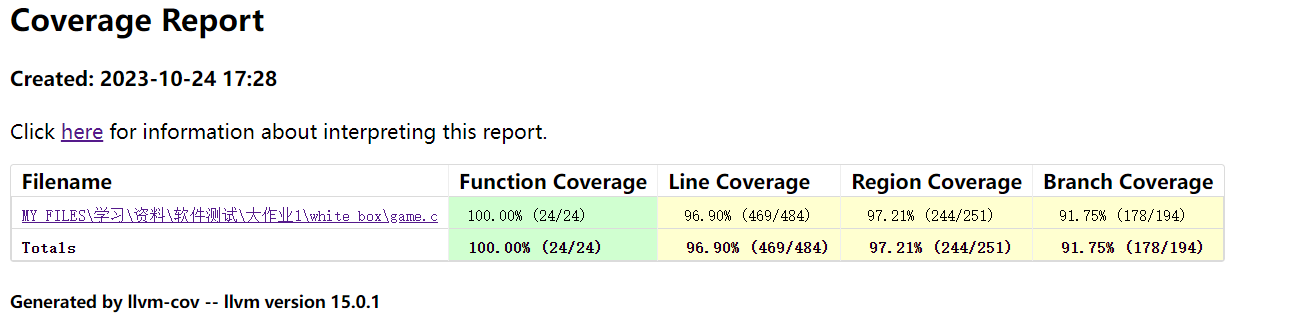
**白盒测试测试报告**

一、整体说明

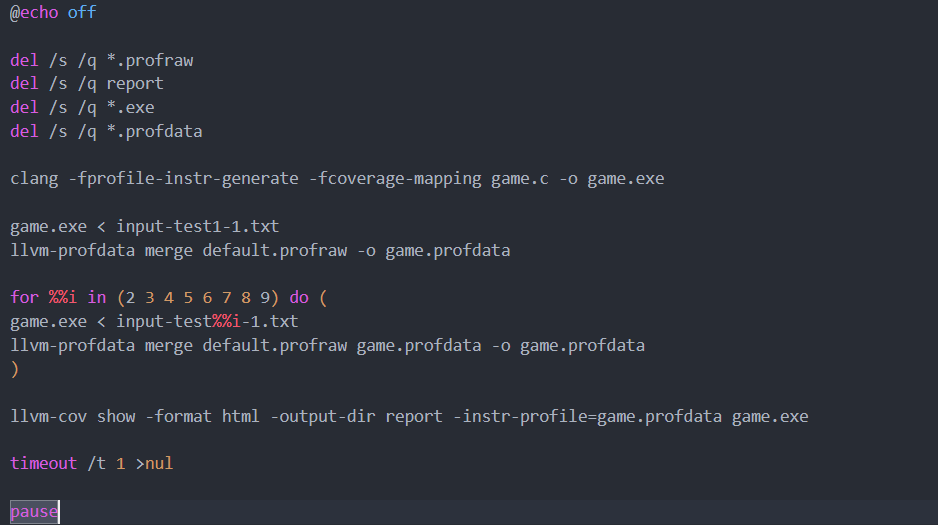
本次白盒测试一共设计了9个测试用例，分别为：input-test1-1.txt，input-test2-1.txt，input-test3-1.txt，input-test4-1.txt，input-test5-1.txt，，input-test6-1.txt，，input-test7-1.txt，，input-test8-1.txt，，input-test9-1.txt

白盒测试使用了Clang + llvm-cov 工具统计测试的覆盖率，得到的

采用了gcov工具，测试用例中input1.txt的内容最多，最终得到的三种覆盖率详细结果如图：

二、结果分析

1. 工具使用说明

运行测试用例脚本test.bat代码如下：

**@echo off**: 这行命令用于关闭命令提示符窗口中的命令回显，即不在执行命令时显示命令本身。

**del**：命令用于删除文件夹中的文件

**clang -fprofile-instr-generate -fcoverage-mapping game.c -o game.exe**这条命令使用 Clang 编译器编译名为 "game.c" 的源代码文件，并生成一个名为 "game.exe" 的可执行文件，并启用代码覆盖率收集和映射功能。-fprofile-instr-generate和-fcoverage-mapping是编译器选项，用于启用代码覆盖率收集和映射功能。

**game.exe < input-bug1-1.txt**这行命令运行"game.exe" 可执行文件，分别使用名为 "input-bug1-1.txt"输入文件作为输入。

**llvm-profdata merge -o game.profdata default.profraw**: 这行命令将之前生成的 ".profraw" 文件合并为一个名为 "game.profdata" 的文件，这个文件包含了多次执行的代码覆盖率的信息。

