Aprender Python tiene mucho de arte y método, porque quieres que tu base sea sólida antes de saltar a cosas más avanzadas. Aquí te doy un **orden lógico y progresivo**, desde lo más simple hasta lo más poderoso:

**1. Conceptos básicos**

* Instalación de Python y un editor (VS Code, PyCharm, Jupyter, etc.).
* Sintaxis básica: comentarios, indentación, print() y input().
* Tipos de datos: int, float, str, bool.
* Operadores: +, -, \*, /, %, //, \*\*.

**2. Variables y estructuras simples**

* Crear, asignar y usar variables.
* Conversión de tipos: int(), float(), str().
* Operaciones básicas con cadenas y números.
* Concatenación de cadenas.

**3. Listas, tuplas y diccionarios**

* Listas: creación, acceso, modificación, métodos (append, remove, sort…).
* Tuplas: inmutables, cuándo usarlas.
* Diccionarios: claves y valores, agregar, modificar y recorrer.
* Sets (conjuntos) y su uso básico.

**4. Condicionales y lógica**

* if, elif, else.
* Operadores lógicos: and, or, not.
* Comparaciones: >, <, >=, <=, ==, !=.

**5. Bucles**

* for y while.
* break, continue, else en bucles.
* Recorrer listas, diccionarios y rangos.

**6. Funciones**

* Definir funciones con def.
* Parámetros y retorno de valores.
* Parámetros opcionales y por defecto.
* Funciones recursivas (opcional al principio).

**7. Manejo de errores**

* try, except, finally.
* Tipos de errores comunes: ValueError, TypeError, IndexError.

**8. Archivos**

* Leer y escribir archivos con open().
* Archivos CSV y su manejo con csv o pandas.
* JSON: leer y escribir.

**9. Módulos y librerías**

* Importar módulos: import math, import random.
* Crear tus propios módulos.
* Librerías esenciales: math, random, datetime, os.

**10. Programación orientada a objetos (OOP)**

* Clases y objetos.
* Métodos y atributos.
* Herencia y polimorfismo (cuando te sientas cómodo con lo básico).

**11. Temas intermedios/avanzados**

* Comprensiones de listas y diccionarios.
* Decoradores y generadores.
* Expresiones regulares (re).
* Manejo de fechas y tiempos (datetime).

**12. Proyectos prácticos**

* Calculadoras, juegos sencillos (Adivina el número, Tic-Tac-Toe).
* Mini-sistemas: inventario, agenda, notas.
* Automatización de tareas con archivos, web scraping o APIs.

Si quieres, puedo armarte un **plan de estudio semanal de Python**, con ejercicios y proyectos para cada etapa, para que lo sigas paso a paso y no te pierdas en lo intermedio.

¿Quieres que haga eso?