**Python 语言技术管理文档**

**·规范级别说明**

强制 (a)：违反将导致CI/CD失败，禁止合并

推荐 (b)：最佳实践，需在代码审查中遵循

允许 (c)：特殊场景可使用，但需团队审批

**一、代码风格**

(a) 缩进必须使用4个空格（禁止Tab）

(a) 每行代码长度不超过120字符（超长需换行对齐）

(a) 导入分组顺序：标准库 → 第三方库 → 本地模块（每组用空行分隔）

(b) 推荐使用Google风格文档字符串（Args/Returns/Raises）

(a) 禁止使用from module import \*（显式列出导入对象）

(b) 函数/变量名使用snake\_case，类名用CamelCase

(a) 常量必须全大写（如MAX\_CONNECTIONS = 100）

**二、编码实践**

(a) 禁止可变对象作为函数默认值（如def f(a=[])）

(a) 必须使用is/is not比较None（禁用== None）

(b) 推荐Type Hints标注类型（Python 3.6+）

(a) 异常处理需指定具体异常（禁止裸except:）

(b) 推荐使用上下文管理器管理资源（如with open()）

(a) 禁止在循环中重复计算不变表达式（如len(list)）

(b) 推荐使用列表推导式替代简单循环

(a) \_\_init\_\_.py禁止包含业务逻辑（仅作包声明）

**三、工程规范**

(a) 项目必须包含requirements.txt/pyproject.toml

(a) 测试覆盖率不低于80%（核心模块需达95%+）

(b) 推荐使用pytest替代unittest

(a) 禁止提交包含print()调试的代码

(b) 推荐使用预提交钩子（pre-commit）检查代码

(a) 模块必须有单元测试（测试文件以test\_开头）

(b) 推荐使用结构化的项目布局：

project/

├── src/ # 源码

├── tests/ # 测试

├── docs/ # 文档

└── setup.py # 安装脚本

**四、性能与安全**

(a) SQL查询必须使用参数化（禁止字符串拼接）

(b) 推荐使用f-string格式化字符串（Python 3.6+）

(a) 敏感配置禁止硬编码（必须从环境变量读取）

(b) 推荐使用连接池管理数据库/HTTP连接

(a) 用户输入必须做合法性校验

(b) 推荐使用structlog或logging替代print

**五、协作与维护**

(a) 提交信息需符合Conventional Commits规范

(b) 推荐使用Pull Request进行代码审查

(a) 弃用的接口必须标记@deprecated

(b) 推荐使用# TODO(owner)标注待办事项

(a) 禁止修改已发布的接口（需版本兼容）

六、工具链

(b) 推荐使用Black自动格式化代码

(b) 推荐使用Flake8进行静态检查

(c) 允许使用TypeGuard进行运行时类型检查

(b) 推荐使用mypy进行静态类型验证

(a) 必须使用虚拟环境隔离依赖（venv/conda）

七、高级特性

(c) 允许谨慎使用元编程（如装饰器、元类）

(b) 推荐使用dataclass替代简单POJO（Python 3.7+）