

# Curso: Inteligencia Artificial

**Unidad 1:** Conceptos fundamentales de inteligencia artificial

**Sesión 7:** Introducción al lenguaje Python

**Docente:** Carlos R. P. Tovar

# INICIO

## Objetivo de la sesión

Al finalizar la sesión, el alumno será capaz de:

- Conocer el origen y evolución de Python.
- Comprender sus características principales.
- Ejecutar programas simples en Python.
- Reconocer la sintaxis y tipos de datos básicos.



# Introducción

- **Frase:** "Los programas deben escribirse para que las personas los lean, y solo incidentalmente para que las máquinas los ejecuten."  
– Harold Abelson

# ¿Qué es Python?

- Lenguaje de programación interpretado, dinámico y de alto nivel.
- Creado por Guido van Rossum en 1991.
- Filosofía: código legible, claro y conciso.
- Uso en IA, análisis de datos, web, automatización, IoT, ciberseguridad.



[Esta foto](#) de Autor desconocido está bajo licencia [CC BY-SA-NC](#)



[Esta foto](#) de Autor desconocido está bajo licencia [CC BY-SA-NC](#)

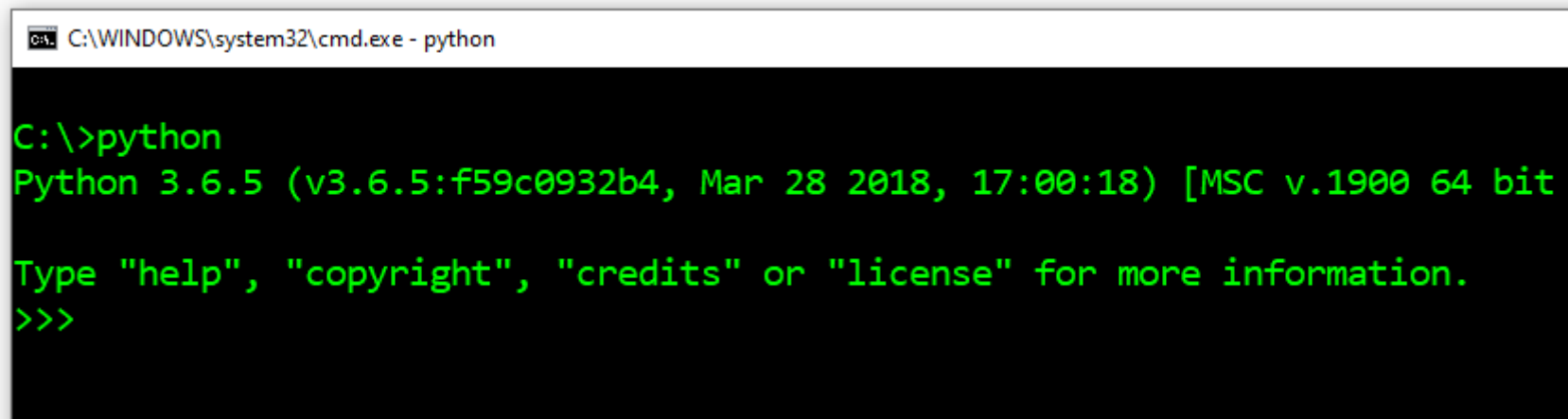
# Características Clave

- Sintaxis simple y legible.
- Multiplataforma.
- Tipado dinámico.
- Multiparadigma: orientado a objetos, imperativo y funcional.
- Gran ecosistema de bibliotecas: NumPy, Pandas, TensorFlow, Flask, etc.



# Ejecución de Python

- Se puede ejecutar de varias formas:
- Intérprete interactivo (python o python3 en consola).
- Archivos .py desde terminal o IDE (VS Code, PyCharm).
- Jupyter Notebooks para ciencia de datos. Ejemplo:



```
C:\WINDOWS\system32\cmd.exe - python

C:\>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

[Esta foto](#) de Autor desconocido está bajo licencia [CC BY-SA-NC](#)

# Ejecución de Python

## Instrucciones simples

- Impresión de un texto  
`print("Hola mundo")`
- Comentario:  
`# Esto es un comentario`
- Comentario multilínea:  
`"""  
Esto es un comentario  
de varias líneas  
"""`

## Variables

```
x = 5  
nombre = "Ana"  
pi = 3.1416
```



# Tipos de Datos Básicos

- Numéricos: int, float, complex.
- Texto: str.
- Booleanos: True, False.
- Ejemplo:

entero = 10

decimal = 3.14

texto = "Python"

bandera = True



# Cadenas de Texto

## Concatenación:

"Hola" + " Mundo"

## Formato:

```
nombre = "Ana"  
print(f"Hola {nombre}")
```

# Listas en Python

## Definición:

- Colecciones ordenadas, indexadas y mutables.
- Permiten elementos duplicados.

## Ejemplos básicos:

```
frutas = ["manzana", "pera",  
"uva"]  
  
print(frutas)      # ['manzana',  
                    'pera', 'uva']  
  
print(frutas[0])   # manzana
```

# Listas en Python

## Definición:

- Colecciones ordenadas, indexadas y mutables.
- Permiten elementos duplicados.

## Ejemplos básicos:

```
frutas = ["manzana", "pera",  
"uva"]
```

```
print(frutas)
```

```
# ['manzana', 'pera', 'uva']
```

```
print(frutas[0])
```

```
# manzana
```

# Listas en Python

## Definición:

- Colecciones ordenadas, indexadas y mutables.
- Permiten elementos duplicados.

## Ejemplos básicos:

```
frutas = ["manzana", "pera",  
"uva"]
```

```
print(frutas)
```

```
# ['manzana', 'pera', 'uva']
```

```
print(frutas[0])
```

```
# manzana
```

# Listas en Python

- **Operaciones comunes:**

```
frutas.append("mango")
```

# Agrega al final

```
frutas.insert(1, "kiwi")
```

# Inserta en posición

```
frutas.remove("pera")
```

# Elimina por valor

```
frutas.pop(0)
```

# Elimina por índice

```
print(len(frutas))
```

# Longitud de la lista

- **Slicing:**

```
print(frutas[1:3])
```

# Sublista desde índice 1 hasta 2

# Tuplas

## Definición:

- Colecciones ordenadas, indexadas e **inmutables**.
- Útiles para datos que no deben cambiar.

```
colores = ("rojo", "verde", "azul")  
print(colores[0]) # rojo
```

## Ventajas:

- Ocupan menos memoria que las listas.
- Pueden usarse como claves en diccionarios.

## Conversión:

```
lista = list(colores)  
tupla = tuple(lista)
```

# Diccionarios

## Definición:

- Colecciones no ordenadas de pares clave: valor.
- Claves únicas y valores mutables.

## Ejemplo:

```
persona = {"nombre": "Juan",  
"edad": 25, "ciudad": "Lima"}  
print(persona["nombre"]) # Juan
```

## Operaciones:

```
persona["profesion"] =  
"Ingeniero" # Agregar
```

```
persona["edad"] = 26  
# Modificar
```

```
del persona["ciudad"]  
# Eliminar
```



# Diccionarios

## Recorrido

```
for clave, valor in persona.items():  
    print(clave, ":", valor)
```

# Operadores

## Aritméticos:

```
a, b = 10, 3
```

```
print(a + b) # 13
```

```
print(a ** b) # 1000 (potencia)
```

## Comparación:

```
print(a > b) # True
```

```
print(a == 10) # True
```

# Operadores

## Lógicos :

```
print(a > 5 and b < 5) # True  
print(not a == b)      # True
```

## De asignación :

```
x = 5  
x += 2 # 7
```

# Condicionales if

**Sintaxis :**

```
edad = 20
```

```
if edad >= 18:
```

```
    print("Mayor de edad")
```

```
elif edad >= 13:
```

```
    print("Adolescente")
```

```
else:
```

```
    print("Menor de edad")
```

**Condiciones múltiples :**

```
if 0 < edad < 120:
```

```
    print("Edad válida")
```

# Bucles: while

**Ejemplo :**

```
i = 1
while i <= 5:
    print(i)
    i += 1
```

**Uso de break y continue:**

```
while True:
    entrada = input("Escribe 'salir'
para terminar: ")
    if entrada == "salir":
        break
```

# Bucles: for

**Ejemplo :**

```
for fruta in ["manzana", "pera",  
"uva"]:  
    print(fruta)
```

**Uso con range():**

```
for i in range(1, 6):  
    print(i)
```

# Funciones

**Ejemplo básico :**

```
def saludar(nombre):  
    return f"Hola, {nombre}"  
  
print(saludar("Carlos"))
```

**Parámetros con valor por defecto:**

```
def potencia(base,  
exponente=2):  
    return base ** exponente
```



# Manejo de Errores

## Estructura try-except :

try:

```
x = int("abc")
```

except ValueError:

```
print("Error: no es un número")
```

## Múltiples excepciones :

try:

```
archivo = open("datos.txt")
```

except (FileNotFoundError,  
PermissionError):

```
print("Error de archivo")
```

# Módulos y Librerías

**Importar módulo completo :**

```
import math  
print(math.sqrt(16))
```

**Importar función específica :**

```
from math import pi  
print(pi)
```

**Instalar paquetes externos :**

```
pip install numpy
```

# Módulos y Librerías

## Simulador de respuestas aleatorias:

```
from random import choice
```

```
respuestas = ["Sí", "No", "Tal vez", "Pregunta de nuevo"]
```

```
print("Pregunta algo a la IA mágica...")
```

```
input("> ")
```

```
print(choice(respuestas))
```

- Introduce el concepto de aleatoriedad y simulación.
- Base para chatbots y modelos simples.



**Universidad  
Tecnológica  
del Perú**