

INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN PYTHON

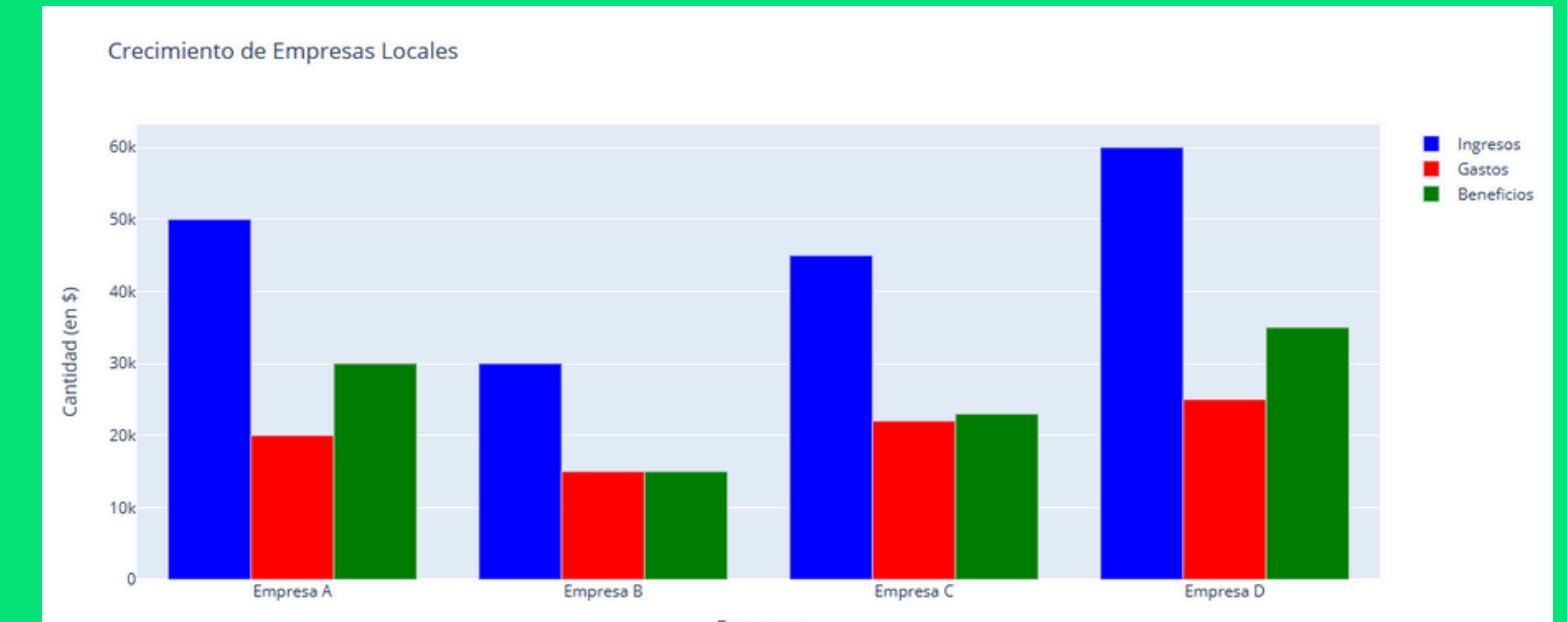
DOCENTE: JEAN MARCOS CABALLERO REQUIZ,
BACHILLER EN INGENIERIA ELECTRONICA

índice

- 01** Introducción a Python y fundamentos de la programación.
- 02** Estructuras de Control
- 03** Estructuras de Datos Básicas
- 04** Funciones y Manejo de Errores
- 05** Introducción a las librerías

Algunas Líneas de
Aprendizaje de
especialidades más
solicitadas utilizando
Python

Línea de Aprendizaje: Data Science



01



FUNDAMENTO
DE PYTHON



Familiarízate con la sintaxis básica, estructuras de datos como listas, diccionarios y tuplas, y conceptos de programación orientada objetos.

