实验4：MCP 应用的设计与实现

1. 实验目标

1. 理解 **Model Context Protocol (MCP)** 的基本概念和工作原理。
2. 掌握 **MCP 服务器的基本构建方法**，能实现请求与响应的交互。
3. 能够结合所学知识，自主设计一个 MCP 服务的应用场景并实现原型。

2. 实验环境

- **语言**：Python 或Nodejs
- **硬件**：普通 PC 或服务器即可

3. 实验任务

1. **理解 MCP 协议**
   - 阅读 MCP 文档，掌握基本消息结构（请求、响应、错误处理）。
   - 说明 MCP 与常见 RPC/REST 的异同点。
2. **设计服务功能**
   - 自行提出一个 MCP 服务的应用场景（例如：文本处理、信息检索、数据分析、图像识别、跨模态应用、游戏 AI 等）。
   - 服务必须至少支持一个核心方法，并能通过 MCP 接口调用。
3. **实现 MCP 服务器**
   - 搭建 MCP 协议兼容的服务端，实现与客户端的通信。
   - 支持基本的请求/响应交互，并对错误输入有合理处理。
4. **测试与演示**
   - 编写客户端脚本或工具，调用 MCP 服务进行功能验证。
   - 提供若干测试案例，展示输入、输出与实际效果。

4. 实验报告要求

1. 摘要（100–150字）：简要概述实验目标、所设计的 MCP 应用、方法实现与主要结果。

2. MCP协议原理：说明 MCP 的作用、基本通信模式。
3. 应用场景与数据
    - 应用场景：描述 MCP 服务器要实现的功能（如 问答、文本处理、系统工具等）。
    - 数据/资源说明：所用数据的来源与规模，或所调用的外部接口/服务。
    - 输入输出定义：说明用户输入的形式、期望输出的形式。
4. 实现细节
    - MCP 服务器实现
        - 工具/资源定义（对应的功能接口）。
        - 核心逻辑与关键代码。
    - 客户端对接
        - 使用的平台（如 VS Code、命令行、自定义前端）。
        - 交互流程。
    - 运行环境：操作系统、编程语言版本、MCP SDK 版本、硬件配置。
5. 功能验证与结果
    - 功能测试表：展示不同输入下的系统输出、正确性。
    - 交互展示：截图/日志，可包含成功与失败案例等。
6. 分析与讨论
    - 错误案例分析
    - 性能与可扩展性
    - 改进与扩展方向
7. 附录：关键代码片段、环境版本、可复现实验指令等。

5. 评分标准

- **实现正确性（35）**：MCP 服务器和客户端功能完整，代码清晰可复现。
- **结果与可视化（25）**：展示完整结果，成功与失败案例清晰，图表规范。
- **分析深度（20）**：能解释错误案例及影响因素，必要时进行对比分析。
- **报告质量（10）**：表达清楚，结构完整，图表标注规范。
- **加分项（10）**：应用创新或功能扩展。