Advanced Python Programming

Course Syllabus & Table of Contents

Module 1: Functional Programming in Python

1.1 Lambda Functions and Higher-Order Functions
1.2 Map, Filter, and Reduce
1.3 Closures and Decorators
1.4 Functools Module

Module 2: Metaprogramming

2.1 Dynamic Code Execution (exec, eval)
2.2 Decorators with Arguments
2.3 Metaclasses and __new__ Method
2.4 Monkey Patching

Module 3: Concurrency & Parallelism

3.1 Threading vs Multiprocessing
3.2 Global Interpreter Lock (GIL) Workarounds
3.3 Asyncio and Coroutines
3.4 Futures and Executors

Module 4: Performance Optimization

4.1 Profiling Python Code (cProfile, timeit)
4.2 Memory Management (__slots__, gc)
4.3 Cython and Just-In-Time (JIT) Compilation
4.4 NumPy for Speed

Module 5: Advanced OOP Patterns

5.1 Abstract Base Classes (ABCs)
5.2 Singleton and Factory Patterns
5.3 Dependency Injection
5.4 Context Managers (__enter__, __exit__)

Module 6: Testing & Debugging

6.1 Mocking with unittest.mock
6.2 Property-Based Testing (Hypothesis)
6.3 Debugging Memory Leaks
6.4 Logging Best Practices

Appendix

- **References**: PEP 8, Python Documentation
- **Tools Used**: PyCharm, Jupyter Notebook, Docker
- **Sample Code Repo**: github.com/demo/python-advanced *(fake link)*