

Older releases are available and the Git source repository is on GitHub.

Latest source Release  
**2.31.0**  
Release Notes (2021-03-15)  
Download 2.31.0 for Windows

### GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

### Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

### Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).

</> About this site  
Patches, suggestions, and comments are welcome.

Git is a member of Software Freedom Conservancy

## ¿Como se ejecutan los programas?



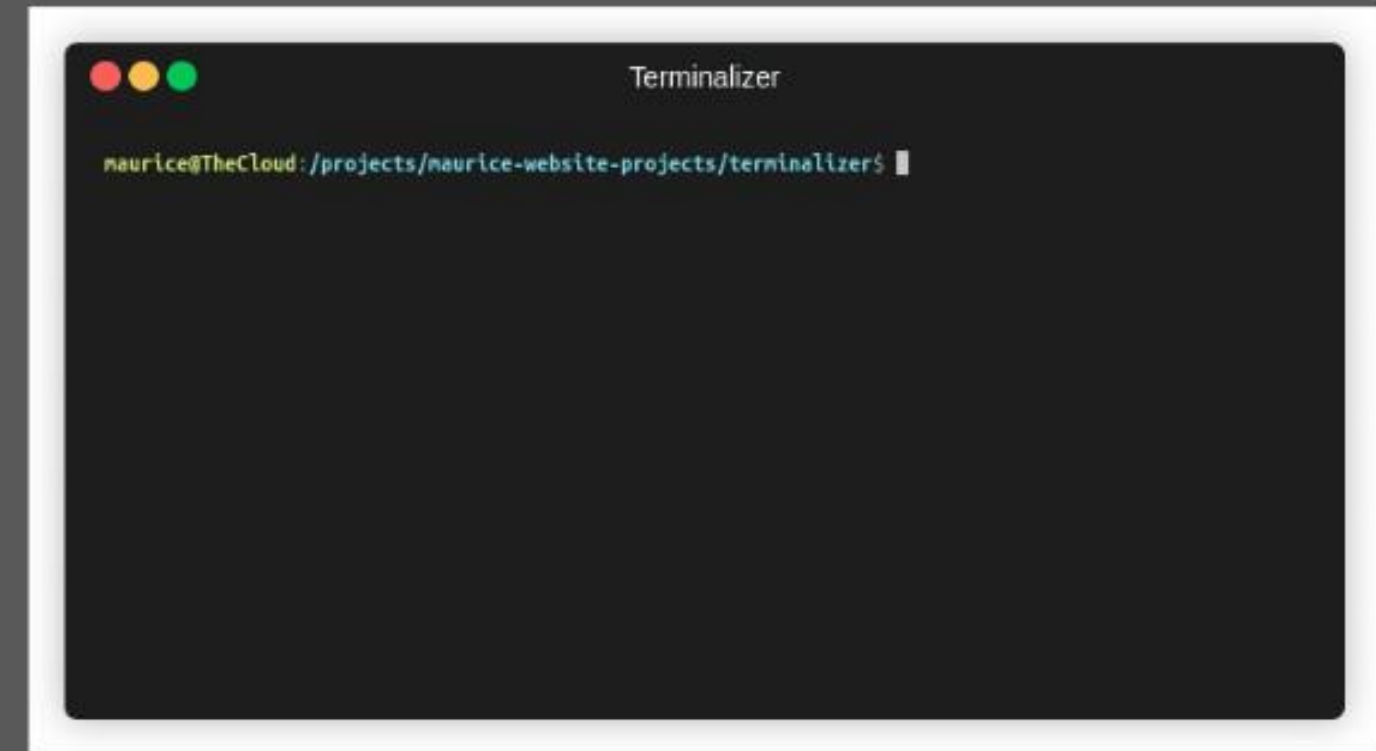
#ciudadanosdigitales



# ¿Qué es La terminal?

La terminal es una herramienta que permite interactuar directamente con el sistema operativo mediante comandos escritos, en lugar de usar ventanas, botones o menús gráficos. En términos prácticos, es una interfaz de línea de comandos o CLI donde puedes:

- Ejecutar programas.
- Navegar por carpetas y administrar archivos.
- Instalar, actualizar o configurar software.
- Automatizar tareas repetitivas.
- Consultar información del sistema.



Es muy utilizada por desarrolladores, administradores de sistemas y técnicos porque ofrece control preciso, rapidez y acceso a funciones avanzadas que a veces no están disponibles en las interfaces gráficas, alguno ejemplos son:

- **Windows:** PowerShell o Símbolo del Sistema.
- **Linux:** Bash (la más común).
- **macOS:** Terminal con shell Bash o Zsh.



# Lógica de programación

La **lógica de programación** es la capacidad de estructurar ideas y pasos de manera ordenada para resolver un problema mediante un programa. No depende de un lenguaje específico; es la forma de pensar que permite construir soluciones claras, coherentes y ejecutables por un computador.

