

# ■ Roteiro de Estudos em Python (Projetos-Base)

## 1. Fundamentos do Python

### ■ *Conceitos:*

- Sintaxe básica (print, variáveis, tipos primitivos)
- Estruturas de controle (if, for, while)
- Funções e módulos
- Listas, tuplas, dicionários, conjuntos
- Manipulação de arquivos (open, leitura/escrita)

### ■ *Projetos sugeridos:*

- Calculadora simples no terminal
- To-do list (armazenar tarefas em .txt)
- Jogo da forca
- Gerador de senhas

## 2. Programação Estruturada e Boas Práticas

### ■ *Conceitos:*

- Funções avançadas (recursão, parâmetros opcionais, \*args/\*\*kwargs)
- Compreensões de lista/dict
- Tratamento de erros (try/except)
- Introdução a POO (classes, objetos, herança)
- Ambiente virtual (venv), pip e dependências
- Boas práticas (PEP8, organização de projetos)

### ■ *Projetos sugeridos:*

- Agenda de contatos (CRUD em arquivos)
- Simulador de banco (saque, depósito, extrato)
- Jogo estilo RPG no terminal (treino de POO)

## 3. Bibliotecas Essenciais e Manipulação de Dados

### ■ *Conceitos:*

- datetime, os, sys, json, csv
- pandas (análise de dados)
- matplotlib/seaborn (gráficos)
- requests (consumir APIs)
- Introdução a bancos de dados (sqlite3)

### ■ *Projetos sugeridos:*

- Analisador de texto (palavras mais usadas em um livro)
- Dashboard simples de finanças pessoais
- Web Scraper de notícias
- API de clima (consumindo uma API pública)

## 4. Desenvolvimento de Sistemas

### ■ **Conceitos:**

- POO mais avançada (métodos mágicos, classes abstratas)
- Arquitetura de software (MVC, SOLID aplicado em Python)
- Criação de APIs com Flask ou FastAPI
- Autenticação e JWT
- Testes (pytest)

### ■ **Projetos sugeridos:**

- API de gerenciamento de tarefas (com SQLite/PostgreSQL)
- Sistema de login com autenticação
- CRUD completo com FastAPI
- Aplicação CLI (linha de comando) para organizar arquivos

## 5. Automação e Sistemas Reais

### ■ **Conceitos:**

- Automação com selenium e pyautogui
- Scripts de organização de pastas/planilhas
- Integração com APIs de terceiros (Telegram, Discord, Google)
- Programação assíncrona (asyncio)

### ■ **Projetos sugeridos:**

- Bot do Telegram (responde mensagens)
- Automatizador de downloads (organiza arquivos em pastas)
- Web Scraper de preços + alerta no Telegram

## 6. Inteligência Artificial e Ciência de Dados

### ■ **Conceitos:**

- Revisão de matemática para IA (álgebra linear, estatística, probabilidade)
- numpy e pandas avançado
- scikit-learn (machine learning clássico)
- Redes neurais com TensorFlow/PyTorch
- NLP (processamento de linguagem natural)
- Visão computacional com OpenCV

### ■ **Projetos sugeridos:**

- Modelo de regressão (prever preços de casas)
- Classificador de e-mails (spam/ham)
- Detector de rostos com OpenCV
- Chatbot simples com NLP
- Rede neural para reconhecimento de dígitos (MNIST)

## 7. Avançado: Integração IA + Sistemas

### ■ **Conceitos:**

- Deploy de modelos (FastAPI + Docker)
- Microserviços em Python
- Integração de IA em sistemas reais
- Cloud (AWS/GCP/Azure)

### ■ **Projetos sugeridos:**

- API de IA para análise de sentimentos
- Sistema de recomendação de filmes/músicas
- Aplicativo web (Flask/Django) integrando modelo de ML
- Bot inteligente para Telegram/Discord com NLP

### ■ **Caminho Final**

Depois de passar por todas as trilhas, você vai:

- Dominar o core de Python
- Saber criar sistemas e APIs completos
- Ter base para IA, ML e NLP
- Ser capaz de integrar IA em sistemas do mundo real