02



MANIPULACIÓN
Y ANÁLISIS
DE DATOS
CON NUMPY
Y PANDAS



Aprende a utilizar estas herramientas para cargar, limpiar, transformar y analizar conjuntos de datos.

03



VISUALIZACIÓN
DE DATOS CON
MATPLOTLIB Y
SEABORN



Aprende a utilizar Matplotlib para gráficos básicos y Seaborn para visualizaciones estadísticas más avanzadas.

04



ESTADÍSTICAS Y
PROBABILIDAD



Esto te ayudará a comprender mejor tus datos y los métodos de análisis que utilizarás más adelante.

05



APRENDIZAJE DE
SCIKIT-LEARN



Aprende a utilizar scikit-learn para aplicar técnicas de aprendizaje supervisado y no supervisado a tus datos.

06



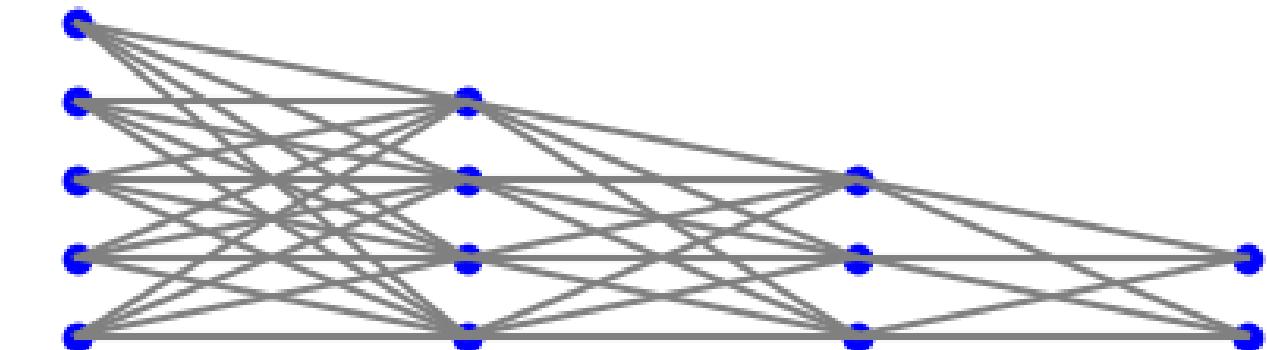
INTRODUCCIÓN AL
MACHINE LEARNING



Comprende los conceptos fundamentales del machine learning.

Línea de Aprendizaje: Redes Neuronales

Red Neuronal



01



CONCEPTOS
BÁSICOS DE
MACHINE
LEARNING Y
PYTHON



Comprensión
básica de los
principios del
machine learning y
familiarizarte con
Python.

02



APRENDIZAJE
DE NUMPY Y
PANDAS



Estas bibliotecas
son esenciales
para el manejo y
procesamiento de
datos en Python

03



ESTUDIO DE LA
BIBLIOTECA
SCIKIT-LEARN



Scikit-learn es una
biblioteca de
machine learning en
Python que ofrece
una amplia gama
de algoritmos de
aprendizaje.

04



INTRODUCCIÓN
A LAS REDES
NEURONALES



Comienza con una
introducción teórica
a las redes
neuronales
artificiales,
comprendiendo los
conceptos
de
neuronas,
funciones
activación,
capas,
de
propagación

05



APRENDIZAJE DE
TENSORFLOW O
PYTORCH



TensorFlow y PyTorch
son dos de las
bibliotecas más
populares para el
desarrollo de redes
neuronales en Python

06



PRÁCTICA CON
PROYECTOS SIMPLES



Empieza a
implementar proyectos
simples de redes
neuronales, como
clasificación
de imágenes con
conjuntos de datos
conocidos

Línea de Aprendizaje: Procesamiento de Imágenes

01



FUNDAMENTOS DE PYTHON



Familiarízate con la sintaxis básica, estructuras de datos como listas, diccionarios y tuplas, y conceptos de programación orientada a objetos.

