AI协作编程能力报告

本报告对测试者在编程任务中与AI协作的能力进行评估

题目一

错误修复与单元测试

2/2

题目二

LLM代理服务开发

1/10

题目三

舆情监控系统开发

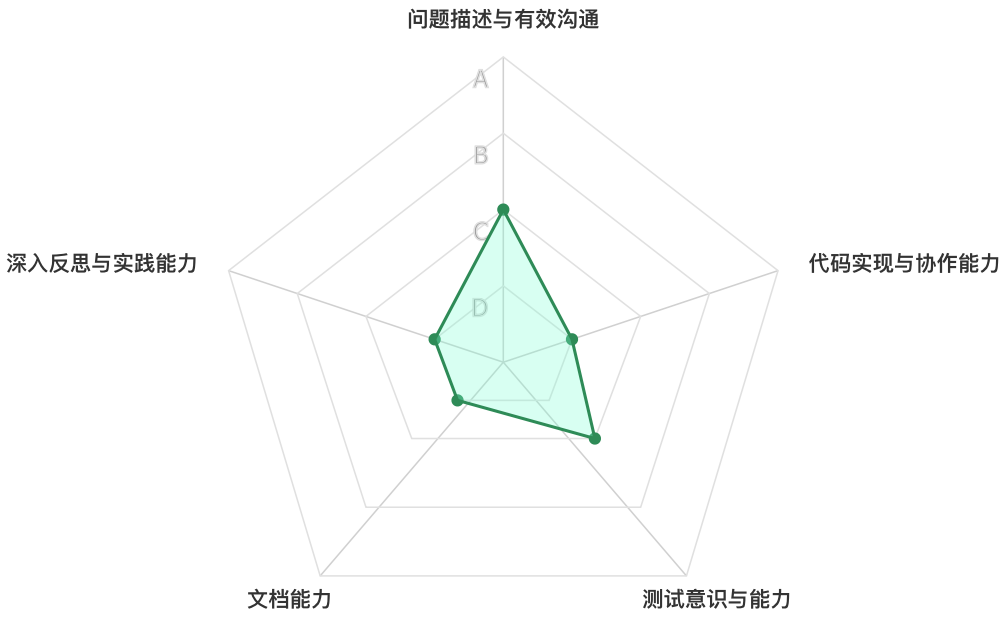
3.5/10

题目四

单元测试开发

8/10

综合能力评估



问题描述与有效沟通 (C)

准确有效地向AI描述需求和问题的能力。

用户经常直接复制原文内容，有时候会提供一些额外的价值性指导（如题目1后续有效提供了具体报错信息）。但整体以复制原文为主，缺乏清晰主动的沟通策略。

代码实现与协作指导 (D)

通过AI协作实现和优化代码的能力。

用户几乎未提供任何代码层面的额外指导或具体的修改建议。多数情况下只是复制或引用已有代码，缺乏有效互动与协作。

测试意识与能力 (C)

设计有效测试用例并正确理解测试原理的能力。

题目1体现出了良好的单元测试能力，能涵盖多种边界场景。后续题目则几乎未体现有效的测试能力或思维，属于较弱水平。

文档能力 (D)

撰写清晰易懂、实用性强的文档的能力。

用户未体现出有效撰写技术文档的能力。所有任务中几乎未提供任何有效的文档说明或指导性文档。

实践反思 (D)

能够积极反思并推动AI进行更深入实践和优化的能力。

用户几乎未体现出任何实践反思的行为。完全依赖AI的生成内容，未进行主动的深入反思或优化。

题目一：错误修复与单元测试 2/2

评估用户发言的有效性

用户的发言过程如下：

- **第一次发言：**

  > *帮我生成修复后的文件*

  这是一个不够明确的请求，未能清晰指出具体的文件或问题所在，不是有效的请求。

- **第二次发言：**

  > *我现在执行test报错了，请根据报错信息帮我找到错误，并修正*

  用户在这里提供了明确的操作信息（执行test并出现错误），但未即时提供具体的错误信息。

- **后续发言：**

  > *TS2307 [ERROR]: Cannot find module*
  > *TS2582 [ERROR]: Cannot find name 'describe'*
  > *TS2582 [ERROR]: Cannot find name 'it'*
  > *TS2304 [ERROR]: Cannot find name 'expect'*

