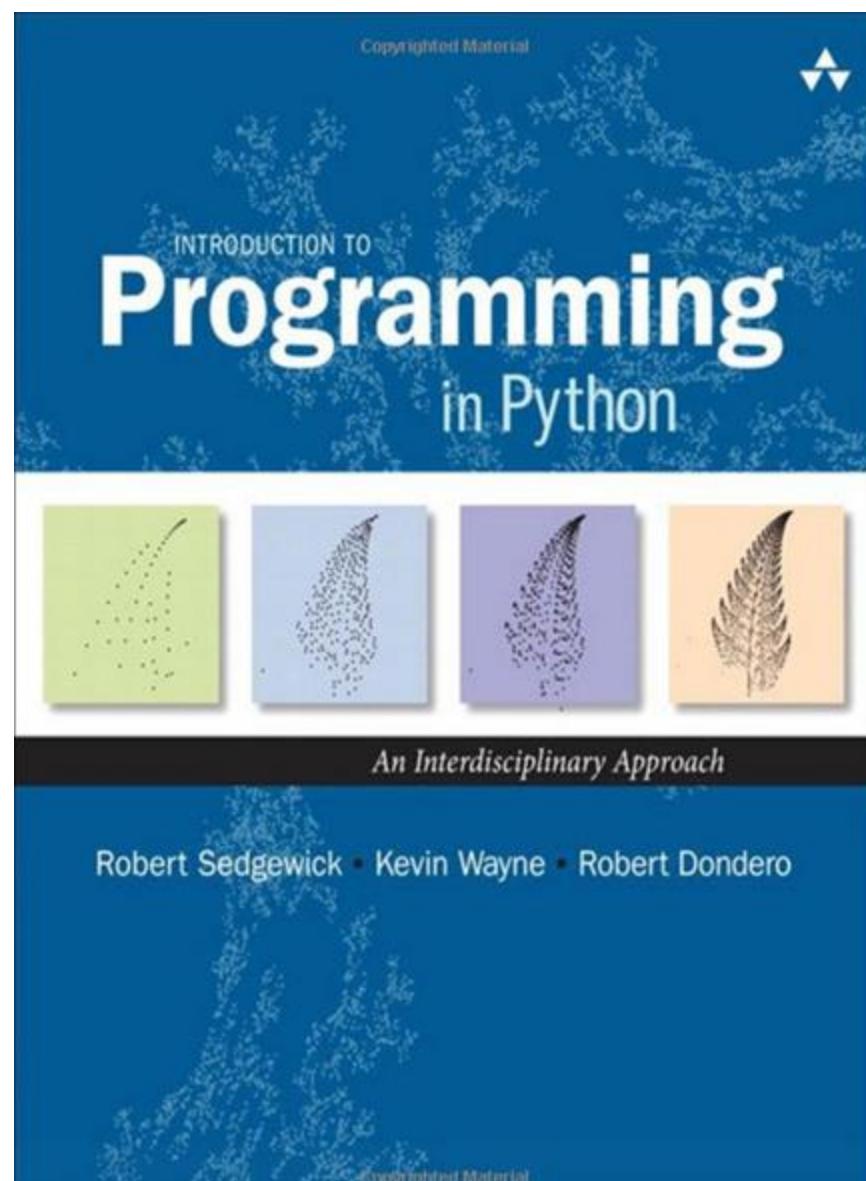


# Taller de Programación

## Introducción

Leonardo Causa  
[l.causa@udd.cl](mailto:l.causa@udd.cl)



Basada en presentaciones oficiales de libro *Introduction to Programming in Python* (Sedgewick, Wayne, Dondero).  
Disponible en <https://introcs.cs.princeton.edu/python>

# Temario

- Presentaciones
- Acuerdos
- Horarios & Bibliografía
- Introducción a Python y Anaconda
- Primer Programa 'Holamundo'

# Presentaciones



**Leonardo Causa  
Morales**

Director de Carrera

PhD. (c) Ingeniería Eléctrica, U. de Chile  
PhD. (c) Informatique Médicale, U. de Lyon  
Magíster Ingeniería Biomédica, U. de Chile  
Ingeniero Civil Electricista, U. de Chile



**Universidad del Desarrollo**  
Facultad de Ingeniería

**contacto**

[l.causa@udd.cl](mailto:l.causa@udd.cl)

# Presentaciones



# Taller de Programación

- **Cátedra:**

Martes y Jueves 11:10 - 12:20 hrs

- **Ayudantía:**

Miércoles 15:10 - 16:20 hrs

- **Consultas**

- Foro del Curso Canvas
- Escribir a [l.causa@udd.cl](mailto:l.causa@udd.cl) para solicitar hora de consulta

# Reglas

- **Evaluaciones**
  - 2 Certámenes en sala
  - 5 Tareas
  - Examen
- **Asistencia**
  - Se requiere un **70 % de asistencia** para aprobar el curso
- **Eximición:**
  - Nota de Presentación mayor o igual a 5,5 y entrega de todas las tareas en la fecha correspondiente

# Notas

C1 = Certamen 1

C2 = Certamen 2

T = Promedio de Tareas

$$\text{Nota Presentación} = 0,4*C1 + 0,4*C2 + 0,2*T$$

$$\text{Nota Final} = 0,7*\text{Nota Presentación} + 0,3*\text{Examen}$$

- Certámenes y tareas no entregadas serán evaluadas con nota 1.0
- Sólo se puede justificar 1 certamen según procedimiento de la Facultad
- Las tareas no se justifican
- **La nota de examenes debe ser mayor o igual a 3,0 para aprobar el curso**

# Contenidos

1. Introducción a programación (sintaxis, variables, funciones).
2. Programación orientada a objetos.
3. Resolución de problemas (proyectos).

# Bibliografía

- Introduction to Programming in Python

<https://introcs.cs.princeton.edu/python/home/>

- Introduction to Computation and Programming

<https://mitpress.mit.edu/books/introduction-computation-and-programming-using-python-revised-and-expanded-edition>

- How to think like a computer scientist

<http://www.greenteapress.com/thinkpython/html/index.html>

# ¿Por qué debemos aprender a programar?

## Desarrollar el pensamiento lógico matemático y algorítmico

Tener profesionales con conocimientos en programación es una gran necesidad, especialmente en el caso de América Latina, ya que IDC informa que hay un déficit de programadores del 38%. Esto podría llevar a pérdidas en ventas de hasta \$80 trillones de dólares para 2035.