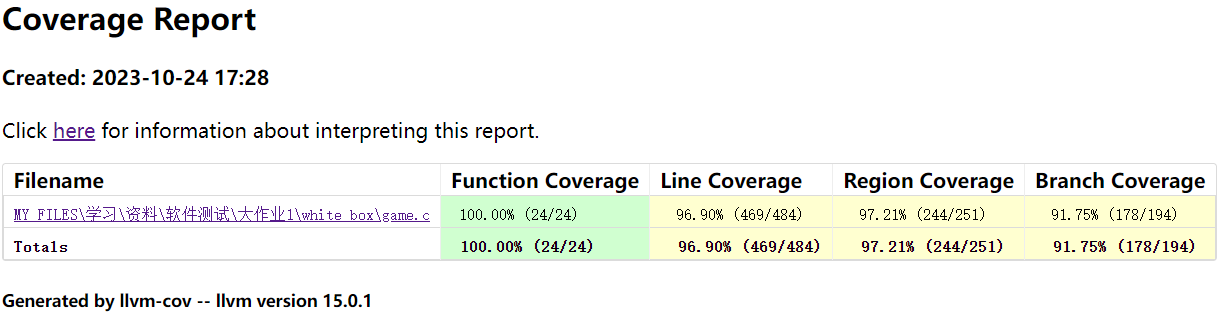
**llvm-cov show -format html -output-dir report -instr-profile=game.profdata game.exe** 这行命令使用 "game.profdata" 文件生成一个代码覆盖率报告，报告以 HTML 格式输出到名为 "report" 的文件夹中。

运行测试用例脚本test.bat执行完毕后会生成default.profraw与game.profdata，这两个文件都是与代码覆盖率相关的文件，用于存储代码执行的覆盖信息。

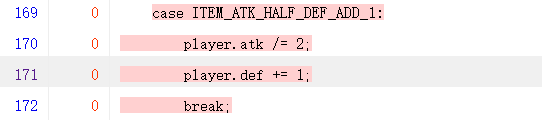
* default.profraw文件：记录了程序执行期间每个代码块（如函数、语句、分支等）的执行情况，包括哪些代码块被执行、执行次数等信息。
* game.profdata文件：它包含了从多个default.profraw文件中提取的覆盖率信息，并对数据进行聚合和汇总，以便生成更全面的代码覆盖率报告。

2. 对覆盖率报告的解释

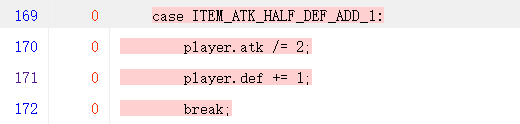
1. 整体报告各项指标的含义

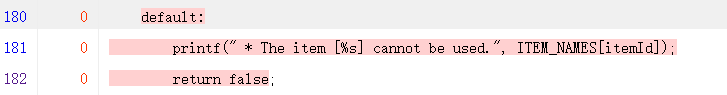
下图所示为整体测试覆盖率结果：

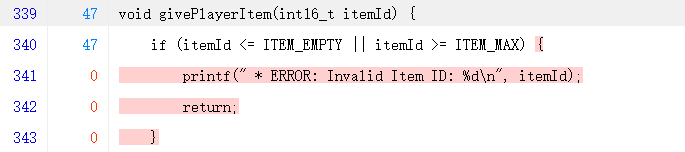
共有四种覆盖率结果，分别为：Lines，Functions，Branches和Region。其中函数覆盖率达到了100%，语句覆盖率（Lines Coverage）达到了96.9%，分支覆盖率（Branches Coverage）达到了91.75%，区域覆盖（Region Coverage）达到了97.21%。

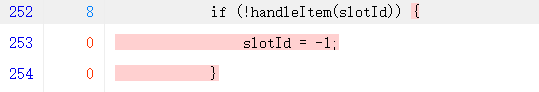
1. 详细报告中各项指标的含义

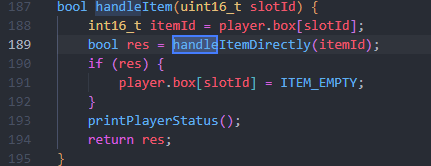
如上图所示，在详细报告中，第一列为代码行数，第二列为代码执行次数。而没有被执行过的代码会用红色的背景颜色标识出来。

1. 未覆盖代码分析

该语句未覆盖是因为代码中无法向玩家提供该道具，自然在游戏中玩家也无法使用该道具导致使用道具判定结果不可能是ITEM\_ATK\_HALF\_DEF\_ADD\_1

该语句未覆盖是因为调用该语句函数bool handleItemDirectly(int16\_t itemId)被调用传入的参数 itemId 能够被所有case处理，导致default代码无法执行。

该语句未覆盖是因为函数

该语句未覆盖是因为函数handleItem()的返回值一定是true，具体函数如下：

handleItem()的返回值即函数handleItemDirectly()返回值，而在该函数中，只有可能在swtich语句的default中返回false，之前分析过该函数无法执行default中的代码，因此该函数返回值只有true，综上所述handleItem()的返回值一定是true导致 slotId = -1 无法执行, handleItemDirectly()函数如下图所示：

1. 总结

在本次白盒测试中，我采用了提高语句覆盖率为目标的测试用例设计方法。我使用了clang编译器和llvm-cov工具来生成覆盖率报告。这种方法旨在通过增加测试用例中的测试内容，尽可能涵盖C文件中的语句。

在本次测试中，我注意到白盒测试在发现错误方面有一定的不足。例如，本次测试用例未涵盖黑盒测试中的几个错误情况。尽管我在已有的测试用例的基础上增加了一些测试用例，例如测试"DEF = 0"导致程序崩溃的情况和输入名称时导致内存篡改的情况，但这些BUG对于使用clang和llvm-cov工具生成的覆盖率报告没有产生影响。因此，综合使用黑盒和白盒测试方法比单独使用一种方法更为有效。