Sam Yu and Mufaro Nyereyemhuka

Professor Wumpus Project

CS 1632 - DELIVERABLE 1: Test Plan and Traceability Matrix

Our test plan for the Professor Wumpus program consists of 26 test cases for the 11 requirements. Our plan tests for a few edge cases, and it found four software defects. The edge cases were conceived from our personal experiences in working in computer science. We looked for off-by-one errors, null pointers, index-out-of-bound errors, and other similar errors.

We extensively used a seed value of “3” for many of the latter requirements’ test cases because this value immediately placed the TA directly east of the student. Seed value 3 also placed the professor four rooms south of the student. These placements made tracking the TA and professor easier for several rounds of testing.

Some immediate difficulties we anticipated were testing for the TA’s movement and Wumpus’s location because both of them are invisible on the display. Another difficulty we encountered after reading through the requirements was the overlap in some pre- and postconditions of different test cases due to their similarities. We tried to distinguish the test cases enough so that each case provides a unique functionality test. We were concerned with how different each test case needed to be from each other and how many test cases would suffice for some of the requirements, but we followed the directions to the best of our ability to provide this test plan.

Test Cases

IDENTIFIER: SIX-BY-SIX-WALLS-TEST

DESCRIPTION: Verify that the game displays a 6 by 6 matrix of rooms at each iteration  
PRECONDITIONS: A room exists to the east of the student, and the professor is not in the room directly east of the student  
EXECUTION STEPS: Press ‘E’ to move east once

POSTCONDITIONS: A new 6 by 6 matrix is displayed

IDENTIFIER: LOCATION-OF-STUDENT-TEST

DESCRIPTION: Verify that the location of the student (indicated as an S) is visible at each iteration  
PRECONDITIONS: The first iteration began with an S indicating the student’s default location, and the professor is not in the room directly west of the student  
EXECUTION STEPS: Enter “W” to move west once

POSTCONDITIONS: The S indication of the student’s location is visible in the matrix

IDENTIFIER: CORRECT-INPUT-TEST1

DESCRIPTION: Verify that the program accepts an input of N, S, E, W and rejects all other input

PRECONDITIONS: Matrix and cardinal direction options are displayed, the program is waiting for user input, and the professor is not in the room directly south of the student  
EXECUTION STEPS: Enter “S” to move south one time

POSTCONDITIONS: A new matrix is displayed, and the program is waiting for the next user input

IDENTIFIER: CORRECT-INPUT-TEST2

DESCRIPTION: Verify that the program accepts an input of N, S, E, W and rejects all other input

PRECONDITIONS: Matrix and cardinal direction options are displayed, the program is waiting for user input, and the professor is not in the room directly east of the student  
EXECUTION STEPS: Enter “E” to move east one time

POSTCONDITIONS: A new matrix is displayed, and the program is waiting for the next user input

IDENTIFIER: INCORRECT-INPUT-TEST

DESCRIPTION: Verify that the program rejects invalid input and prompts the user for correct input

PRECONDITIONS: Matrix and cardinal direction options are displayed, and the program is waiting for user input  
EXECUTION STEPS: Enter “NSEW”

POSTCONDITIONS: “Please enter N,S,E, or W” is displayed

IDENTIFIER: UPPERCASE-INPUT-TEST1

DESCRIPTION: Verify that an input of “S” properly displays a new matrix that includes the student’s location

PRECONDITIONS: Matrix and cardinal direction options are displayed, and the program is waiting for user input  
EXECUTION STEPS: Enter “S” to move south one time

POSTCONDITIONS: A new matrix is displayed, showing the student’s location

IDENTIFIER: LOWERCASE-INPUT-TEST1

DESCRIPTION: Verify that an input of “s” properly displays a new matrix that includes the student’s location

PRECONDITIONS: Matrix and cardinal direction options are displayed, and the program is waiting for user input  
EXECUTION STEPS: Enter “s” to move north one time

POSTCONDITIONS: A new matrix is displayed, showing the student’s location

IDENTIFIER: LOWERCASE-INPUT-TEST2

DESCRIPTION: Verify that an input of “n” properly displays a new matrix that includes the student’s location

PRECONDITIONS: Matrix and cardinal direction options are displayed, and the program is waiting for user input  
EXECUTION STEPS: Enter “n” to move north one time

POSTCONDITIONS: A new matrix is displayed, showing the student’s location

IDENTIFIER: ROOM-EXISTS-TEST1

DESCRIPTION: Verify that the student moves to the room they indicated if the room exists