# Taller de Programación

- Curso de nivel básico de programación
- Lenguaje de programación: Python



# ¿Por qué Python?



- De "**fácil uso**" -> se puede utilizar en diferentes niveles de complejidad (scripts, funciones, clases, módulos)
- **Sintaxis simple**: más parecido a un idioma como el inglés (en comparación a C, C# o Java)
- **Comunidad grande**, muy dinámica con librerías apareciendo y actualizándose regularmente
- **Versátil**: en plataformas, sistemas operativos
- **Conciso**: con menos líneas de programación se puede implementar lo mismo que en otros lenguajes

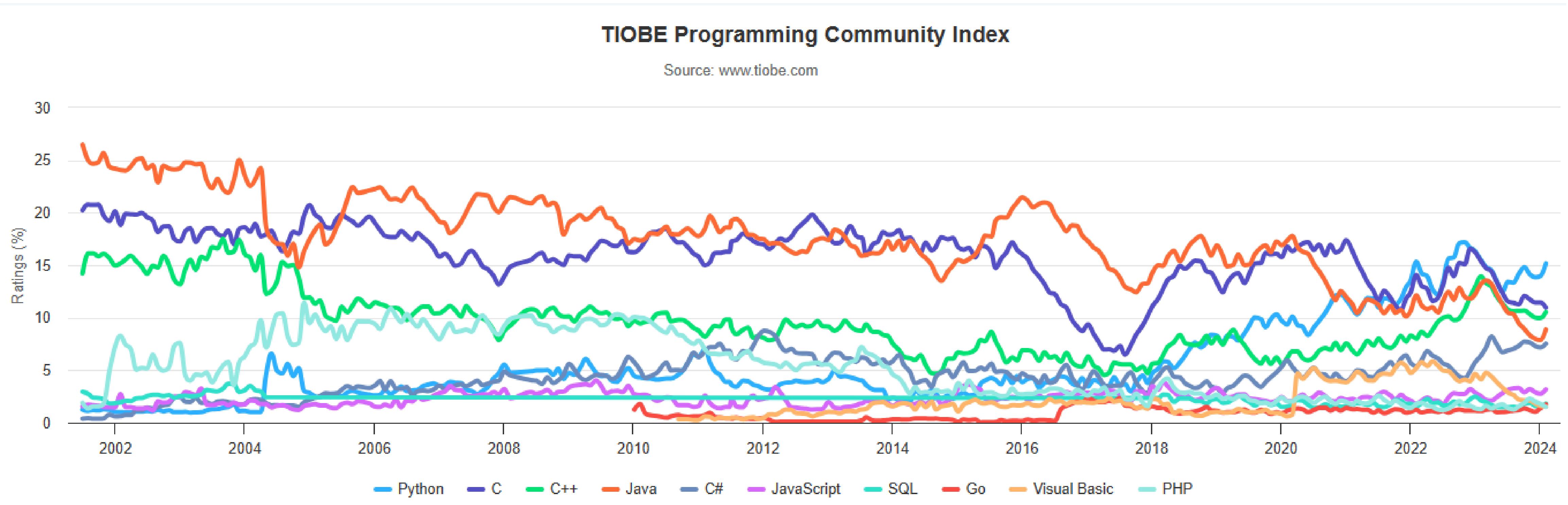


**Python** es lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.



# Popularidad de Lenguajes

- Según del Índice TIOBE



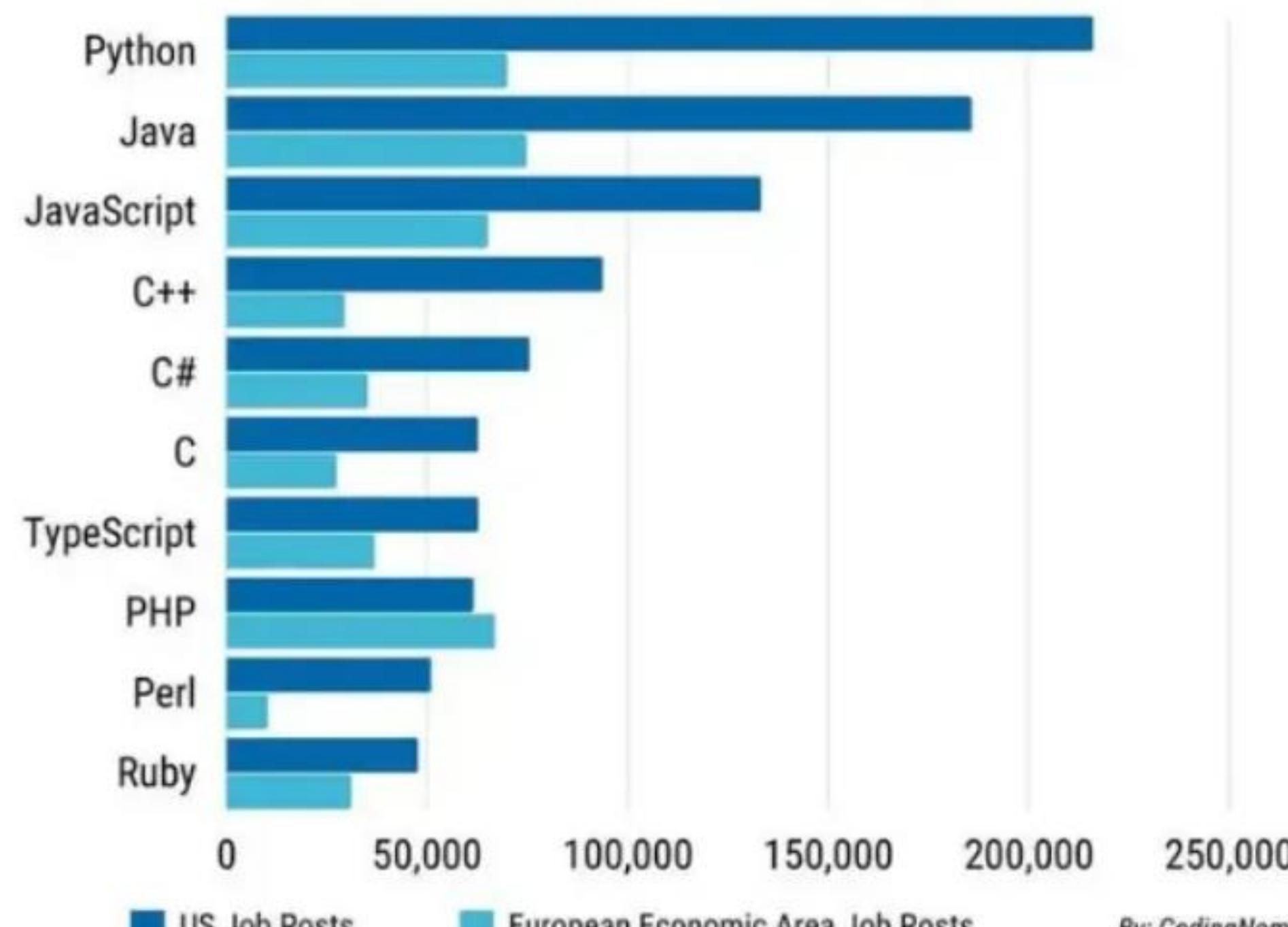
- Indicador de la popularidad de los lenguajes de programación.
- Las calificaciones se basan en la cantidad de ingenieros capacitados en todo el mundo, cursos y proveedores externos.
- El índice se actualiza una vez al mes.

