Contenido

[**Módulo 1: Fundamentos de Python** 3](#_Toc195283778)

[1.1 ¿Qué es Python? Instalación y configuración del entorno. 3](#_Toc195283779)

[1.2 Primeros pasos: sintaxis, comentarios y ejecución de código. 3](#_Toc195283780)

[1.3 Variables: Declaración, tipos de datos básicos (números, cadenas, booleanos) 3](#_Toc195283781)

[1.4 Operaciones básicas: Matemáticas, concatenación, operadores lógicos. 3](#_Toc195283782)

[1.5 Ejemplo práctico: 3](#_Toc195283783)

[1.6 Ejercicio: 4](#_Toc195283784)

[**Módulo 2: Control de Flujo** 4](#_Toc195283785)

[2.1 Estructuras condicionales: if, elif, else. 4](#_Toc195283786)

[2.2 Bucles: for y while. 4](#_Toc195283787)

[2.3 Operadores de comparación y lógicos. 4](#_Toc195283788)

[2.4 Ejemplo práctico: 4](#_Toc195283789)

[2.5 Ejercicio: 4](#_Toc195283790)

[**Módulo 3: Funciones y Modularidad** 4](#_Toc195283791)

[3.1 ¿Qué es una función? Declaración y uso (def). 4](#_Toc195283792)

[3.2 Parámetros y valores de retorno. 4](#_Toc195283793)

[3.3 Modularidad: Dividir proyectos en archivos manejables. 4](#_Toc195283794)

[3.4 Pruebas unitarias: Introducción a testing con unittest o pytest. 4](#_Toc195283795)

[3.5 Ejemplo práctico: 4](#_Toc195283796)

[3.6 Ejercicio: 4](#_Toc195283797)

[**Módulo 4: Colecciones y Manipulación de Datos** 5](#_Toc195283798)

[4.1 Listas: Creación, manipulación y métodos (append(), pop(), etc.). 5](#_Toc195283799)

[4.2 Tuplas: Uso y propiedades. 5](#_Toc195283800)

[4.3 Diccionarios: Claves, valores y métodos avanzados (keys(), values(), items()). 5](#_Toc195283801)

[4.4 Slicing: Manipulación de listas, cadenas y tuplas. 5](#_Toc195283802)

[4.5 JSON: Introducción y manipulación de datos estructurados. 5](#_Toc195283803)

[4.6 Ejemplo práctico: 5](#_Toc195283804)

[4.7 Ejercicio: 5](#_Toc195283805)

[**Módulo 5: Programación Orientada a Objetos (POO)** 5](#_Toc195283806)

[5.1 Clases y objetos: Definición y uso. 5](#_Toc195283807)

[5.2 Métodos y atributos. 5](#_Toc195283808)

[5.3 Herencia y polimorfismo. 5](#_Toc195283809)

[5.4 Ejemplo práctico: 5](#_Toc195283810)

[5.5 Ejercicio: 6](#_Toc195283811)

[**Módulo 6: Manejo de Errores y Archivos** 6](#_Toc195283812)

[6.1 Excepciones: Captura y manejo con try, except, finally. 6](#_Toc195283813)

[6.2 Archivos: Lectura y escritura (open(), read(), write()). 6](#_Toc195283814)

[6.3 Entornos virtuales con venv: Configuración y gestión de dependencias. 6](#_Toc195283815)

[6.4 Ejemplo práctico: 6](#_Toc195283816)

[6.5 Ejercicio: 6](#_Toc195283817)

[**Módulo 7: Bibliotecas para Análisis de Datos** 6](#_Toc195283818)

[7.1 Pandas: Creación y manipulación de DataFrames. 6](#_Toc195283819)

[7.2 NumPy: Arrays, operaciones matemáticas y manipulación matricial. 6](#_Toc195283820)

[7.3 Ejemplo práctico: 6](#_Toc195283821)

[7.4 Ejercicio: 6](#_Toc195283822)

[**Módulo 8: Visualización de Datos con Matplotlib** 7](#_Toc195283823)

[8.1 Introducción a Matplotlib: ¿Qué es y cómo instalarlo? 7](#_Toc195283824)

[8.2 Uso de pyplot: Creación de gráficos básicos. 7](#_Toc195283825)

[8.3 Personalización de gráficos: Colores, etiquetas, títulos y estilos. 7](#_Toc195283826)

[8.4 Tipos de gráficos: 7](#_Toc195283827)

[8.5 Ejemplo práctico: 7](#_Toc195283828)

[8.6 Ejercicio: 7](#_Toc195283829)

[**Módulo 9: Frameworks de Desarrollo Web** 7](#_Toc195283830)

[9.1 Flask: Crear aplicaciones web básicas y APIs REST. 7](#_Toc195283831)

[9.2 Django: Crear proyectos robustos con modelos y vistas. 7](#_Toc195283832)

[9.3 Ejemplo práctico: 7](#_Toc195283833)

[9.4 Ejercicio: 8](#_Toc195283834)