## Qué implica la lógica de programación

### 1. Entender el problema

Identificar qué se necesita, qué datos entran y qué resultados deben salir.

### 2. Dividir el problema en pasos

Descomponerlo en tareas pequeñas y manejables (pensamiento algorítmico).

### 3. Establecer el orden correcto de las acciones

Decidir qué va primero, qué depende de qué, y cómo fluye la información.

### 4. Tomar decisiones

Usar estructuras como condiciones (si... entonces...) para manejar diferentes escenarios.

### 5. Repetir procesos cuando sea necesario

Usar ciclos para tareas repetitivas (mientras, para, etc.).

### 6. Traducir la solución a un algoritmo

Representarlo con pseudocódigo, diagramas de flujo o directamente en un lenguaje.



# Pseudocódigo

El **pseudocódigo** es una forma de escribir los pasos de un algoritmo usando un lenguaje **similar al español**, claro y estructurado, sin seguir estrictamente la sintaxis de un lenguaje de programación real.

Su propósito es **explicar la lógica** de un programa de manera fácil de entender, tanto para humanos como para luego traducirlo a código.

## Características

- ☐ No pertenece a ningún lenguaje de programación.
- ☐ Usa palabras clave comunes: *INICIO*, *LEER*, *SI*, *ENTONCES*, *MIENTRAS*, *MOSTRAR*, etc.
- ☐ Se centra en **qué se hace**, no en detalles técnicos como tipos de datos, llaves o paréntesis.
- ☐ Sirve como puente entre la lógica y el código final.

```
INICIO
  LEER numero
  SI (numero MOD 2 = 0) ENTONCES
    MOSTRAR "El número es par"
  SINO
    MOSTRAR "El número es impar"
  FIN_SI
FIN
```



# Introducción a Python

Python es un lenguaje de programación **interpretado, moderno y de propósito general**. Se destaca por su sintaxis sencilla, legible y muy cercana al lenguaje humano, lo que lo convierte en una excelente opción tanto para principiantes como para profesionales.

Hoy es uno de los lenguajes más utilizados en el mundo, especialmente en áreas como desarrollo web, automatización, análisis de datos, inteligencia artificial, scripting, ciberseguridad y más.

## Historia de Python

**Creado por:** *Guido van Rossum*, un programador neerlandés.

**Año de inicio del proyecto:** Finales de 1989.

**Primera versión pública:** Python 0.9.0 en 1991.





# Python - ¿Por qué se creó?

Guido buscaba un lenguaje que fuera:

- Más fácil de leer.
- Más rápido de escribir.
- Más flexible y menos complejo que C o Perl.

Su objetivo era ofrecer una sintaxis limpia y elegante que permitiera enfocarse en la lógica, no en reglas complicadas de lenguaje.

## Evolución rápida

**Python 2 (2000):** amplió el uso global del lenguaje.

**Python 3 (2008):** versión modernizada, más limpia y potente.

(Hoy es el estándar, mientras que Python 2 quedó obsoleto en 2020).

## Popularidad actual

Desde 2018, Python ha sido uno de los lenguajes más usados mundialmente por su facilidad y potencia, impulsado especialmente por la ciencia de datos, el machine learning y la automatización.