02



LIBRERÍAS BÁSICAS DE MANIPULACIÓN DE IMÁGENES



Familiarízate con las bibliotecas básicas de Python para el manejo de imágenes, como Pillow y OpenCV

03



INTRODUCCIÓN A OPENCV



Aprende los fundamentos de OpenCV, incluyendo la lectura y escritura de imágenes, manipulación de píxeles, detección de bordes y contornos, y segmentación de imágenes

04



PROCESAMIENTO AVANZADO DE IMÁGENES CON OPENCV



Explora técnicas más avanzadas de procesamiento de imágenes con OpenCV.

05



DETECCIÓN DE OBJETOS Y RECONOCIMIENTO DE PATRONES



Aprende sobre técnicas de detección de objetos y reconocimiento de patrones en imágenes utilizando algoritmos.

06

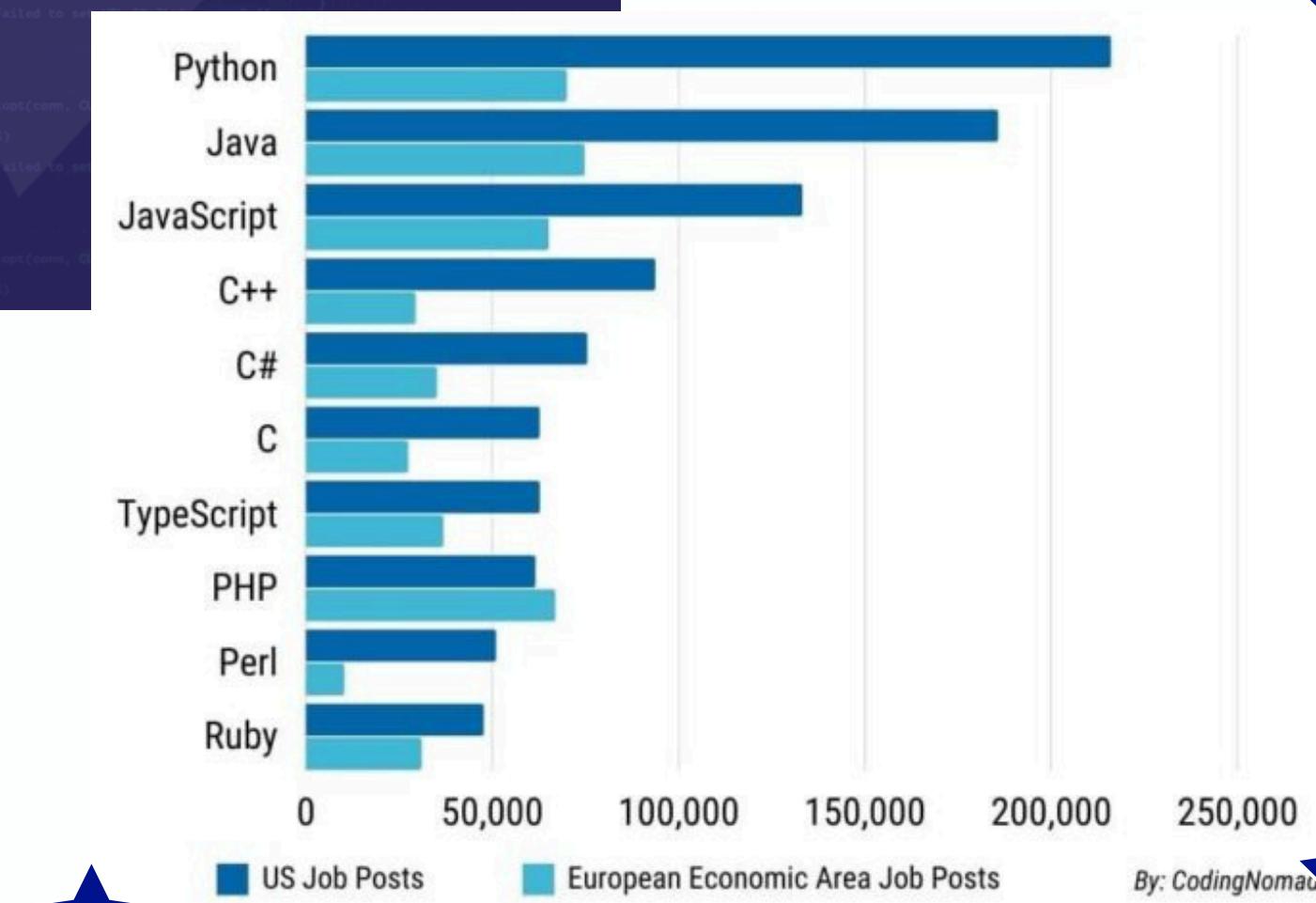


PROYECTOS PRÁCTICOS



Aplica conocimientos en tus proyectos prácticos de procesamiento de imágenes.

LENGUAJES DE PROGRAMACIONES SOLICITADOS EN EL AÑO 2023



By: CodingNomad

CREADOR DE PYTHON