[**Módulo 10: Git y Entorno de Trabajo** 8](#_Toc195283835)

[10.1 Git: Iniciar repositorios, comandos básicos (clone, commit, push, pull). 8](#_Toc195283836)

[10.2 Uso de ramas y resolución de conflictos (branch, merge). 8](#_Toc195283837)

[10.3 Configurar requirements.txt en proyectos. 8](#_Toc195283838)

[10.4 Ejemplo práctico: 8](#_Toc195283839)

[10.5 Ejercicio: 8](#_Toc195283840)

[**Módulo 11: Proyecto Final** 8](#_Toc195283841)

[11.1 Ideas de proyectos: 8](#_Toc195283842)

[**Módulo 12: Automatización con Python** 8](#_Toc195283843)

[12.1 Introducción a la automatización: ¿Qué es y por qué es útil? 8](#_Toc195283844)

[12.2 Uso de pyautogui: 8](#_Toc195283845)

[12.3 Web Scraping con BeautifulSoup y requests: 9](#_Toc195283846)

[12.4 Selenium para automatización web: 9](#_Toc195283847)

[12.5 Ejemplo práctico: 9](#_Toc195283848)

[12.6 Ejercicio: 9](#_Toc195283849)

# **Módulo 1: Fundamentos de Python**

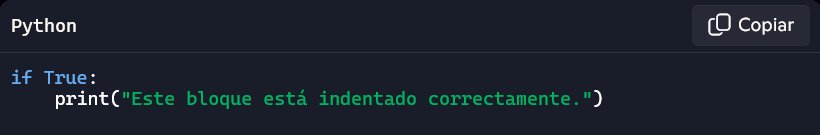
En este módulo se tiene como objetivo aprender lo esencial para empezar a programar en Python.

**1.1 ¿Qué es Python? Instalación y configuración del entorno.**

**¿Qué es Python?** Python es un lenguaje de programación versátil y fácil de aprender. Se utiliza en desarrollo web, análisis de datos, inteligencia artificial, automatización y más. Su sintaxis simple lo hace ideal tanto para principiantes como para expertos.

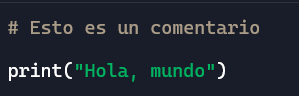
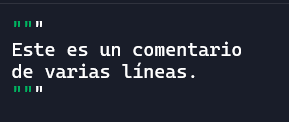
**1.2 Primeros pasos: sintaxis, comentarios y ejecución de código.**

**Sintaxis básica en Python**

* Python utiliza una **sintaxis simple y legible**.
* No necesitas usar ; al final de cada línea.
* ****Los bloques de código se identifican con **indentación** (4 espacios o tabulador).

**Comentarios**

Son útiles para explicar tu código y no se ejecutan, existen dos tipos de comentarios los que se pueden hacer en una sola línea y los que se pueden hacer en multiples líneas.

* **Comentarios de una línea:** Usa # al inicio.
* **Comentarios de varias líneas:** Usa comillas triples (""" o ''').

**Ejecucion de codigo**

1. Guarda tu archivo, por ejemplo, sintaxis.py.
2. Abre la terminal, navega hasta la carpeta donde está el archivo:

cd "C:\Users\franc\Documents\Mis proyectos de Python"

1. Ejecuta el archivo con: python sintaxis.py

1.3 Variables: Declaración, tipos de datos básicos (números, cadenas, booleanos)

1.4 Operaciones básicas: Matemáticas, concatenación, operadores lógicos.

1.5 Ejemplo práctico:

* Crear un programa que calcule la edad del usuario a partir de su fecha de nacimiento.

## 1.6 Ejercicio:

* Escribir un programa que convierta kilómetros a millas.

# **Módulo 2: Control de Flujo**

En este módulo se tiene como objetivo manejar la lógica en programas.

2.1 Estructuras condicionales: if, elif, else.

2.2 Bucles: for y while.

2.3 Operadores de comparación y lógicos.

2.4 Ejemplo práctico:

* Crear una calculadora que evalúe operaciones básicas según las opciones del usuario

## 2.5 Ejercicio:

* Escribir un programa que imprima los números primos en un rango dado.

# **Módulo 3: Funciones y Modularidad**

En este módulo se introduce la reutilización de código.

3.1 ¿Qué es una función? Declaración y uso (def).

3.2 Parámetros y valores de retorno.

3.3 Modularidad: Dividir proyectos en archivos manejables.

3.4 Pruebas unitarias: Introducción a testing con unittest o pytest.

3.5 Ejemplo práctico:

* Crear una función que calcule el área de un círculo.

3.6 Ejercicio:

* Implementar una función que convierta grados Celsius a Fahrenheit y escribir pruebas unitarias.

# **Módulo 4: Colecciones y Manipulación de Datos**

En este módulo se trabaja con estructuras de datos avanzadas.