<https://www.tiobe.com/tiobe-index/>

# Popularidad de Lenguajes

## Most in-demand programming languages of 2022

Based on LinkedIn job postings in the USA & Europe



By: CodingNomads

Datos recopilados de la comunidad de LinkedIn en Estados Unidos y Europa.

## Size of programming language communities in Q3 2022

Active software developers, globally, in millions

	Most popular in	Least popular in
JavaScript*	19.6 M	Apps for 3rd-party ecosystems, Cloud
Python	16.9 M	DS/ML/AI, Embedded
Java	16.5 M	Web, Mobile
C/C++	12.3 M	Cloud, Desktop
C#	10.6 M	DS/ML/AI, IoT devices
PHP	8.9 M	Embedded, IoT apps
Kotlin	6.1 M	Web, Games
Visual development tools	4.9 M	Cloud
Swift	4.2 M	AR/VR, Games
Go	4.2 M	Mobile, AR/VR
Objective C	3.8 M	Embedded, Cloud
Rust	3.0 M	Mobile, DS/ML/AI
Ruby	2.8 M	AR/VR, IoT devices
Dart	2.4 M	DS/ML/AI, Embedded
Lua	1.9 M	Mobile, Web
		Mobile, AR/VR
		Mobile, Embedded

/DATA



**La elegante sintaxis en Python, así como su código legible, son muy elogiadas; si estás comenzando tu carrera de programación, Python se adapta a tus necesidades**

- ¿Quiénes usan Python?



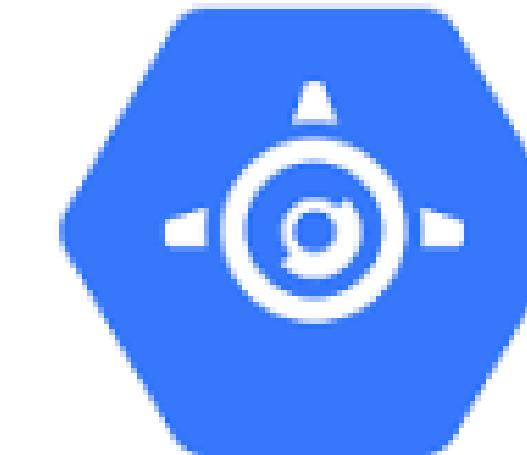
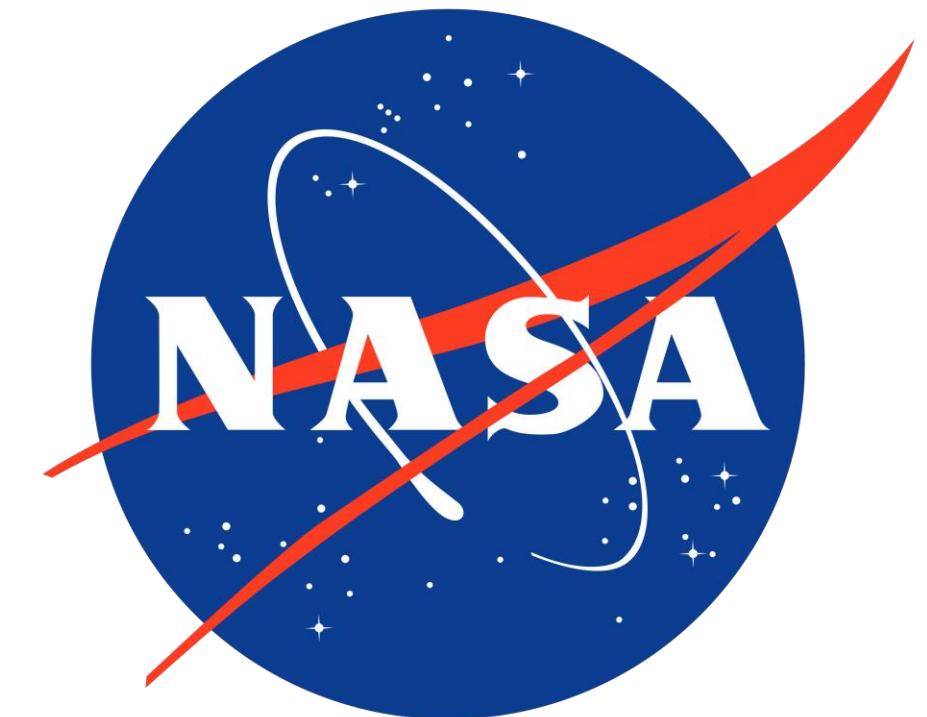
BATTLEFIELD 2™



