**ChatGPT辅助软件单元测试用例生成**

我们尝试让基于GPT3.5的ChatGPT来为一段代码生成单元测试用例。

# 直接生成

我们用图1中所示的提示模版让ChatGPT为给定的被测代码片段生成单元测试。



图1：直接生成

ChatGPT生成测试在正确率上并不很理想，只有42.1%用例可通过编译，24.8%的用例可正确执行。但其生成的测试用例实现的测试覆盖率十分接近人工编写的水准，并**具有良好的可读性。**

# 两步骤生成

将测试生成的任务分解成了两个子任务：首先通过intention prompt让 ChatGPT理解被测代码的意图；然后通过generation prompt，令ChatGPT基于被测代码的意图为其生成单元测试。

在图2中，通过intention prompt，ChatGPT首先正确生成了被测函数“setCharAt()”的意图描述；然后通过generation prompt，ChatGPT生成了带有正确断言的测试（即“assertEquals(Hello Jorld, strBuilder.toString())”）。

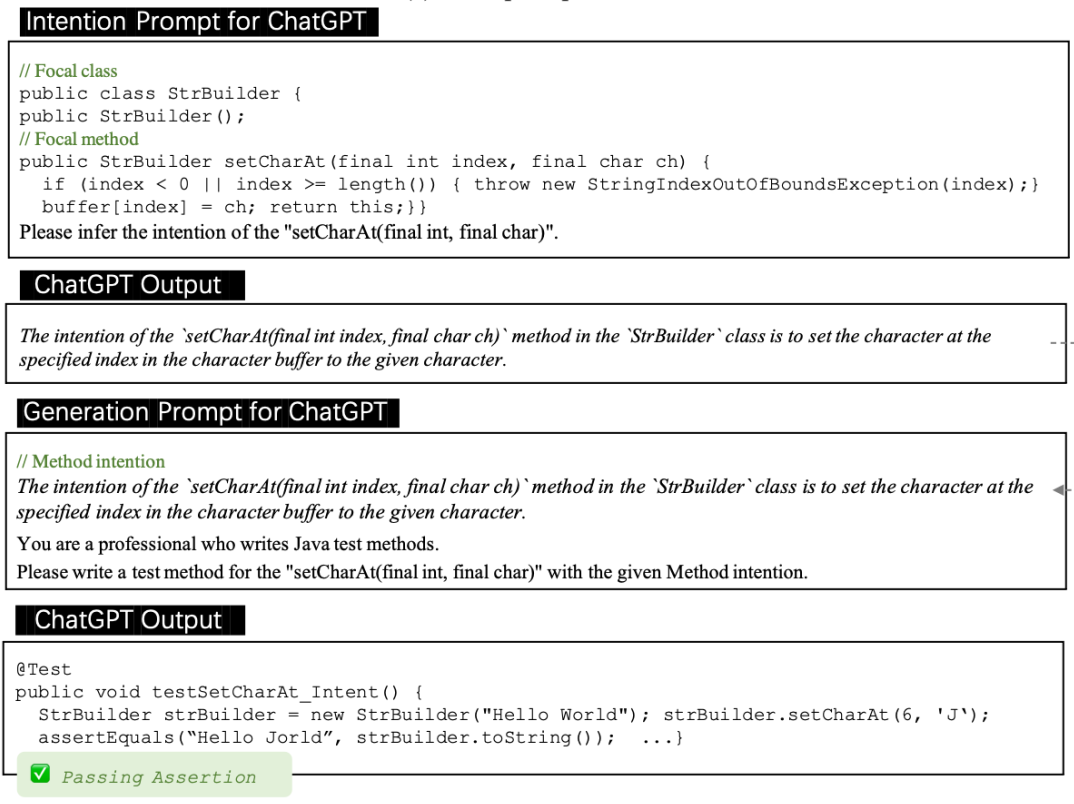


图2：两步生成

和直接生成相比，采用两步生成的测试用例正确率有了明显提升，50+%用例可通过编译，30%的用例可正确执行。

# 迭代生成

通过迭代的方式修复初始测试生成器所生成测试中的编译错误。每次迭代都会依次执行两个步骤：(i)首先通过在验证环境中编译生成的测试来验证测试的正确性；（ii）其次根据编译过程中的错误消息和与编译错误相关的额外代码上下文构建提示。将构建好的提示输入到ChatGPT当中，以获得更加完善的测试。该修复过程不断重复直到生成的测试能够成功编译或达到最大迭代次数。图3展示了如何在两次迭代中修复了测试用例中的编译错误的例子。