4.1 Listas: Creación, manipulación y métodos (append(), pop(), etc.).

4.2 Tuplas: Uso y propiedades.

4.3 Diccionarios: Claves, valores y métodos avanzados (keys(), values(), items()).

4.4 Slicing: Manipulación de listas, cadenas y tuplas.

4.5 JSON: Introducción y manipulación de datos estructurados.

4.6 Ejemplo práctico:

* Crear un diccionario para gestionar inventario de productos.

## 4.7 Ejercicio:

* Trabajar con datos en formato JSON y gestionar contactos telefónicos.

# **Módulo 5: Programación Orientada a Objetos (POO)**

En este módulo se comprenden los principios básicos de la POO.

5.1 Clases y objetos: Definición y uso.

5.2 Métodos y atributos.

5.3 Herencia y polimorfismo.

5.4 Ejemplo práctico:

* Crear una clase para representar cuentas bancarias y operaciones de depósito/retiro.

## 5.5 Ejercicio:

* Implementar una clase para gestionar estudiantes con métodos avanzados (calcular promedio, etc.).

# **Módulo 6: Manejo de Errores y Archivos**

En este módulo se gestionan excepciones y se aprende a trabajar con archivos.

6.1 Excepciones: Captura y manejo con try, except, finally.

6.2 Archivos: Lectura y escritura (open(), read(), write()).

6.3 Entornos virtuales con venv: Configuración y gestión de dependencias.

6.4 Ejemplo práctico:

* Crear un programa que lea un archivo CSV y analice datos.

## 6.5 Ejercicio:

* Implementar un sistema de registro diario que guarde datos en un archivo.

# **Módulo 7: Bibliotecas para Análisis de Datos**

En este módulo se utilizan herramientas avanzadas.

7.1 Pandas: Creación y manipulación de DataFrames.

7.2 NumPy: Arrays, operaciones matemáticas y manipulación matricial.

7.3 Ejemplo práctico:

* Analizar datos de ventas usando Pandas y crear gráficos básicos

## 7.4 Ejercicio:

* Realizar operaciones matriciales y cálculos estadísticos usando NumPy.

# **Módulo 8: Visualización de Datos con Matplotlib**

En este módulo se introduce la creación de gráficos y visualizaciones con Matplotlib.

8.1 Introducción a Matplotlib: ¿Qué es y cómo instalarlo?

8.2 Uso de pyplot: Creación de gráficos básicos.

8.3 Personalización de gráficos: Colores, etiquetas, títulos y estilos.

8.4 Tipos de gráficos:

* Gráficos de líneas.
* Gráficos de barras.
* Gráficos de dispersión.
* Histogramas.

8.5 Ejemplo práctico:

* Crear un gráfico de barras para mostrar las ventas mensuales de un negocio

8.6 Ejercicio:

* Crear un gráfico de dispersión que muestre la relación entre dos conjuntos de datos (por ejemplo, horas de estudio y calificaciones).

# **Módulo 9: Frameworks de Desarrollo Web**

En este módulo se introduce el desarrollo web.

9.1 Flask: Crear aplicaciones web básicas y APIs REST.

9.2 Django: Crear proyectos robustos con modelos y vistas.

9.3 Ejemplo práctico:

Crear un formulario web y una API básica con Flask

## 9.4 Ejercicio:

* Desarrollar un sistema CRUD (Crear, Leer, Actualizar, Eliminar) con Django.

# **Módulo 10: Git y Entorno de Trabajo**

En este módulo se aprenden herramientas esenciales para la gestión de proyectos.

10.1 Git: Iniciar repositorios, comandos básicos (clone, commit, push, pull).

10.2 Uso de ramas y resolución de conflictos (branch, merge).

10.3 Configurar requirements.txt en proyectos.

10.4 Ejemplo práctico:

* Crear y colaborar en un proyecto usando GitHub

## 10.5 Ejercicio:

* Subir un proyecto a GitHub y trabajar en equipo.

# **Módulo 11: Proyecto Final**

En este módulo se consolidan los conocimientos con un proyecto completo.

11.1 Ideas de proyectos:

* Chatbot básico: Responder preguntas predefinidas.
* Gestión de inventarios: Registrar, buscar y actualizar datos.
* Análisis de datos: Visualización y estadísticas con Pandas.

# **Módulo 12: Automatización con Python**

En este módulo aprenderás cómo usar Python para automatizar tareas.

12.1 Introducción a la automatización: ¿Qué es y por qué es útil?

12.2 Uso de pyautogui:

* Controlar el mouse y el teclado con Python.
* Automatizar clics y escritura de texto

## 12.3 Web Scraping con BeautifulSoup y requests:

* Extraer información de sitios web.
* Manipular HTML para obtener datos específicos.

## 12.4 Selenium para automatización web:

* Simular interacción con un navegador.
* Completar formularios automáticamente.
* Descargar archivos de forma automática.

## 12.5 Ejemplo práctico:

* Crear un script que abra varias páginas web y extraiga información útil

## 12.6 Ejercicio:

* Diseñar un bot que complete formularios en línea automáticamente.