PANDA3D



Spotify



Google  
App Engine

**WHEN YOU HERE SOMEONE SAY PYTHON  
ISN'T THE BEST PROGRAMMING LANGUAGE EVER**

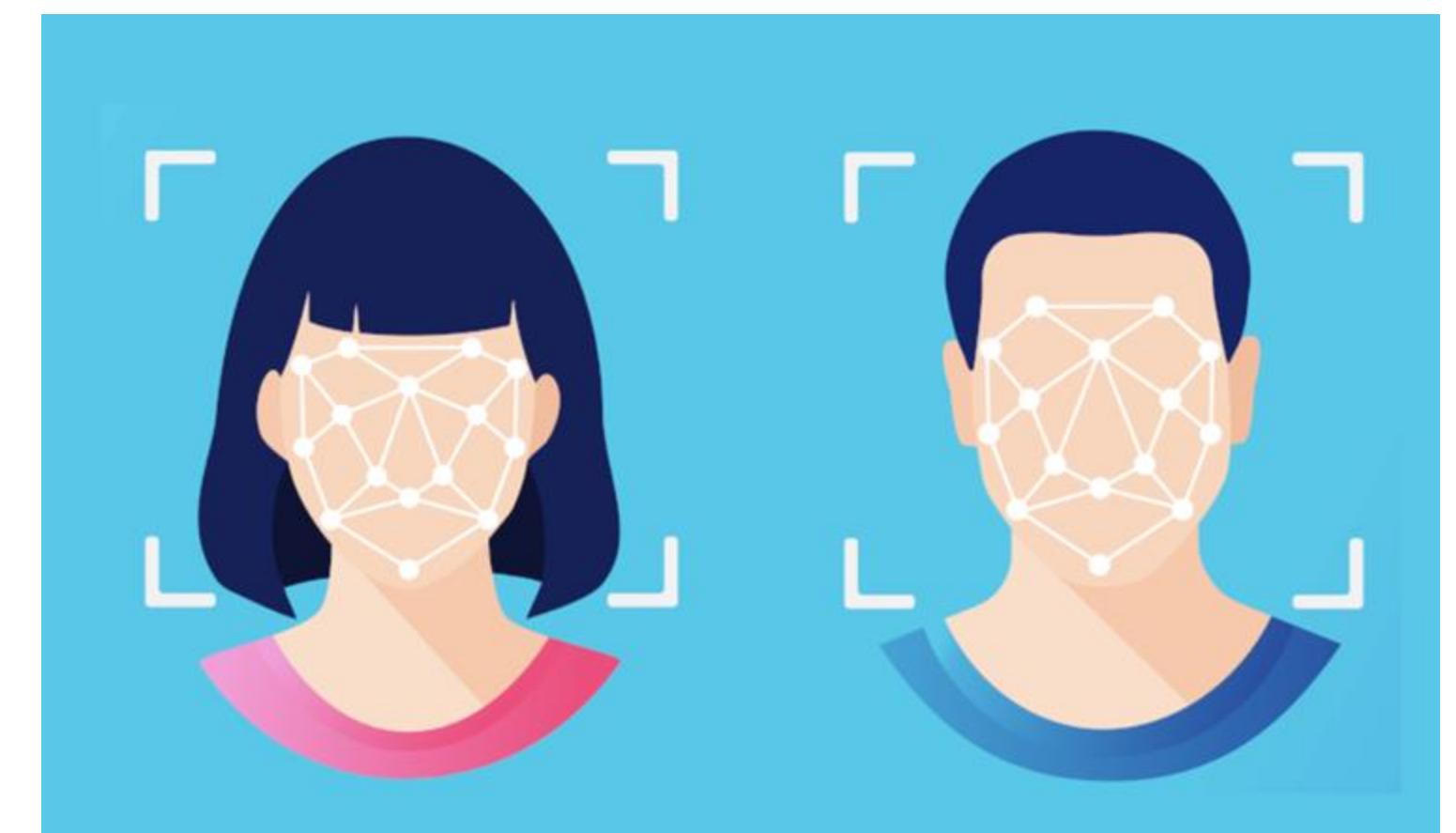


# Algoritmo

Conjunto de instrucciones (finitas) que deben seguirse por orden para ejecutar una tarea.

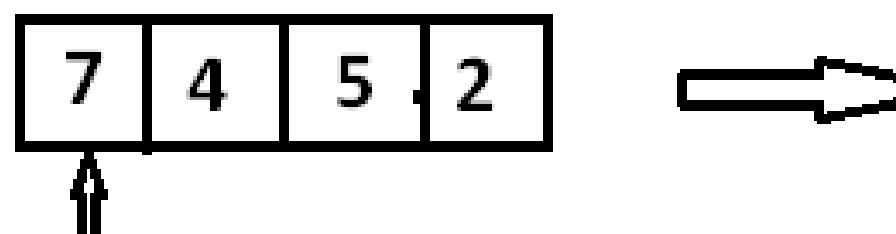
Ej:

- receta de cocina
- algoritmo para girar dinero cajero automático
- algoritmo de reconocimiento facial



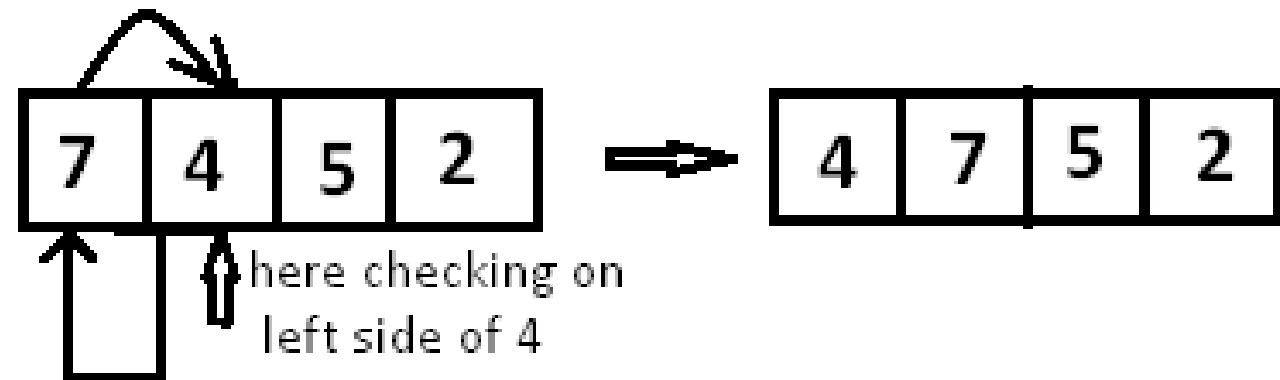
# Algoritmo

STEP 1.



