**a. 强制 (Mandatory)**

1. 源文件必须使用 UTF-8 编码。
2. 应统一使用 JDK 11 或团队规定的更高版本。
3. 包名全部小写，且以个人或团队名称开头。
4. 每个 public 类或接口必须放在独立的 .java 文件中，且文件名与类名一致。
5. 类名、接口名、枚举名应采用首字母大写的驼峰命名法。
6. 方法名和变量名应采用首字母小写的驼峰命名法。
7. 常量名应全部大写，单词间以下划线分隔。
8. 单行代码长度不得超过 100 个字符。
9. 左大括号应与声明在同一行，采用 K&R 风格。
10. 禁止使用通配符 import，必须显式导入类。
11. 文件末尾必须保留一个空行，方法间至少空一行。
12. I/O、数据库、网络等资源必须在 try-with-resources 或 finally 中关闭。
13. 只能捕获需要处理的具体异常类型，禁止捕获 Exception 或 Throwable。
14. 自定义业务异常必须继承自 RuntimeException 或团队基类，并提供错误码与详细信息。
15. 核心逻辑必须编写单元测试，测试覆盖率不得低于 80%。
16. 测试类应与被测类位于同一包，命名以 ClassNameTest 结尾。
17. CI 必须通过 Checkstyle、SpotBugs、PMD 等静态分析，且无严重警告。
18. 所有代码提交必须经过至少一次 Code Review，且所有意见需处理完毕。
19. 依赖管理必须使用 Maven 或 Gradle，禁止手动引入第三方 JAR。
20. 主干分支应始终保持可编译状态，禁止直接推送到 main 或 master。
21. 分支命名应遵循 feature/xxx、bugfix/xxx、release/xxx 等规范。
22. 敏感信息（如密码、密钥、Token）禁止硬编码，必须通过配置中心或环境变量读取。
23. 对外暴露的公共 API 必须提供完整的 JavaDoc，包括参数、返回值和异常说明。

**b. 推荐 (Recommended)**

1. 方法参数不宜超过五个，必要时应封装为对象或使用 Builder 模式。
2. 避免返回 null，可使用 Optional 表示可选值。
3. 优先使用 Stream、Lambda 和方法引用以提高可读性。
4. 重要场景下应添加 @Override、@Deprecated、@SuppressWarnings 等注解。
5. 适当使用工厂、单例、策略、模板方法等设计模式提高扩展性。
6. 建议采用 Controller–Service–Repository 三层架构，层间通过接口交互。
7. 错误码应统一定义并通过统一异常处理器转换为 HTTP 响应。
8. 使用 Swagger/OpenAPI 自动生成并维护 API 文档。
9. 对长期维护的模块应定期重构，每季度至少评估一次死代码和冗余依赖。
10. 入参校验应使用 Hibernate Validator 注解，避免手写校验逻辑。
11. 数据库操作应在事务内执行，使用 @Transactional 并指定 rollbackFor。
12. 对热点数据使用本地或分布式缓存（如 Caffeine、Redis），并设置合理过期策略。
13. 并发场景应使用 java.util.concurrent 下的原语，尽量避免 synchronized。
14. 当涉及用户可见文本时应使用国际化资源文件管理。
15. 对重要接口应编写集成测试或契约测试以保证稳定性。

**c. 允许 (Permitted)**

1. 在简单场景下可使用 // 单行注释进行说明。
2. 对复杂逻辑可使用 /\* … \*/ 多行注释，但需谨慎使用。
3. 在函数式场景可使用匿名内部类替代 Lambda 表达式。
4. 可以引入 Guava、Apache Commons、Lombok 等成熟开源工具包。
5. 在特殊场景下可直接使用原生 JDBC，但必须确保资源正确释放。