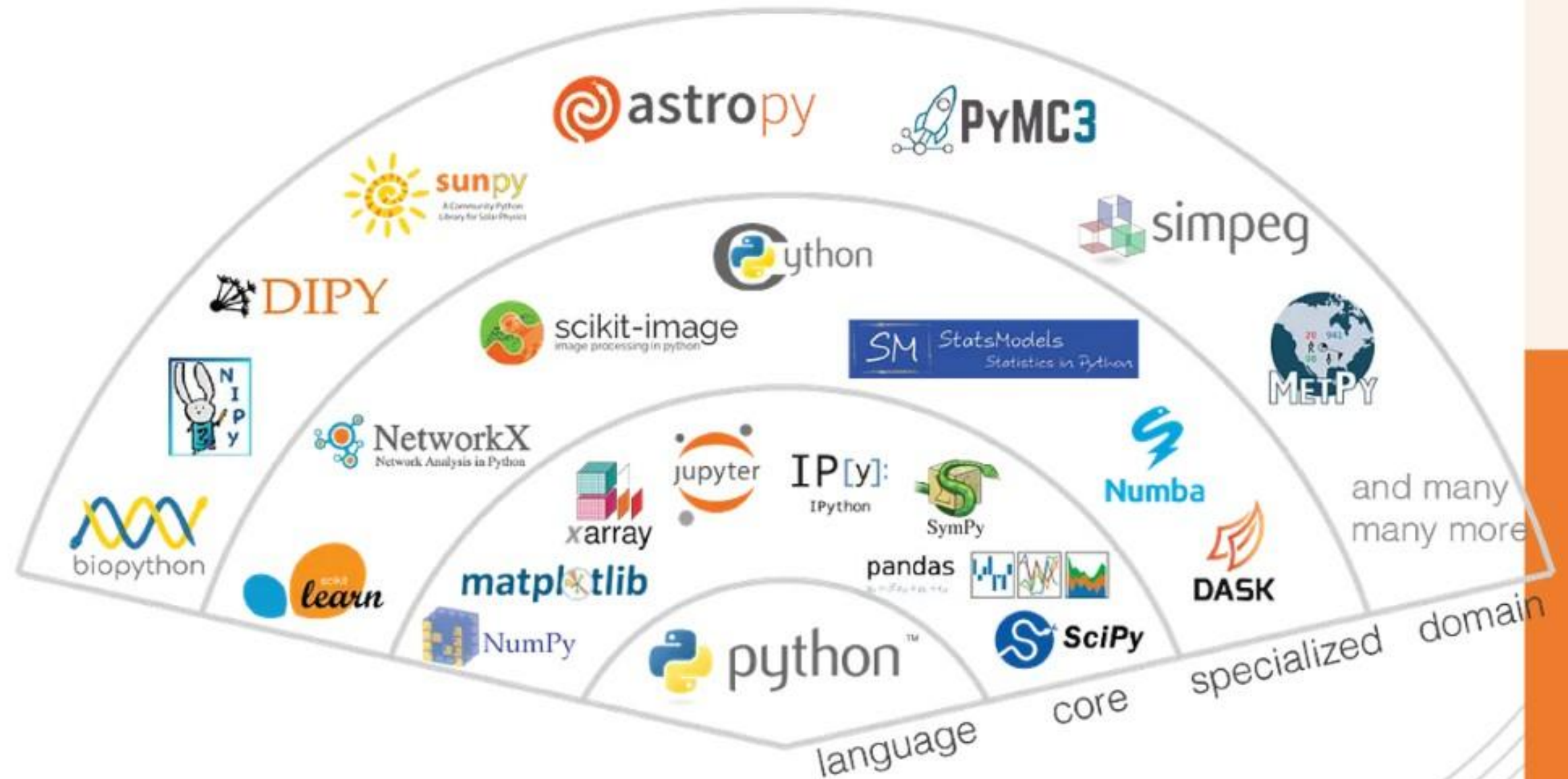
#ciudadanosdigitales





# Características principales de Python

1. Legible y sencillo
2. Interpretado
3. Multiparadigma
4. Multiplataforma
5. Extenso ecosistema de librerías
6. Comunidad enorme
7. Usos principales



<https://medium.com/@instailacademy/understanding-the-python-ecosystem-a-comprehensive-guide-1fe78b7c6440>



# Interprete de Python - REPL

**REPL** significa **Read–Eval–Print Loop**, y describe un entorno interactivo donde escribes código línea por línea y recibes una respuesta inmediata. En español, se entiende como: "**Leer – Evaluar – Imprimir – Repetir**"

## ¿Qué hace exactamente un REPL?

### **Read (Leer):**

Toma el texto que escribes.

### **Eval (Evaluar):**

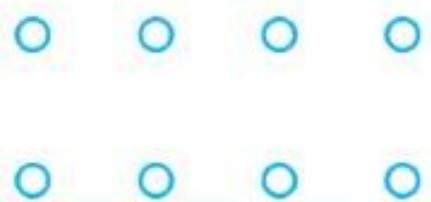
Ejecuta esa línea o expresión.

### **Print (Imprimir):**

Muestra el resultado.

### **Loop (Bucle):**

Vuelve a comenzar para la siguiente instrucción.



# Break!

[Info][Reveal\*]

Welcome! Select a time and click Start.

0 hr ▾

20 min ▾

0 sec ▾

# 00:20:00

☒ Play sound when timer ends

Start Timer

Dark Theme ▾



# Retos de programación en Python

Ejercicios prácticos:

- Calculadora simple
- Programa de saludo personalizado
- Conversor de unidades
- Crear un programa que solicite datos al usuario y genere un perfil personalizado







# Estud-IA

# GRACIAS

Apoya:



Organiza:



**Alcaldía de Medellín**  
Distrito de  
Ciencia, Tecnología e Innovación

#ciudadanosdigitales





**Título**

Subtítulo



**Título**

Subtítulo



**Título**

Subtítulo



**Título**

Subtítulo

# Título



#ciudadanosdigitales

✓ **Título**

- loremipsuloremipsu
- loremipsu
- loremipsu

✓ **Título**

- loremipsu
- loremipsuM
- loremipsu



# Título



**Título**

loremipsuloremipsuloremipsuloremipsuloremipsulorem  
ipsu