  此时用户有效地传达了问题所在。

- **再次发言：**

  > *你给我写一个完整的输出ts代码和对应的测试用例*

  用户明确要求AI生成具体代码和测试，这一请求非常有效。

- **最后发言：**

  > *使用了上述测试用例，和main里的内容，但是执行时报错，报错内容如下，请帮我修改为可执行的*

  用户指出了实际运行中的问题，并请求AI进行修复，清晰有效。

用户行为评分（满分2分）

错误修复（1分）：

- 用户在AI的帮助下明确识别并修复了最初main.ts中存在的严重逻辑错误（即初始sum的设置错误，导致空数组访问越界）。
- 用户接受并使用了AI建议的正确修复方案（初始化为0），有效解决了问题。
- **得分：满分（1分）**

单元测试撰写（1分）：

- 用户清晰地要求AI撰写了额外的测试用例，并且包含了空数组、单元素数组、负数、以及大量数据等边界情况，覆盖度较全面。
- 用户确实执行了这些单元测试，但过程中出现了环境配置问题（Jest未正确配置），导致测试未成功执行。然而，这属于环境配置问题而非单元测试本身的问题。
- 虽然环境配置尚未成功解决，但单元测试本身已经完整撰写。
- **得分：满分（1分）**

综合得分：

错误修复：1分

撰写单元测试：1分

总分：2 / 2 分

题目二：LLM代理服务开发 1/10

评估用户发言的有效性

用户的发言过程如下：

- **第一次发言：**

  > *@llmproxy/app.py Add a while True loop to proxy_chat_completions, include the full file in your response*

  用户明确提出了具体代码修改请求，但增加无限循环并不符合README中的技术要求，未体现对题目核心需求的理解。**有效性较低**。

- **第二次发言：**

  > *[直接复制粘贴了README原文]*

  无具体问题或新要求，未体现清晰目的或与AI的有效互动。**有效性极低**。AI主动给出正确实现，但用户无反馈。

- **第三次发言：**

  > *[再次复制README原文]*

  完全重复上一步，未展现出进一步理解或互动。**有效性极低**。

用户行为评分（满分10分）

1. 成功完整实现README的要求（4分）：

- 用户未主动实现，也未体现出通过测试。AI虽然提供了初步实现方案（zero-shot），但用户未验证也未通过已有测试。
- **得分：1分**（仅实现了基本功能，未通过验证测试）

2. 用户根据AI回复指导下一步工作（4分）：

- 用户未体现出任何根据AI回复调整或指导下一步工作的能力。
- **得分：0分**

3. 用户良好实践体现（2分）：

- 用户未体现任何反思、请求AI解释或积极探索理解代码的行为。
- **得分：0分**

综合得分：

成功实现 **README** 要求：**1**分

用户有效指导**AI**工作：**0**分

用户良好实践体现：**0**分

总分：**1 / 10** 分

题目三：舆情监控系统开发 3.5/10

评估用户发言的有效性

用户的发言过程如下：

- **第一次发言：**

  > *# AI Agent 自动化舆情监控开发任务*
  > *（完整题目描述内容）*

  用户在第一次发言中只是复制了原始题目，没有明确提出要求。这种做法虽然不是最有效的提问方式，但为接下来的讨论提供了完整的上下文基础，属于有效但不够精确的发言。

- **第二次发言：**

  > *你是高级程序员，请编写一段代码实现舆情任务的开发，本程序可以监控到用户输入，拿到用户如后分析内容，给出结果。请生产代码*

  用户明确提出代码生成请求，但未具体指出任何开发的细节或对AI回复的要求，这使得AI的回复只是给出了基本的逻辑框架，而未深入实现用户自定义prompt的需求。这是一个有效的请求，但缺乏针对性和指导性。

- **第三次发言：**

  > *为了实现这个舆情监控程序，我们将创建一个 Python 脚本，按步骤执行提示词任务。在 main.py 中，我们将实现每个任务的功能，并在 test.py 中编写简单的测试用例。*
  > *实现的代码要通过下述描述生成的测试用例。（再次复制提示词详细内容）*

  这是一次重复了第一次发言内容的行为，仍然未提供更精细或具体的指导。重复详细题干，没有有效推动问题深入，没有指出前一次AI回复中的不足之处，属于低效重复发言。

用户行为评分（满分10分）

1. 可自定义prompt与实现功能（4分）：

- 用户代码实现中，没有真实调用提供的gpt()函数，只是简单的字符串匹配实现模拟功能。
- 虽然整体流程结构合理，但实现非常粗糙，且缺乏测试模式，没有体现自定义prompt的空间。
- 根据标准（实现粗糙，未使用提供的函数，未实现测试模式）
- **得分：2 / 4分**