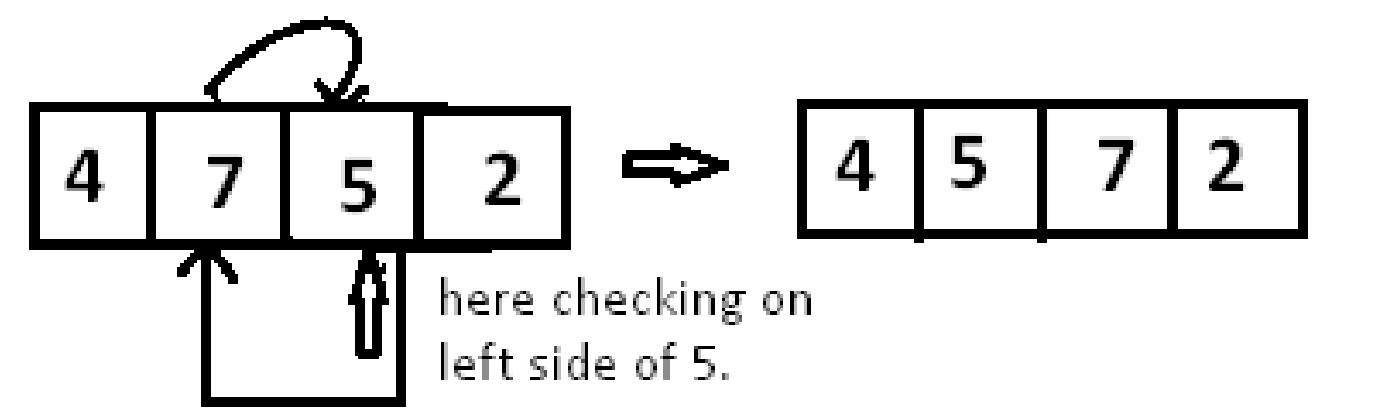
No element on left side  
of 7,so no change in its  
position .

STEP 2.



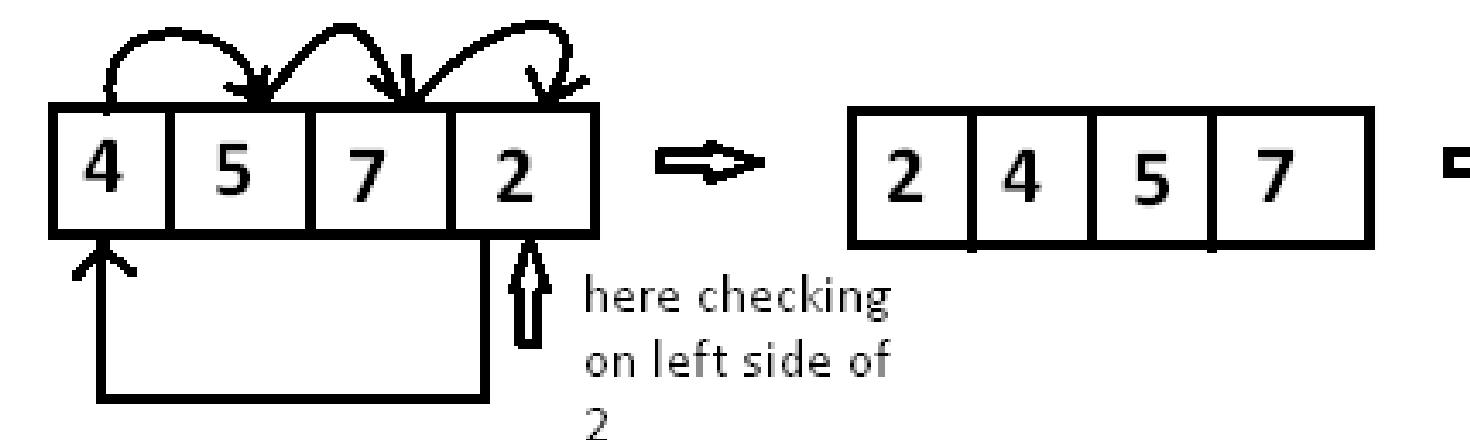
As  $7 > 4$  ,therfore 7 will  
be moved forward and 4  
will be moved to 7's  
position.

STEP 3.



As  $7 > 5$ ,7 will be moved  
forward,but  $4 < 5$ ,so no  
change in position of 4.  
And 5 will be moved to  
position of 7.

STEP 4.



As all the element on left side  
of 2 are greater than 2,so all  
the elements will be moved  
forward and 2 will be shifted  
to position of 4

# Lenguajes de Programación

Lenguaje formal creado para darle instrucciones a un computador.



# Anaconda

 **ANACONDA NAVIGATOR**

Sign in to Anaconda Cloud

Home Applications on base (root) Channels Refresh

**JupyterLab** 1.2.6 An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture. Launch

**jupyter Notebook** 6.0.3 Web-based, interactive computing notebook environment; Edit and run human-readable docs while describing the data analysis. Launch

**IPyConsole** 4.6.0 PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more. Launch

**Spyder** 4.0.1 Scientific Python Development Environment: Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features. Launch

**dioptas** 0.4.1 Documentation

**fsleyes** 0.33.2 Documentation

**Glueviz** 0.15.2 Multidimensional data visualization across files. Explore relationships within and among related datasets. Documentation

**gnuradio** 3.8.1.0 Documentation

[Documentation](#) [Developer Blog](#)

# Anaconda



Windows	MacOS	Linux
Python 3.8	Python 3.8	Python 3.8
<a href="#">64-Bit Graphical Installer (466 MB)</a>	<a href="#">64-Bit Graphical Installer (462 MB)</a>	<a href="#">64-Bit (x86) Installer (550 MB)</a>
<a href="#">32-Bit Graphical Installer (397 MB)</a>	<a href="#">64-Bit Command Line Installer (454 MB)</a>	<a href="#">64-Bit (Power8 and Power9) Installer (290 MB)</a>