Fue creado por el informático Guido van Rossum, quien había estado trabajando con un lenguaje llamado ABC en su anterior trabajo en el Centrum Wiskunde & Informática (CWI) - Instituto Nacional de Investigación en Matemáticas e Informática en los Países Bajos-. Aunque le gustaban algunos aspectos de ABC, estaba frustrado por lo difícil que era difundir este lenguaje.



¿QUÉ ES PYTHON?

Python es un lenguaje de programación de alto nivel, orientado a objetos, con una semántica dinámica integrada, principalmente para el desarrollo web y de aplicaciones informáticas.



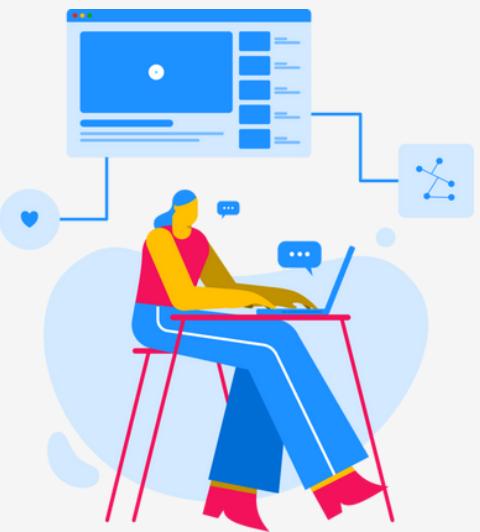


VENTAJAS

- Sintaxis clara y legible
- Gran comunidad y amplia disponibilidad de bibliotecas
- Multiplataforma
- Orientado a objetos
- Amplio uso en diferentes áreas

Desventajas

- Velocidad de ejecución
- Uso intensivo de memoria
- GIL (Global Interpreter Lock)
- Menor rendimiento en aplicaciones de escritorio



Diferentes
plataformas que
pueden utilizar
PYTHON



DIFERENTES PLATAFORMAS QUE PUEDEN UTILIZAR PYHTON

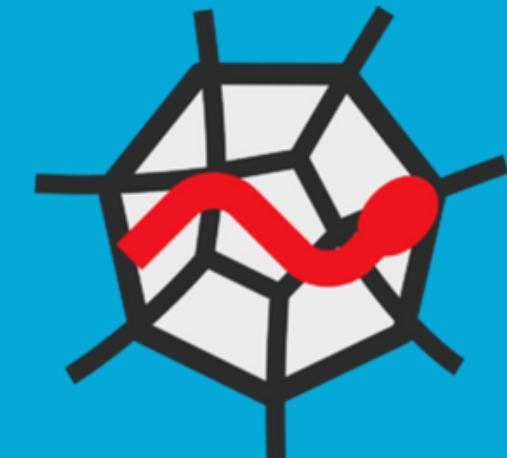
ANACONDA



GOOGLE COLAB



SPYDER
(ANACONDA)



