Python Developer Fundamentals Roadmap

This roadmap helps you master Python development fundamentals while focusing on key software principles:  
- Clean Code  
- Readability  
- Predictability  
- DRY (Don't Repeat Yourself)

# 1. Core Python Syntax & Data Types

Goal: Write predictable, readable code from the start.

- Variables & Data Types: int, float, str, bool  
- Control Flow: if, elif, else  
- Loops: for, while  
- Basic I/O: input(), print()  
- Collections: list, tuple, dict, set  
  
Clean Code Tip:  
# Bad: a = 5  
# Good: user\_age = 5

# 2. Functions (Modularity + DRY Principle)

Goal: Avoid code repetition and increase reusability.

- def keyword  
- Parameters, default arguments, return values  
- \*args and \*\*kwargs  
  
DRY Example:  
# Bad:  
print("Welcome Alice")  
print("Welcome Bob")  
  
# Good:  
def greet\_user(name):  
 print(f"Welcome {name}")

# 3. Working with Modules and Packages

Goal: Make code predictable and clean by organizing it.

- import statements  
- Standard Library: math, os, random  
- Creating custom .py modules  
- Virtual environments (venv)  
  
Clean Project Structure:  
project/  
├── main.py  
├── utils/  
│ └── helpers.py  
└── requirements.txt

# 4. Error Handling (for Predictability)

Goal: Make your code robust and predictable.

- try-except blocks  
- Common Exceptions: ValueError, TypeError, KeyError  
- finally and else  
  
Example:  
try:  
 num = int(input("Enter a number: "))  
except ValueError:  
 print("That's not a number!")

# 5. Object-Oriented Programming (OOP)

Goal: Improve structure, avoid repetition, and increase clarity.

- Classes and Objects  
- \_\_init\_\_, self  
- Inheritance & Polymorphism  
- Encapsulation  
  
Example:  
class Car:  
 def \_\_init\_\_(self, brand):  
 self.brand = brand  
  
 def start\_engine(self):  
 print(f"{self.brand} engine started.")

# 6. Code Formatting & Style

Goal: Ensure readable and consistent code.

- Follow PEP8 style guide  
- Use formatters: black, autopep8  
- Use linters: flake8, pylint  
  
Tools:  
- VS Code Extensions: Python + Pylance + Black Formatter  
- Pre-commit hooks to auto-format code

# 7. Testing & Debugging

Goal: Ensure predictable behavior and maintainability.

- print() debugging  
- pdb (Python Debugger)  
- Unit testing: unittest, pytest  
  
Example:  
import unittest  
  
def add(a, b):  
 return a + b  
  
class TestMath(unittest.TestCase):  
 def test\_add(self):  
 self.assertEqual(add(2, 3), 5)

# 8. Version Control (Git)

Goal: Keep your project clean, safe, and manageable.

- git init, git add, git commit  
- .gitignore file  
- Write meaningful commit messages

# 9. Practice Projects

Apply clean, DRY, and predictable coding skills.

Start Simple:  
- Calculator  
- To-Do CLI app  
- Contact Book  
  
Level Up:  
- Weather CLI (with API)  
- File Organizer  
- Expense Tracker

# Summary: Think Like a Python Developer

Principle | You Practice It By...  
------------------|----------------------------  
Clean | Writing formatted, modular code  
Readability | Good naming, comments, PEP8  
Predictability | Handling errors, writing tests  
DRY | Using functions and classes wisely