# Spyder & Python



The screenshot shows the Spyder IDE interface. On the left is a code editor with the following Python code:

```
7 import pylab
8 from numpy import cos, linspace, pi, sin, random
9 from scipy.interpolate import splprep, splev
10
11 # Generate data for analysis
12
13# Make ascending spiral in 3 space
14t = linspace(0, 1.75 * 2 * pi, 100)
15
16x = t * sin(t)
17y = -cos(t)
18z = t
19
20# Add noise
21x += random.normal(scale=0.1, size=x.shape)
22y += random.normal(scale=0.1, size=y.shape)
23z += random.normal(scale=0.1, size=z.shape)
24
25
26# Perform calculations
27
28# Spline parameters
29smoothness = 1.0 # Smoothness parameter
30k_param = 2 # Spline order
31nests = -1 # Estimate of number of knots needed (-1 = maximal)
32
33# Find the knot points
34knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=nests)
35
36# Evaluate spline, including interpolated points
37xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)
38
39
40# Plot results
41
42# TODO: Rewrite to avoid code smell.
43pylab.subplot(2, 2, 1)
44data, = pylab.plot(x, 'bo', label='Data with X-Y Cross Section')
45surf, = pylab.plot_surface(xnew, ynew, znew, 'r', label='Fit with X-Y Cross Section')
46pylab.legend()
47pylab.xlabel('X')
48pylab.ylabel('Y')
```

On the right, there is a variable explorer showing objects like 'bars', 'df', 'filenames', 'list\_text', 'nrows', 'r', 'resid', 'region', 'rgb', and 'series'. Below the code editor is a 3D surface plot of the data.

- Scientific Python Development Environment
- Entorno de desarrollo interactivo para el lenguaje Python



colab

- Jupyter Notebook es un entorno informático interactivo para programar.

- Colaboratory es un entorno informático interactivo en la nube para programar (Google).

# Descargar e Instalar Anaconda

- Mac: [https://repo.anaconda.com/archive/Anaconda3-2022.05-MacOSX-x86\\_64.pkg](https://repo.anaconda.com/archive/Anaconda3-2022.05-MacOSX-x86_64.pkg)
- Windows:  
<https://docs.anaconda.com/anaconda/install/windows/#:~:text=Download%20the%20Anaconda%20installer>
- Linux: [https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86\\_64.sh](https://repo.anaconda.com/archive/Anaconda3-2021.11-Linux-x86_64.sh)

# repl.it

The screenshot shows the repl.it web-based development environment. At the top, there are icons for user profile, workspace name 'dopitz / Clase01', Python logo, and a refresh arrow. Below the header is a 'Run ▶' button. On the left is a sidebar titled 'Files' containing three files: 'main.py' (selected), '01\_holamundo.py', and '02\_variables.py'. The main area is a code editor for 'main.py' with the following content:

```
1 Not sure what to do? Run some examples (start typing to dismiss)
```

- Entorno de desarrollo interactivo gratuito para varios lenguajes incluido Python

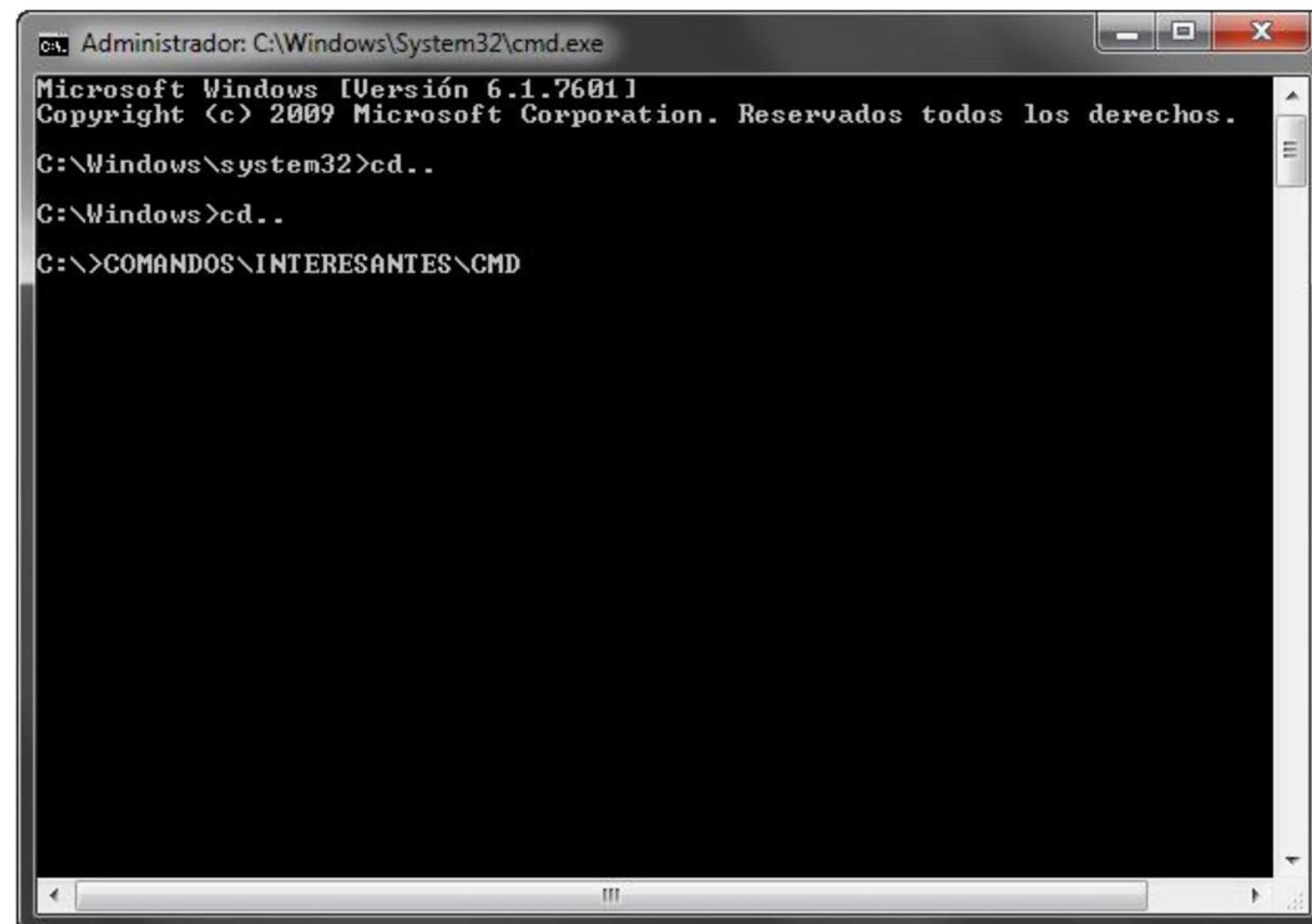
# Shells

Shell: intérprete de órdenes o intérprete de comandos. Provee una interfaz de usuario para acceder a los servicios del sistema operativo.

