ESOF 322 Executive Summary – A3 – Matthew Rohrlach

1 APPROACH AND TOOLS

Verification was performed by instantiating test objects of the requisite classes. The test objects were run through a coverage of use cases. After each action, an assertSame() call was used to verify the correctness of the action (relative to an expected, known result).

Verification tests were created with code coverage in mind. Using the JaCoCoverage plugin for Netbeans, created by the EclEmma team, coverage was ensured on almost all the branches and instructions of the methods tested.

2 NUMBER OF TESTS

Door.java – One method tested – 10 assertSame calls in enter() to test player location and status during actions

Player.java – Three methods tested – 1 assertSame call to test status in go(), 7 in pickUp() to test inventory, 11 calls in drop() to test inventory

Room.java – Four methods tested – 9 assertSame calls to test room contents in addItem(), 6 to test room contents in removeItem(), 1 to test player status in enter(); 2 calls to test player status in exit()

3 COVERAGE OF PATHS

Coverage of instructions was completed for all tested methods. Coverage of branches was completed in all methods tested, with the exception of Player.drop(). The drop method employs a switch that operates on an input integer of value 1 or 2. An if statement prevents other inputs from being used in the method. This prevents the third branch of the switch from ever seeing use.

All coverage data is available in the .jacocoverage/report.html folder of the JAR file, and also submitted separately. The file to open is ‘index.html’. An XML file is part of the inclusion.

4 FAILURES OBSERVED

There were no failures identified during JUnit testing of the program.