PRECONDITIONS: A room exists to the north of the student  
EXECUTION STEPS: Enter “N” to move north one time

POSTCONDITIONS: A new matrix is displayed with the student (marked by the letter S) in the room directly north of the previous room

IDENTIFIER: ROOM-EXISTS-TEST2

DESCRIPTION: Verify that the student moves to the room they indicated if the room exists and the TA

PRECONDITIONS: A room exists to the west of the student  
EXECUTION STEPS: Enter “W” to move west one time

POSTCONDITIONS: A new matrix is displayed with the student (marked by the letter S) in the room directly west of the previous room

IDENTIFIER: ROOM-NOT-EXIST-TEST1

DESCRIPTION: Verify that the the game indicates the user cannot move in the direction that the user specifies

PRECONDITIONS: A room does not exist to the north of the student  
EXECUTION STEPS: Press “N”

POSTCONDITIONS: A message is displayed telling the user that there is a wall, and the matrix is displayed with the student in the same cell

IDENTIFIER: ROOM-NOT-EXIST-TEST2

DESCRIPTION: Verify that the the game indicates the user cannot move in the direction that the user specifies. This tests an edge case as it is at the end of both arrays

PRECONDITIONS: The student is located in the bottom right hand corner

EXECUTION STEPS: Press “E”

POSTCONDITIONS: A message is displayed telling the user that there is a wall, and the matrix is displayed with the student in the same cell

IDENTIFIER: VALID-NEGATIVE-SEED-TEST

DESCRIPTION: Verify that the program accepts a negative 32-bit signed integer seed argument from the command line (This is an edge case because it analyzes the outcome from the extreme minimum acceptable integer)

PRECONDITIONS: The command line is open and the user is in the proper file directory, and the command “java -jar profwumpus.jar ” is written on the line  
EXECUTION STEPS: Enter “−2147483648” as an argument for the seed value

POSTCONDITIONS: The first iteration began, and “−2147483648” is displayed as the seed value

IDENTIFIER: VALID-POSITIVE-SEED-TEST

DESCRIPTION: Verify that the program accepts a positive 32-bit signed integer seed argument from the command line (This is an edge case because it analyzes the outcome from the extreme minimum acceptable integer)

PRECONDITIONS: The command line is open and the user is in the proper file directory. The command, “java -jar profwumpus.jar ” is written on the line.   
EXECUTION STEPS: Enter “2147483647” as an argument for the seed value

POSTCONDITIONS: The first iteration began, and “2147483647” is displayed as the seed value

IDENTIFIER: NO-SEED-ARGUMENT-TEST

DESCRIPTION: Verify that the program generates its own seed value for random numbers if the user does not specify one (This is an edge case because an empty argument is still considered a valid input)

PRECONDITIONS: The command line is open and the user is in the proper file directory. The command, “java -jar profwumpus.jar” is written on the line  
EXECUTION STEPS: Start the program without entering a seed value as an argument

POSTCONDITIONS: The first iteration began, and the seed value generated by the program is displayed

IDENTIFIER: INVALID-SEED-TEST1

DESCRIPTION: Verify that system ignores the invalid value that is past the bounded limits of a 32-bit signed integer value

PRECONDITIONS: The command line is open and the user is in the proper file directory. The command, “java -jar profwumpus.jar ” is written on the line  
EXECUTION STEPS: Enter “2147483648” as a seed value argument

POSTCONDITIONS: The first iteration began, and the seed value generated by the program is displayed.

IDENTIFIER: INVALID-SEED-TEST2

DESCRIPTION: Verify that the system properly ignores the invalid seed argument of an alphabet letter

PRECONDITIONS: The command line is open and the user is in the proper file directory. The command, “java -jar profwumpus.jar ” is written on the line  
EXECUTION STEPS: Enter “abc” as a seed value argument

POSTCONDITIONS: The first iteration began, and the seed value generated by the program is displayed

IDENTIFIER: TA-MOVES-EVERY-ITERATION-TEST

DESCRIPTION: Verify that the TA attempts to move every iteration. The TA moves west and even after confronting the student moves west again. This is a corner case as this test case shows TA continues to move even after confronting the student

PRECONDITIONS: “3” is used to as the seed argument, the student is in the default position in the upper left-hand corner, and the TA has spawned one cell to the east of the player  
EXECUTION STEPS: Enter “S” to move south one time, then enter “N” to move north one time

POSTCONDITIONS: The student saw the TA and fled in terror. The TA hit into the western wall.