SPYDER

SUBLIME TEXT 3



APRENDIZAJE INICIAL: ERRORES DE SINTAXIS

- Necesitamos aprender Lengua Python para poder comunicar nuestras instrucciones a Python. Al principio, cometemos muchos errores y hablaremos mal como ocurre con los niños pequeños.
- Cuando usted comete un error, la computadora no cree que usted es “TIERNO”. Le dice que hay “error de sintaxis” (Syntax error) porque ella conoce el lenguaje, pero usted recién lo está aprendiendo. Da la sensación de que Python es cruel y carece de sentimientos.
- Sin embargo, recuerda que usted es inteligente y puede aprender. La computadora es simple y muy veloz pero incapaz de aprender. Entonces, es más sencillo para usted aprender Python que para la computadora aprenda español.

SENTENCIAS O LÍNEAS

VARIABLE

CONSTANTE

FUNCTION

OPERADOR

```
1 Y = 2  
2 X = 4 + Y  
3  
4 print(X)
```

Variable = Y, X

Constante = 2, 4

Funcion = +, =

Operador = print()

PALABRAS RESERVADAS

Estas palabras no pueden ser utilizadas como nombres o identificación de variables.

False / class / return / is / finally / none if / for
/ lambda / continue / true / from while / non /
local / not / as / and / elif global / not / with /
try / or / assert / yield else / import / pass /
break / except / in raise

INICIANDO CON LOS EJEMPLOS

Estos ejemplos son para que vayan entrando de lleno a la programación:

Ejecutar los siguientes códigos:

>>> print("Hola a todos!") ->¿que se va a ejecutar?

>>> #print("Hola Mundo") ->¿que se va a ejecutar?

Constantes

Los valores fijos con números, letras y cadenas reciben el nombre de constantes porque su valor no cambia.

- Las constantes numéricas son las que usted espera.
- Las constantes de la cadena son comillas simples ('') o dobles ('').

VARIABLES

Las variables son contenedores que se utilizan para almacenar datos en la memoria. Cada variable tiene un nombre único que se utiliza para acceder y manipular los datos almacenados en ella. Las variables pueden contener diferentes tipos de datos, como números, cadenas de texto, listas, diccionarios y objetos personalizados.

FACILIDAD EN PONER NOMBRE A SUS VARIABLES

BLOQUE 1

DIFÍCIL:

```
asdadasd = 80.0
```

```
aqwwqaaqw = 12.9
```

```
xrtxrtrtx = asdasdasd
```

```
+ aqwwqaaqw
```

```
print(xrtxrtrtx)
```



BLOQUE 2

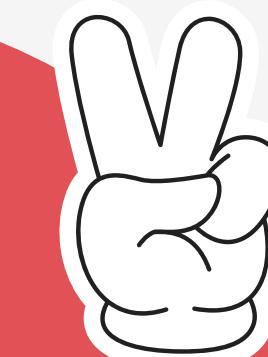
FACIL:

```
a = 80.0
```

```
b = 12.9
```

```
c = a + b
```

```
print(c)
```



Expresiones Numéricas

En este cuadro podemos ver las expresiones matemáticas básicas que puede hacer Python.



Operador	Operación
+	Suma
-	Resta
*	Multiplicación
/	División
**	Potencia
%	Resto

INICIANDO CON LOS EJEMPLOS

Realice un código donde se puedan ver las siguientes expresiones.

Datos:

A = 12

B = 44

Realizar suma, resta, multiplicación, división, potencia y resto en un solo bloque de código.

