Python 编程语言技术管理规范文档

一、代码风格规范

a. 强制

1. 必须使用 4 个空格进行缩进，不允许使用 Tab。
2. 每行代码不得超过 100 个字符。
3. 文件必须使用 UTF-8 编码。
4. 文件必须以 .py 为扩展名。
5. 顶级函数和类定义之间必须空两行。
6. 方法定义之间必须空一行。
7. 类名必须使用驼峰式命名风格，例如 MyClass。
8. 函数名、变量名、参数名必须使用 snake_case 风格，例如 process_data。
9. 模块名和文件名必须使用 snake_case，不允许大写字母。
10. 常量名必须全大写，并使用下划线分隔，例如 MAX_RETRIES。
11. 导入必须分组为：标准库、第三方库、自有模块，每组之间空一行。
12. 导入语句必须在文件开头，且不得在函数内部导入模块（除非必要）。
13. 不得使用通配符导入。
14. 必须遵守 PEP8 代码格式规范。
15. 必须使用显式编码声明（如必要）： # -*- coding: utf-8 -*-（适用于 Python 2 项目）。

b. 推荐

16. 使用 is 和 is not 比较单例（如 None, True, False）。
17. 使用 enumerate() 替代手动维护索引。
18. 使用列表推导代替 map() 和 filter()。
19. 使用 f-string 格式化字符串。
20. 函数参数建议加类型注解：def func(x: int) -> str。

21. 类成员变量建议加类型注解或在 __init__ 中初始化。
22. 使用 with 语句管理文件、资源。
23. 使用 logging 模块而非 print 输出日志。
24. 所有公共函数和类必须添加 docstring。
25. 复杂逻辑建议加注释，避免魔法数字。
26. 单元测试推荐使用 unittest 或 pytest。
27. 推荐将模块结构分层：如 api/, service/, utils/。
28. 推荐使用 __all__ 控制模块导出接口。
29. 多条件判断应拆解成多个 if，提升可读性。
30. 优先使用异常而非错误码处理异常逻辑。

c. 允许

31. 允许在函数内部定义嵌套函数。
32. 允许使用三元表达式，但应保持简洁：x = a if cond else b。
33. 允许使用 lambda 表达式处理简单函数，但不推荐嵌套或复杂结构。
34. 允许使用装饰器，但应清晰命名并文档化用途。
35. 允许使用 dataclasses 简化模型类。
36. 允许使用生成器（yield）优化内存开销。
37. 允许使用 try...except 捕获异常，但必须指定异常类型。
38. 允许使用 type hint 的 Optional、Union 等高级类型注解。
39. 允许引入静态检查工具如 mypy、flake8、black 进行代码质量把控。
40. 允许使用 Jupyter Notebook 进行原型验证，但正式项目必须模块化封装。