Java 编程技术管理规范

以下是基于 Google Java 编程风格指南（中文版）整理的 Java 技术管理规范，分为强制（a）、推荐（b）和允许（c）三个级别，共计 40 条规则。

# 一、强制（a）— 必须遵守的规则

1. 缩进：每级缩进使用 2 个空格，禁止使用制表符（Tab）。
2. 每行长度：代码行长度不得超过 100 个字符。
3. 类命名：类名使用大驼峰命名法（PascalCase），例如：MyClass。
4. 方法命名：方法名使用小驼峰命名法（camelCase），例如：myMethod。
5. 常量命名：常量名使用全大写字母，单词间以下划线分隔，例如：MAX\_SIZE。
6. 每行一条语句：每行只能写一条语句。
7. 包名：包名一律使用小写字母，禁止使用下划线。
8. 导入顺序：导入语句按以下顺序排列：标准库、第三方库、项目内部包，每组之间空一行。
9. 异常处理：禁止捕获 Exception 或 Throwable，应捕获具体异常类型。
10. 空语句块：禁止使用空的代码块，除非有明确的注释说明。
11. 注释格式：使用 // 进行单行注释，/\* \*/ 进行多行注释，/\*\* \*/ 用于 Javadoc 注释。
12. 空格使用：关键字后必须有空格，例如：if (condition)。
13. 括号位置：左大括号 { 必须与控制语句在同一行，右大括号 } 单独成行。
14. 禁止使用魔法值：应将常量定义为具名常量，避免在代码中直接使用字面值。
15. 禁止使用 System.out 和 System.err：应使用日志框架进行日志记录。
16. 禁止使用原始类型：泛型中禁止使用原始类型，例如：List 应写为 List<String>。
17. 禁止多重继承：类只能继承一个父类，但可以实现多个接口。
18. 禁止使用 finalize() 方法：应使用 try-with-resources 或显式关闭资源。
19. 禁止使用 == 比较字符串：应使用 .equals() 方法比较字符串内容。
20. 禁止使用过时的 API：应使用最新的、推荐的 API。

# 二、推荐（b）— 建议遵守的规则

1. 方法长度：方法不宜过长，建议控制在 40 行以内。
2. 类长度：类的长度应适中，建议控制在 1000 行以内。
3. 每个类一个职责：一个类应只负责一项职责，符合单一职责原则。
4. 使用 @Override 注解：重写方法时应使用 @Override 注解。
5. 使用 final 修饰不可变参数：方法参数和局部变量如果不需要修改，建议使用 final 修饰。
6. 使用 this 关键字：在类的成员变量与方法参数同名时，应使用 this 关键字区分。
7. 使用枚举代替常量类：当一组常量具有相关性时，建议使用枚举类型。
8. 使用 try-with-resources：处理资源时，建议使用 try-with-resources 语句自动关闭资源。
9. 使用 StringBuilder 拼接字符串：在循环中拼接字符串，建议使用 StringBuilder。
10. 使用日志框架：建议使用统一的日志框架，如 SLF4J，避免直接使用 System.out.println。
11. 使用 Optional 处理可能为 null 的值：避免直接返回 null，建议使用 Optional。
12. 使用 lambda 表达式和 Stream API：在合适的场景下，建议使用 lambda 表达式和 Stream API 提高代码简洁性。
13. 使用注解简化配置：在使用框架时，建议使用注解简化配置，例如 Spring 的注解。
14. 使用单元测试：建议为每个公共方法编写单元测试，确保代码质量。
15. 使用代码检查工具：建议使用静态代码分析工具，如 Checkstyle、PMD、FindBugs 等。

# 三、允许（c）— 可根据实际情况选择遵守的规则

1. 水平对齐：为了提高可读性，可以在代码中使用水平对齐，但不强制要求。
2. 空行使用：可以在逻辑相关的代码块之间添加空行，以提高代码的可读性。
3. 方法链换行：在方法链较长时，可以选择在每个点操作符后换行。
4. 使用工具生成代码：可以使用 IDE 或其他工具生成代码模板，如 getter/setter 方法。
5. 使用第三方库：在满足项目需求的前提下，可以使用经过评估的第三方库。

以上规范旨在提高代码的一致性、可读性和可维护性，建议在团队中推广实施。