Reglas cuando se elabora una operación matemáticas

Existe una regla cuando operadores matemáticos en python es el que:

- Siempre se respetan los paréntesis
- Potencias (elevar a la potencia)
- Multiplicación, división, resto
- Suma resta
- Izquierda a derecha

Paréntesis

Potencia

Multiplicación

Suma

Izquierda a
derecha

¿QUE SON LOS INT, FLOAT, STR Y BOLEANS?

INT = integer

LOS INTEGERS SON LOS NÚMEROS ENTEROS.

STR = STRINGS

LOS STRINGS SON LAS CADENAS DE CARACTERES, ES DECIR, UN CONJUNTO DE CARACTERES QUE PUEDEN SER LETRAS, NÚMEROS O SÍMBOLOS.

FLOAT = numero flotante

LOS FLOAT SON LOS NÚMEROS DECIMALES

BOOLEANS

SON UN TIPO DE DATO FUNDAMENTAL EN LA PROGRAMACIÓN QUE REPRESENTA UN VALOR DE VERDAD.

¿Que son los INT, FLOAT y STR? EJEMPLOS

```
1 #QUE ES UN STR:  
2 X = "HOLA MUNDO"  
3 print(X)
```

HOLA MUNDO

```
1 #QUE ES UN INT:  
2 Y = 2  
3 print(Y)
```

2

```
1 #QUE ES UN FLOAT  
2 Z = 8.77  
3 print(Z)
```

8.77

```
1 #QUE ES UN BOOLEANO  
2 A = True  
3 B = False  
4 print(A)  
5 print(B)
```

True
False

¿QUE ES UN "TYPE" EN PYTHON?

Es una función incorporada que devuelve el tipo de los objetos/elementos almacenados en cualquier tipo de datos o devuelve un nuevo objeto de tipo dependiendo de los argumentos pasados a la función.

```
1 #QUE ES UN TYPE:  
2 print(type(X))  
3 print(type(Y))  
4 print(type(Z))  
5 print(type(A))
```

```
<class 'str'>  
<class 'int'>  
<class 'float'>  
<class 'bool'>
```

Operadores de comparación

- Operador de igualdad (`==`)

El operador de igualdad compara si dos valores son iguales y devuelve `True` si lo son, y `False` si no lo son.

- Operador de desigualdad (`!=`)

El operador de desigualdad compara si dos valores no son iguales y devuelve `True` si son diferentes, y `False` si son iguales.

Operadores de comparación

- Mayor que (>):

Compara si el valor de la izquierda es mayor que el de la derecha.

- Menor que (<):

Compara si el valor de la izquierda es menor que el de la derecha.

- Mayor o igual que (>=):

Compara si el valor de la izquierda es mayor o igual que el de la derecha.

- Menor o igual que (<=):

Compara si el valor de la izquierda es menor o igual que el de la derecha.

Operadores Lógicos

Los operadores lógicos posibilitan la ejecución de acciones relacionadas con la lógica booleana, como la inversión, la unión y la disyunción, dentro de las condiciones y expresiones que controlan el flujo del programa.

Operador and:

El operador and devuelve True si ambas expresiones son verdaderas. Si al menos una de las expresiones es falsa, el resultado será False.

Operador or:

El operador or devuelve True si al menos una de las expresiones es verdadera. Si ambas expresiones son falsas, el resultado será False.

Operadores Lógicos

Operador not:

El operador not se utiliza para negar el valor de una expresión. Si la expresión es True, el resultado será False, y viceversa.

Ejemplos:

```
1 a = True  
2 b = False  
3 resultado = a and b  
4 print(resultado)
```

False

```
1 a = True  
2 b = False  
3 resultado = a or b  
4 print(resultado)
```

True

```
1 a = False  
2 resultado = not a  
3 print(resultado)
```

True

GRACIAS

**¡...POR SU PARTICIPACION POR EL DIA DE
HOY...!
LOS ESPERO EN LA PROXIMA CLASE**



**AVANZA A LA
SIGUIENTE CLASE**