- De líneas texto (CLI)
- Gráficos (GIU)
- De lenguaje natural (NUI)

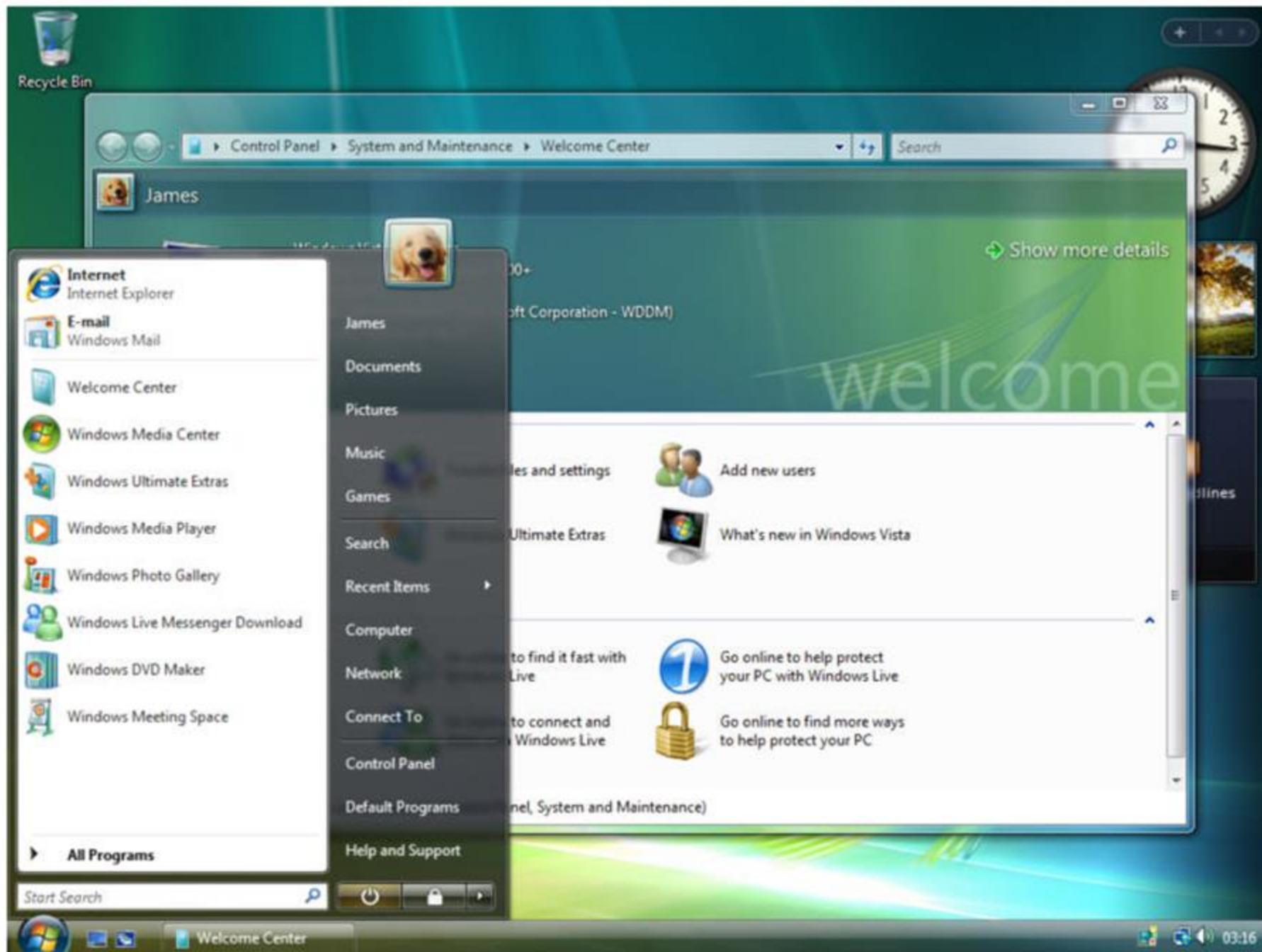
# CLI

Command interface: permite a los usuarios dar instrucciones por medio de una línea de texto simple.



