**一、代码风格与格式**

1. **（a）缩进必须用 4 个空格**，禁止使用 Tab 字符。
2. **（a）行长度不超过 100 字符**（注释/文档字符串 80 字符）。
3. **（b）运算符前后、逗号后需加空格**（如 x = y + 1）。
4. **（b）import 语句分组排序**：标准库 > 第三方库 > 本地模块，每组空一行。
5. **（c）允许单行简单语句**（如 if x: return），但复杂逻辑需换行。

**二、文档与注释**

1. **（a）公共模块/类/函数必须写文档字符串（Docstring）**，格式遵循 Google Style 或 reST。
2. **（b）注释需解释“为什么”**，而非重复代码行为。
3. **（a）文档字符串包含：功能描述、参数、返回值、异常**（示例）：

def fetch\_data(url: str) -> list:   
 """从 API 获取数据并解析为列表。   
 Args:   
 url: 请求地址，必须以 HTTPS 开头。   
 Returns:   
 解析后的 JSON 数据列表。   
 Raises:   
 ValueError: URL 协议非法时抛出。   
 """

1. **（b）为复杂算法写概念性文档**，解释设计思路。
2. **（c）允许用行内注释标注临时性代码**（如 # TODO: 重构为缓存方案）。

**三、设计原则**

1. **（a）禁止全局变量**（常量除外）。
2. **（b）单一职责原则**：函数不超过 50 行，类不超过 500 行。
3. **（a）API 设计需先写设计文档**，包含目标、方案对比、决策原因。
4. **（b）优先组合而非继承**，避免深度继承链（>3 层）。
5. **（c）允许使用 @property 封装属性访问逻辑**。

**四、性能与安全**

1. **（a）SQL 查询必须参数化**，禁止拼接字符串防注入。
2. **（b）高频路径代码避免 for 循环嵌套**（改用生成器或向量化操作）。
3. **（a）对外接口强制认证**（如 OAuth2.0/JWT）。
4. **（b）大文件处理使用流式读写**（如 with open(…, 'rb') as f）。
5. **（c）允许在非关键路径使用缓存提升性能**（如 functools.lru\_cache）。

**五、测试与质量**

1. **（a）核心逻辑必须覆盖单元测试**（覆盖率 ≥80%）。
2. **（b）测试用例按 Given-When-Then 结构组织**（示例）：

def test\_fetch\_data\_timeout():   
 # Given: 模拟超时异常   
 with patch("requests.get", side\_effect=Timeout):   
 # When: 调用被测函数   
 result = fetch\_data("https://api.example")   
 # Then: 验证返回空列表   
 assert result == []

1. **（a）代码变更需通过 CI 流水线**（构建 + 测试 + Lint）。
2. **（b）集成测试覆盖主要用户场景**（如端到端 API 调用链）。
3. **（c）允许跳过耗时测试**（用 @pytest.mark.slow 标注）。

**六、团队协作**

1. **（a）代码提交前必须 Peer Review**，重点关注设计/文档。
2. **（b）文档需明确读者类型**（新手/专家）并适配认知水平。
3. **（a）项目必须有 README.md**，包含：快速启动、部署流程、关键文档链接（落地页规范）。
4. **（b）使用 CHANGELOG.md 记录版本变更**（语义化版本格式）。
5. **（c）允许提交时自动格式化代码**（如 black --check）。

**七、维护与监控**

1. **（a）日志分级输出**（DEBUG/INFO/ERROR），错误包含堆栈。
2. **（b）关键指标埋点**（如请求延迟、错误率）。
3. **（a）废弃接口标记 @deprecated 并保留至少 2 个版本**。
4. **（b）文档随代码同步更新**，版本控制系统统一管理。
5. **（c）允许临时注释掉弃用代码**（需附删除计划日期）。

**八、工具与流程**

1. **（a）统一开发环境**：pyproject.toml 定义依赖和 Lint 规则。
2. **（b）用 Type Hints 标注类型**（Python 3.8+）。
3. **（a）设计文档评审需三种视角**：
   * **技术准确性**（资深工程师）
   * **可读性**（新人或用户）
   * **一致性**（技术作家）。
4. **（b）自动化生成 API 参考文档**（如 Sphinx + ReadTheDocs）。
5. **（c）允许使用 Jupyter Notebook 编写探索性代码**（但需转脚本上线）。
6. **（a）定期（每季度）审计文档有效性**，删除过期内容。
7. **（b）重大事故后更新 Runbook**，包含根因与应对步骤。