IDENTIFIER: TA-HITS-WALL-TEST

DESCRIPTION: Verify that if the TA attempts to move to room that does not exist the TA does not move

PRECONDITIONS: “3” is used to as the seed argument, the student is in the default position in the upper left-hand corner, and the TA has spawned one cell to the east of the player

EXECUTION STEPS: Enter “E” to move east one time

POSTCONDITIONS: A message is displayed that states the TA hit into the northern wall

IDENTIFIER: WUMPUS-MOVEMENT-TEST

DESCRIPTION: Verify that Professor Wumpus never moves from his lecture cell.

PRECONDITIONS: “3” is used to as the seed argument, and the student has moved south three times.  
EXECUTION STEPS: Enter “E” to move south one time, Enter “S” to move south one time

POSTCONDITIONS: The student never ran into Professor Wumpus and the end game never displayed, showing Wumpus has not moved from his cell

IDENTIFIER: FOUND-ASSIGNMENT-TEST

DESCRIPTION: Verify that the user is notified of finding the assignment when they move into the cell with the assignment

PRECONDITIONS: “3” is used to as the seed argument, and the user has moved south three times and east once  
EXECUTION STEPS: Enter “E” to move east one time

POSTCONDITIONS: Display a message stating that the user found the assignment

IDENTIFIER: WIN-CONDITION-TEST

DESCRIPTION: If the user moves into the cell with Professor Wumpus and has the assignment, he is notified he has won and the program ends

PRECONDITIONS: “3” is used to as the seed argument, the student has moved south three times and east twice, and the student has found the assignment  
EXECUTION STEPS: Enter “W” twice to move west twice, then enter “S” to move south once.

POSTCONDITIONS: A message stating that the user won is displayed

IDENTIFIER: LOSE-CONDITION-TEST

DESCRIPTION: If the user moves into the cell with Professor Wumpus and does not have the assignment, he is notified he has lost and the program ends.

PRECONDITIONS: “3” is used to as the seed argument, and the user has moved south three times  
EXECUTION STEPS: Enter “S” to move south once

POSTCONDITIONS: A message stating that the user lost is displayed

IDENTIFIER: WUMPUS-NOTIFIES-PONTIFICATING-TEST

DESCRIPTION: Verify that the program detects when the student is one cardinal direction away of Professor Wumpus and displays a pontification message each time

PRECONDITIONS: “3” is used to as the seed argument and the student has moved south two times  
EXECUTION STEPS: Enter “S” to move south one time, enter “E” to move east time, enter “S” to move south one time

POSTCONDITIONS: The pontification message is displayed for each southern movement, but not for the eastern movement

IDENTIFIER: RUSTLING-PAPERS-NOTIFICATION-TEST

DESCRIPTION: Once the program detects that the student and TA are one cardinal direction apart the program sends the user a notification of shuffling papers.

PRECONDITIONS: “3” is used to as the seed argument, the notification of rustling papers is immediately given. The TA will move west on the next iteration  
EXECUTION STEPS: Enter “S” to move south once

POSTCONDITIONS: The notification of shuffling papers is displayed to the user

IDENTIFIER: TA-ENCOUNTER-TEST

DESCRIPTION: Verify that the program detects when the student and TA encounter one another and causes the user to move to a different, random room.

PRECONDITIONS: “3” is used to as the seed argument, and the notification of rustling papers is immediately given  
EXECUTION STEPS: Enter “N” to move north once

POSTCONDITIONS: The student fled in terror and is now in a different, “random” room (This seed causes the student to move to the bottommost-and-rightmost

