## **Temario Propuesto para Aprender Python**

### **Nivel Básico: Fundamentos**

* **Introducción a Python:**
  + ¿Qué es Python? Historia y aplicaciones.
  + Instalación y configuración del entorno de desarrollo.
  + Sintaxis básica: variables, tipos de datos, operadores.
  + Entrada y salida de datos.
* **Estructura de control:**
  + Condicionales: if, else, elif.
  + Bucles: for, while.
* **Funciones:**
  + Definición y llamada de funciones.
  + Parámetros y argumentos.
  + Retorno de valores.
  + Funciones recursivas.
* **Colecciones de datos:**
  + Listas: creación, acceso, modificación.
  + Tuplas: características e inmutabilidad.
  + Diccionarios: clave-valor.
  + Conjuntos: elementos únicos.
* **Módulos y paquetes:**
  + Organización del código.
  + Importar módulos.
  + Crear módulos propios.

### **Nivel Intermedio: Profundizando**

* **Programación orientada a objetos:**
  + Conceptos básicos: clases, objetos, atributos, métodos.
  + Herencia y polimorfismo.
  + Encapsulación.
* **Manejo de excepciones:**
  + Try, except, finally.
  + Creación de excepciones personalizadas.
* **Archivos:**
  + Lectura y escritura de archivos.
  + Diferentes modos de apertura.
* **Bibliotecas estándar:**
  + OS: interacción con el sistema operativo.
  + Math: funciones matemáticas.
  + Random: generación de números aleatorios.
  + Datetime: manejo de fechas y horas.
* **Introducción a bibliotecas científicas:**
  + NumPy: operaciones numéricas y matrices.
  + Pandas: análisis y manipulación de datos.
  + Matplotlib: visualización de datos.

### **Nivel Avanzado: Especialización**

* **Programación funcional:**
  + Map, filter, reduce.
  + List comprehensions.
  + Decoradores.
* **Desarrollo web con frameworks:**
  + Django o Flask: creación de aplicaciones web.
  + Bases de datos: SQL, ORM.
* **Programación concurrente:**
  + Hilos y procesos.
  + Concurrencia con asyncio.
* **Machine Learning:**
  + Scikit-learn: algoritmos de aprendizaje supervisado y no supervisado.
  + TensorFlow o PyTorch: redes neuronales profundas.
* **Desarrollo de juegos:**
  + Pygame: creación de juegos 2D.

## **Ciclos de Temas Sugeridos**

* **Ciclo 1: Fundamentos:**
  + Cubrir los temas del nivel básico.
  + Realizar ejercicios prácticos para consolidar los conocimientos.
* **Ciclo 2: Profundización:**
  + Introducir la programación orientada a objetos y el manejo de archivos.
  + Explorar bibliotecas estándar y científicas.
  + Desarrollar pequeños proyectos para aplicar los conceptos aprendidos.
* **Ciclo 3: Especialización:**
  + Elegir un área de interés (web, ciencia de datos, juegos) y profundizar en ella.
  + Trabajar en proyectos más complejos y desafiantes.

**Recomendaciones Adicionales:**

* **Práctica constante:** La mejor manera de aprender es haciendo. Resuelve ejercicios, crea pequeños proyectos y participa en comunidades de Python.
* **Recursos en línea:** Aprovecha tutoriales, cursos interactivos y documentación oficial.
* **Proyectos personales:** Define objetivos claros y trabaja en proyectos que te motiven.
* **Colaboración:** Trabaja en equipo y comparte tus conocimientos con otros.

**Plataformas Recomendadas:**

* **Codecademy:** Cursos interactivos y proyectos prácticos.
* **Coursera:** Cursos de universidades de prestigio.
* **edX:** Plataforma similar a Coursera.
* **YouTube:** Tutoriales gratuitos y canales especializados.
* **Documentación oficial de Python:** La fuente más confiable de información.

- Experiencia mínima de 2 años en el desarrollo con Java y Python, orientado a microservicios y API Rest.  
- Inglés B2 - C1  
- Gestión de bases de datos como PostgreSQL, SQL Server, y Oracle.  
- Familiaridad con entornos de nube (Preferiblemente Azure), contenedores (Docker) y orquestadores (Kubernetes).  
- Deseable experiencia en herramientas de DevOps como Jenkins, GitLab CI/CD y Azure DevOps.  
- Conocimiento en seguridad informática, estándares como OWASP, y prácticas de ciberseguridad.