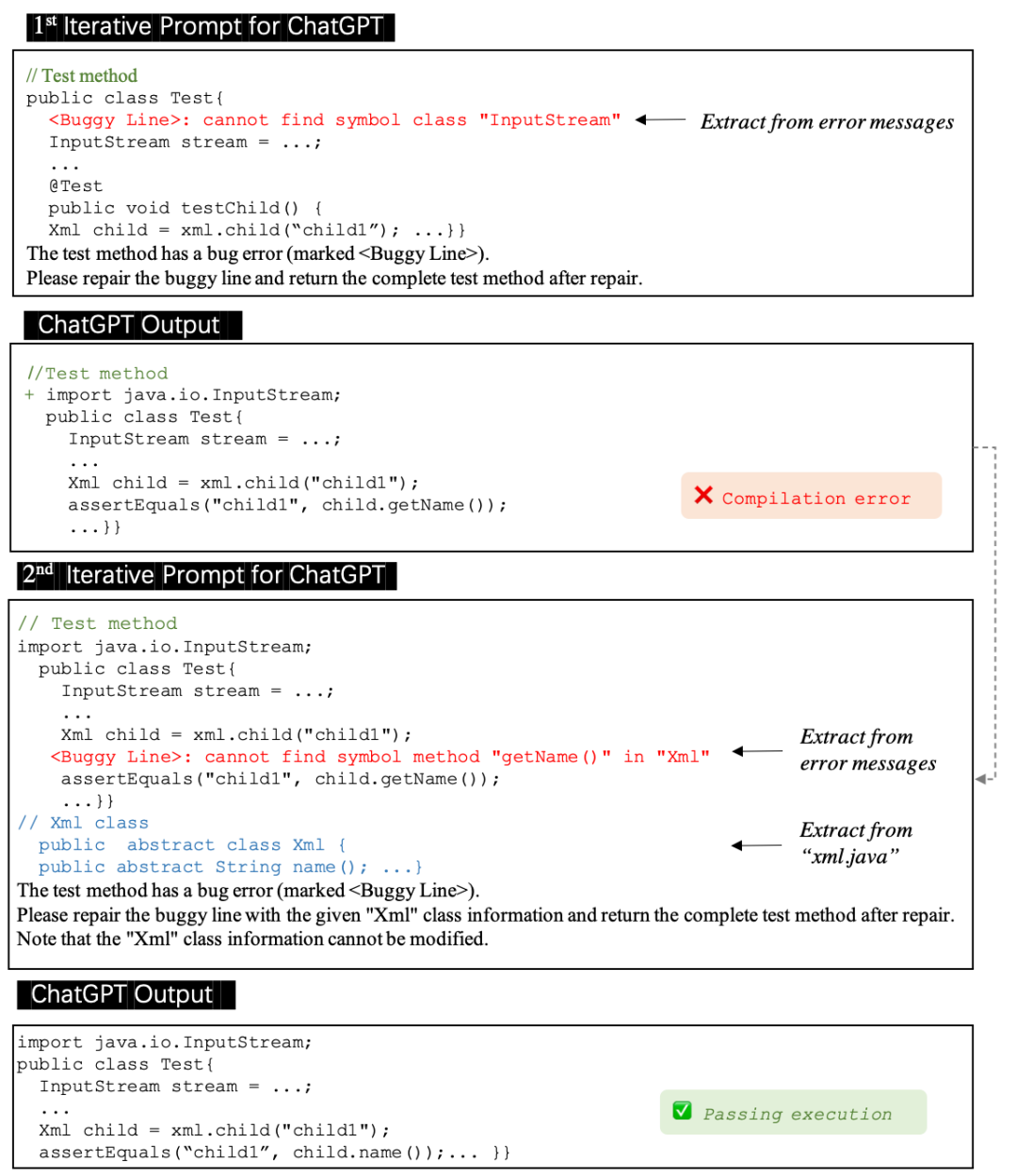


图3 迭代生成

采用迭代生成的测试用例正确率又有了提升，70+%用例可通过编译，40+%的用例可正确执行。