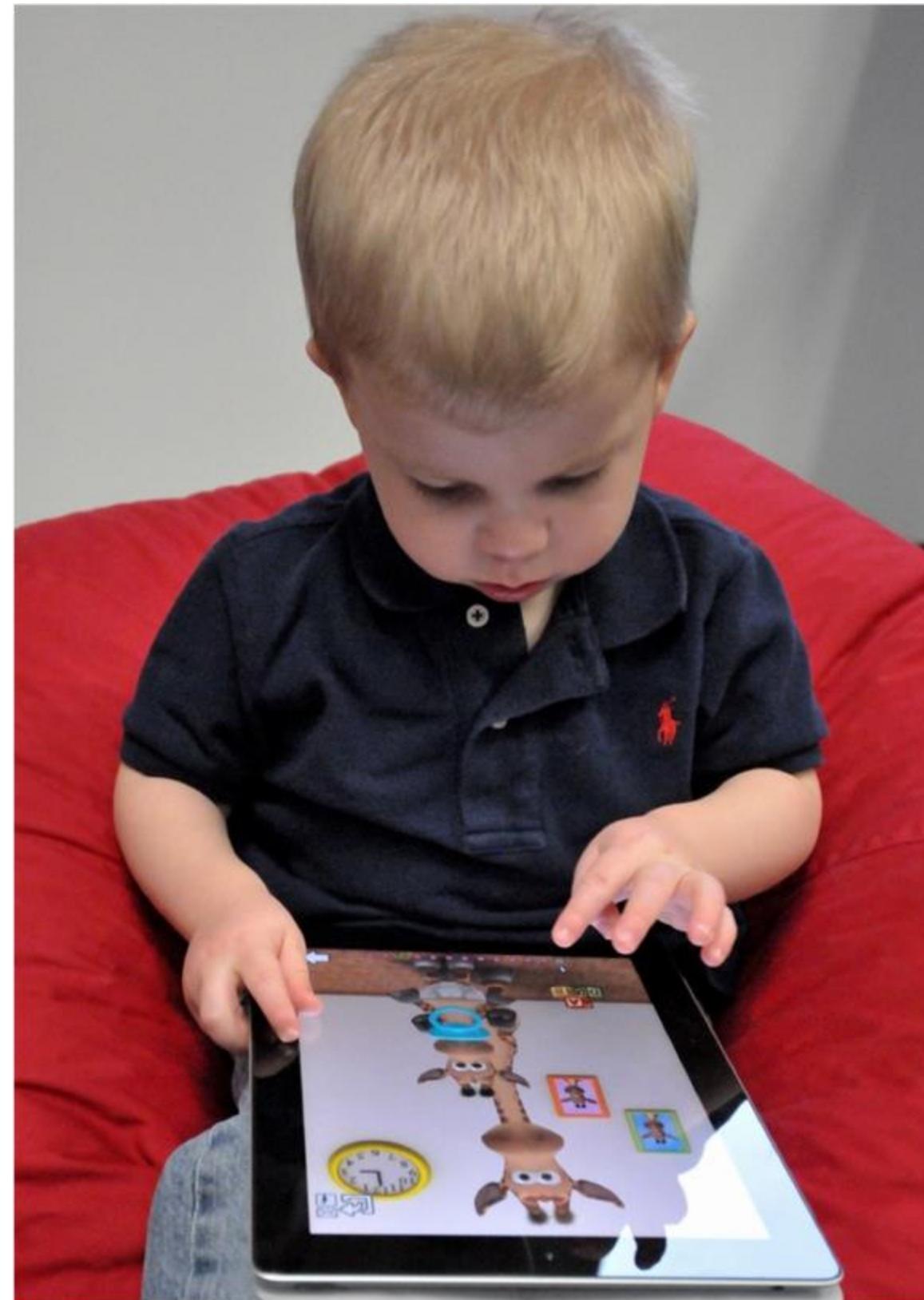
# GUI

Graphical user interface: utiliza imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo.



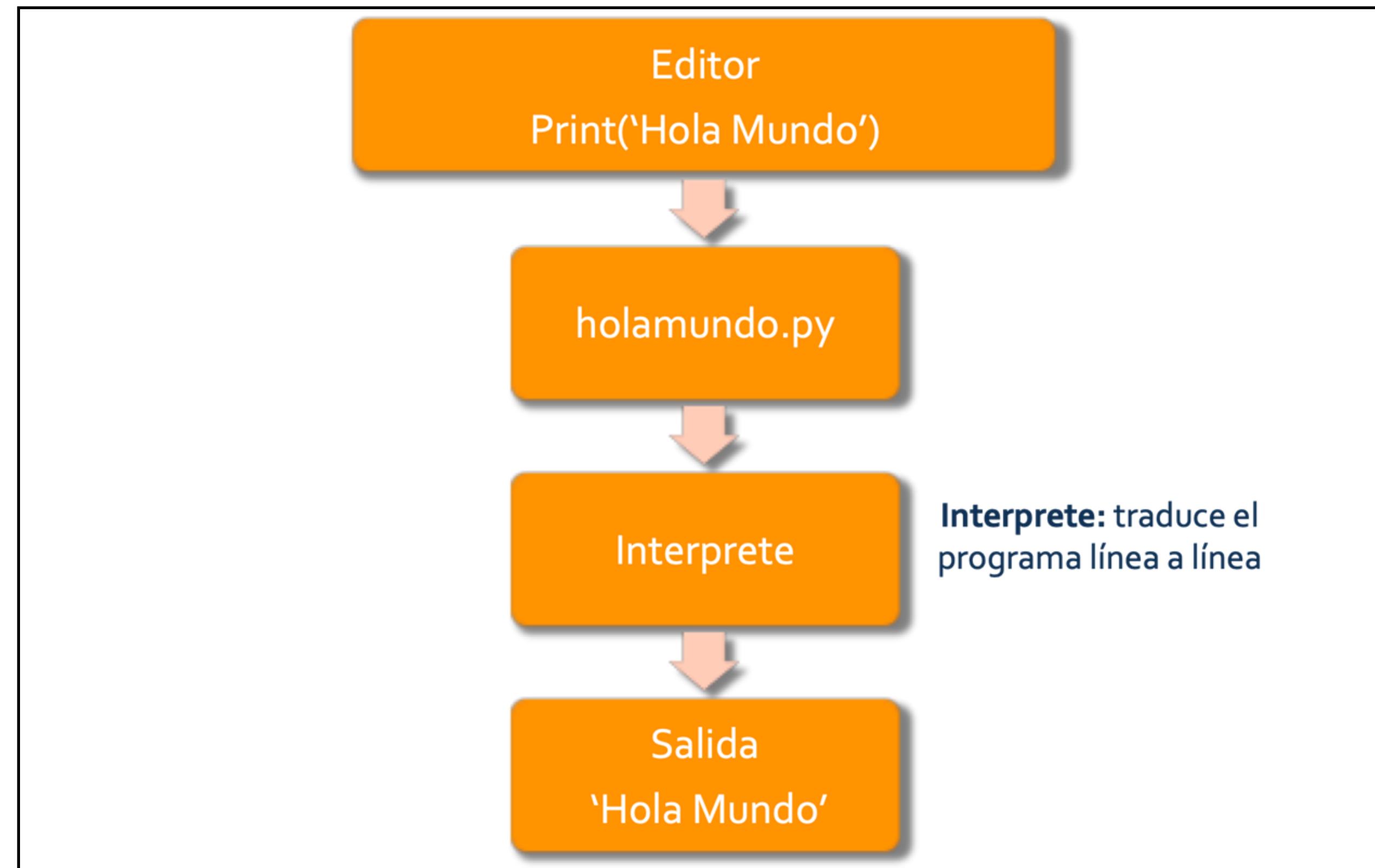
# NUI

Natural user interface: utiliza movimientos gestuales del cuerpo o de alguna de sus partes tales como las manos o los pies.



# Hola mundo

```
# holamundo.py  
print('Hola  
mundo!')
```



```
2. bash  
$ python3 holamundo.py  
Hola mundo!  
$
```

# Tips

## **PRÁCTICA, PRÁCTICA, PRÁCTICA!**

- Haz los ejercicios de las ayudantías y las tareas
- Pregunta a la profesores, ayudantes y a tus compañeros
- Haz grupos de estudio
- No copies ni pagues por tareas
- Usa sabiamente las tecnologías disponibles como chatGPT u otros