实践三说明_通过LLM生成测试用例输出

> 注意：本次实践需要在**实践二完成之后**方可进行。

主要任务

本次实践的主要任务是模拟在没有标准代码的情况下，通过将测试用例输入（实践二得到的）与功能说明注释交给大语言模型，让其生成一系列的测试用例输出，以评估大语言模型在解析能力与计算能力，同时连同实践二的内容一起，探索该借助大语言模型生成完整测试用例集的方法的可行性。

而大语言模型基于测试用例输入与功能说明注释所生成的输出并不一定是正确的，因此需要检查其生成的输出（即以实践一所转换的 Cpp 代码运行测试用例输入来核对其输出是否正确），并将正确的输出作为最终的测试用例输出，也请记录大语言模型生成的测试用例输出的准确率，由于大语言模型的不稳定性，在评估准确率时，应用相同的提示语多次向大语言模型提问（无上下文，开新的对话框提问才算为多次），取平均值，本次实践要求每份代码生成测试用例输出时至少重复提问 3 次，方可评估准确率。

本次实践的成果要求与前两次实践类似，为各份代码各个输入所对应的测试用例输出（由大语言模型生成的多份 + 由程序跑出来的一份），与实验报告。实验报告需包含你所设计的提示语（prompt）与大语言模型对话的截图（两三张即可），并记录下大语言模型生成的测试用例输出的准确率（请按照上述要求重复生成后再计算准确率）。

请将每份代码（含测试用例输出）与实验报告（pdf格式）打包，压缩包名称与邮件主题名称均为“**实践三 <学号> <姓名>**”，通过邮箱发送至 sakiyary@smail.nju.edu.cn。

举例

以下仍旧以 `008_sum_product.cpp` 举例（下简称 `008`），在成果中期望得到的测试用例输出的格式如下：

```cpp
// other codes ...

std::tuple<int, int> sum_product(const std::vector<int>& numbers) {
    /*
    For a given list of integers, return a tuple consisting of a sum and a
product of all the integers in a list.
    Empty sum should be equal to 0 and empty product should be equal to 1.
    */

    // other codes ...
}

int main() {
    std::vector<std::vector<int>> cases = {{}, {1}, {1, 2}};
    std::tuple<int, int> result;

    // 以下为 LLM 根据 cases 与 commment 生成的测试用例输出，可能并不正确
    std::vector<std::tuple<int, int>> llm_outs_1 = {{0, 1}, {1, 1}, {3, 3}};
    std::vector<std::tuple<int, int>> llm_outs_2 = {{0, 1}, {1, 1}, {3, 2}};
    std::vector<std::tuple<int, int>> llm_outs_3 = {{0, 1}, {1, 1}, {3, 2}};
    // 以下为程序运行 cases 生成的测试用例输出，一定是正确的，用于检验 LLM 生成的测试用例输出的准确率
    std::vector<std::tuple<int, int>> expected_outs = {{0, 1}, {1, 1}, {3, 2}};
```

```
    for (auto i = 0; i < cases.size(); i++) {
        result = sum_product(cases[i]);
        assert(result == expected_outs[i]);
        assert(result == llm_outs_1[i]);
        assert(result == llm_outs_2[i]);
        assert(result == llm_outs_3[i]);
        // 由于上述假设的 LLM 第一次生成的第三个测试用例输出不正确，所以运行到  i == 3 程序会异常退出
        // 那么这份测试用例输出的生成准确率为 8/9 = 88.9%
    }
}
```