2. 明确输出格式（1分）：

- 用户未要求GPT明确输出为JSON，而采用了简单的分行解析，稳定性不足。
- **得分：0.5 / 1分**

3. PromptDoc 中明确告知提示词工程师输出格式（1分）：

- 用户给出的 PromptDoc 仅描述了如何修改提示词，但完全未提及具体的输出格式要求。
- **得分：0 / 1分**

4. 实现提示词的串联处理（1分）：

- 用户确实有在实现中明确体现了提示词步骤的串联处理逻辑。
- **得分：1 / 1分**

5. DesignDoc 明确记录设计决策（1分）：

- 用户未提供，也未明确指导AI提供设计决策记录。
- **得分：0 / 1分**

6. 优秀实践（2分）：

（1）正确指导AI下一步代码工作（1分）

- 用户在整个交互中，并未给出明确反馈或具体的指导以帮助AI改进代码实现，未达到标准。
- 得分：0 / 1分

（2）引导AI反思（1分）

- 用户在整个过程中未曾要求AI反思代码实现或设计，未达到标准。
- 得分：0 / 1分

额外得分（1分）：

- 粉丝量级判断逻辑代码实现（0.5分）：未实现，仅有硬编码模拟。
- 得分：0 / 0.5分
- 累计性处理逻辑实现（0.5分）：未给分。
- 得分：0 / 0.5分

综合得分：

可自定义**prompt**与功能实现：**2 / 4**分

明确输出格式：**0.5 / 1**分

PromptDoc说明：**0 / 1**分

实现提示词串联：**1 / 1**分

DesignDoc设计记录：**0 / 1**分

优秀实践：**0 / 2**分

额外得分：**0 / 1**分

总分：**3.5 / 10** 分

题目四：单元测试开发 8/10

评估用户发言的有效性

用户的发言过程如下：

- **第一次发言：**

  > *用户明确复制了题目描述*

  用户意图清晰，有效。

- **第二次发言：**

  > *用户提供了实际的错误信息*

  用户提供了有效信息，但此时用户试图通过修改`test.py`来修正错误，而非识别实际问题所在（main.py逻辑缺陷），是不正确的行为。

- **第三次发言：**

  > *用户再次提出重新编写`test.py`，并要求完整覆盖测试用例*

  但还是仅聚焦于修改测试脚本，而没有真正要求修复主函数，属于误区。

- **第四次发言：**

  > *用户解决了模块导入路径问题*

  这是有效行为，但这并未改变其整体方法上的误区。

综合评估：

用户发言虽然意图明确且有效，但其实际行动未能符合题目真实要求（修正函数本身的逻辑缺陷），而持续误导性地专注于修改测试代码。

用户行为评分（满分10分）

1. 测试用例覆盖程度（满分4分）：

- 用户的单元测试设计明确、完整，涵盖了所有必要场景（空列表、单元素、分组为1、整除、不整除、超大分组、0和负数情况）。
- **得分：4分（满分）**

2. 测试用例的解释合理性（满分2分）：

- 用户对于每个测试用例的解释清晰，边界条件和异常情况分析到位。
- 明确说明了为什么每个测试都需要包含。
- **得分：2分（满分）**

3. 常见的Good Practice（满分2分）：

用户展现了如下good practice：

- 提供了单元测试执行结果（实际报错）。
- 让AI反思并重新设计更完善的单元测试方案。

- **得分：2分（满分）**

4. 测试意识（满分2分）：

- 用户始终试图通过修改`test.py`的方式解决函数逻辑本身的问题（如分组大小为负数未抛异常、最后一组未正确返回），而不是修改`main.py`本身。这种行为完全违背了单元测试的初衷。
- 未表现出理解"测试是为了发现函数问题并修改函数"的正确思维。
- **得分：0分（满分2分）**

关键扣分原因：

- 用户严重误解了单元测试的目的，未修改存在问题的函数`main.py`，而错误地坚持修改`test.py`以掩盖问题。

综合得分：

测试覆盖程度：4分

测试用例解释合理性：2分

常见Good Practice：2分

测试意识：0分

总分：8 / 10 分