Traceability Matrix

**MATRIX-DISPLAY-REQ:** SIX-BY-SIX-TEST, LOCATION-OF-STUDENT-TEST

**CARDINAL-INPUT-REQ:** CORRECT-INPUT-TEST1, CORRECT-INPUT-TEST2, INCORRECT-INPUT-TEST

**CASE-INSENSITIVE-REQ:** UPPERCASE-INPUT-TEST, LOWERCASE-INPUT-TEST1, LOWERCASE-INPUT-TEST2

**STUDENT-MOVEMENT-REQ:** ROOM-EXISTS-TEST1, ROOM-EXISTS-TEST2

**NO-ROOM-REQ:** ROOM-NOT-EXIST-TEST1, ROOM-NOT-EXIST-TEST2

**VALID-SEED-REQ:** VALID-NEGATIVE-SEED-TEST, VALID-POSITIVE-SEED-TEST, NO-SEED-ARGUMENT-TEST

**INVALID-SEED-REQ:** INVALID-SEED-TEST1, INVALID-SEED-TEST2

**TA-MOVEMENT-REQ:** TA-MOVES-EVERY-ITERATION-TEST, TA-HITS-WALL-TEST, WUMPUS-MOVEMENT-TEST

**WIN-LOSS-REQ:** FOUND-ASSIGNMENT-TEST, WIN-CONDITION-TEST, LOSE-CONDITION-TEST

**WUMPUS-REQ:** WUMPUS-NOTIFIES-PONTIFICATING-TEST

**TA-REQ:** RUSTLING-PAPERS-NOTIFICATION-TEST, TA-ENCOUNTER-TEST

Defects

SUMMARY: Out of bounds seeds are not handled as specified by the requirements.

DESCRIPTION: For test case INVALID-SEED-TEST1 of requirement INVALID-SEED-REQ, when the seed “2147483648” is tested, the program does not ignore the out of bounds seed, nor does it generate a pseudorandom 6x6 matrix based on the Random class in Java. Instead, four exceptions are thrown and the program crashes.

REPRODUCTION STEPS: While in the correct file directory of the command prompt or terminal, type “java -jar profwumpus.jar 2147483648” into the command line and hit enter

EXPECTED BEHAVIOR: “Welcome to Professor Wumpus” should be displayed with a 6x6 matrix and student S in a default location. The message, “Playing seed X” should be shown with the program’s generated seed replacing X.

OBSERVED BEHAVIOR: “Welcome to Professor Wumpus” displays, then a series of exceptions are thrown and program ends.

java.lang.NumberFormatException.forInputString(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at ProfWumpus.main(ProfWumpus.java:358)

SUMMARY: The user attempts to move east into a nonexistent cell, and the program crashes.

DESCRIPTION: When running test case ROOM-NOT-EXIST-TEST2 of requirement NO-ROOM-REQ, the student S is located in the bottom right hand corner and attempts to move east. After pressing enter, it immediately throws an index out of bounds exception.

REPRODUCTION STEPS: Position the student S at the bottommost-and-rightmost corner and move the player east.

EXPECTED BEHAVIOR: The student should not move and a notification of a wall should display.

OBSERVED BEHAVIOR: The program immediately throws an index out of bound error

java.lang.ArrayIndexOutOfBoundsException: 5

at ProfWumpus.moveStudent(ProfWumpus.java:51)

at ProfWumpus.playGame(ProfWumpus.java:335)

at ProfWumpus.main(ProfWumpus.java:362)

SUMMARY: Alphabetical input for the seed value argument causes the program to crash.

DESCRIPTION: When running test case INVALID-SEED-TEST1 of requirement INVALID-SEED-REQ, the seed “abc” is inputted. The program does not ignore the invalid seed to generate a pseudorandom 6x6 matrix based on the Random class in Java. Instead, four exceptions are thrown and the program crashes.

REPRODUCTION STEPS: While in the correct file directory of the command prompt or terminal, enter “java -jar profwumpus.jar abc” into the command line

EXPECTED BEHAVIOR: “Welcome to Professor Wumpus” should be displayed with a 6x6 matrix and student S in a default location. The message, “Playing seed X” should be shown with the program’s generated seed replacing X.

OBSERVED BEHAVIOR: “Welcome to Professor Wumpus” displays, then four exceptions are thrown and program ends.

java.lang.NumberFormatException: For input string: "abc"

at java.lang.NumberFormatException.forInputString(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at java.lang.Integer.parseInt(Unknown Source)

at ProfWumpus.main(ProfWumpus.java:358)

SUMMARY: Alphabetical input for the seed value argument causes the program to crash.

DESCRIPTION: When running test case TA-ENCOUNTER-TEST of requirement TA-REQ, the student is in the topmost-and-leftmost room with the TA in the room directly to the east.

REPRODUCTION STEPS: While in the correct file directory of the command prompt or terminal, enter “java -jar profwumpus.jar 3” into the command line. Press “N” to try to move forward to no avail because of the wall.

EXPECTED BEHAVIOR: “There’s a wall there, buddy!” displays, and a new matrix represents the TA moving to the west by one spot, causing the student to flee into a different, random room

OBSERVED BEHAVIOR: “There’s a wall there, buddy!” displays, but the student is still in the same room in the new matrix, and there is no indication of hearing shuffling papers.