

**CS 1632 Software Quality Assurance**

**Deliverable 1**

Member 1 Name: Nicholas Hicks

Member 2 Name: Dane Halle

1. **Introduction**

We had some concerns when approaching this assignment. The biggest issue is that the game can run theoretically forever and there is a potential for bugs to pop up after some large number of commands are called. With there being a large amount of actions, a user can do at any given moment, it makes it difficult to account for all cases, and thus, check for cases that may cause a defect with a specific set of actions done in a certain order.

We considered the tests we used to test each requirement once or twice. Just about each requirement has one or two Test cases meant to test whatever that requirement calls for. However, some cases fulfill other requirements as well leading to them testing multiple requirements at a time.

1. **Traceability Matrix**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | FUN-ITERATION | FUN-UNKNOWN-COMMAND | FUN-INPUT-CAPS | FUN-MOVE | FUN-WIN | FUN-LOSE | FUN-INVENTORY | FUN-LOOK | FUN-HELP | FUN-UNIQ-ROOM | FUN-UNIQ-ROOM-FURNISHING |
| TEST-ROOM-FURNISHING | x |  | x | x |  | x |  |  |  | x | x |
| TEST-SOUTH-NO-DOOR | x |  | x | x |  | x |  |  |  |  |  |
| TEST-NORTH-NO-DOOR | x |  | x | x |  | x |  |  |  |  |  |
| TEST-ITERATION-INPUT-CAPS | x |  | x | x |  | x | x | x | x |  |  |
| TEST-UNKNOWN-COMMAND | x | x | x | x |  | x |  |  |  |  |  |
| TEST-WIN | x |  | x | x | x | x | x | x |  |  |  |
| TEST-ITERATION-UNDER-HUNDRED-HIGH | x |  | x | x |  | x |  | x |  |  |  |
| TEST-ONLY-CREAM | x |  | x | x |  | x | x | x |  |  |  |
| TEST-ONLY-COFFEE | x |  | x | x |  | x | x | x |  |  |  |
| TEST-ONLY-SUGAR | x |  | x | x |  | x | x | x |  |  |  |
| TEST-CREAM-COFFEE | x |  | x | x |  | x | x | x |  |  |  |
| TEST-COFFEE-SUGAR | x |  | x | x |  | x | x | x |  |  |  |
| TEST-CREAM-SUGAR | x |  | x | x |  | x | x | x |  |  |  |

1. **Test Cases**

ID: TEST-ROOM-FURNISHING

TEST CASE: Test the FUN-UNIQ-ROOM and FUN-UNIQ-ROOM-FURNISHING requirements

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231""

EXECUTION STEPS: 1) Start the program.

2) Take note of room adjective and furnishing.

3) Type "N" and press enter and confirm that listed room adjective and furnishings aren't repeated.

4) Repeat steps 2 and 3 until you reach the end of the hallway (no door to the north).

5) Repeat typing "S" and enter until you get to the beginning of the house.

6) Type "N" and enter and check that each room has the same unique adjective describing it and same unique furnishing it did before.

7) Type "D" and enter to exit program.

POSTCONDITIONS: All rooms have A unique adjective and unique furnishing in it while traversing though the house. Program shall state user has lost and exit with error code 1.

ID: TEST-SOUTH-NO-DOOR

TEST CASE: Test the FUN-MOVE by attempting to go south even when no door exists

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". After program has started, user typed "S" and enter.

EXECUTION STEPS: 1) Type "S" and Enter

2) Type "D" and enter to exit the program

POSTCONDITIONS: The execution step 1 shall be disallowed with a message "A door in that direction does not exist." Program shall state user has lost and exit with error code 1.

ID: TEST-NORTH-NO-DOOR

TEST CASE: Test the FUN-MOVE by attempting to go north even when no door exists

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". After program has started, user typed "N" and enter five (5) times.

EXECUTION STEPS: 1) Type "N" and Enter

2) Type "N" and Enter

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "N" and Enter

6) Type "D" and enter to exit the program

POSTCONDITIONS: The execution step 1 shall be disallowed with a message "A door in that direction does not exist." Program shall state user has lost and exit with error code 1.

ID: TEST-ITERATION-INPUT-CAPS

TEST CASE: Test the FUN-ITERATION and FUN-INPUT-CAPS, and thus FUN-INVENTORY, FUN-LOOK, and FUN-HELP by attempting use all commands available to the user in both upper case and lower case.

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "N" and Enter

2) Type "S" and Enter

3) Type "L" and Enter

4) Type "I" and Enter

5) Type "H" and Enter

6) Type "D" and Enter to exit the program

7) Run the program.

8) Type "n" and Enter

9) Type "s" and Enter

10) Type "l" and Enter

11) Type "i" and Enter

12) Type "h" and Enter

13) Type "d" and Enter to exit the program

POSTCONDITIONS: Every command shall result in some meaningful response from the program and no command shall result in the resposne of "What?". Program shall state user has lost and exit with error code 1.

ID: TEST-UNKNOWN-COMMAND

TEST CASE: Test the FUN-UNKNOWN-COMMAND by attempting to use commands now allowed by FUN-ITERATION.

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "word" and Enter

2) Type "word with a space" and Enter

3) Type "null" and Enter

4) Type "Nothing" and Enter

5) Type "someone" and Enter

6) Type "Something" and Enter

7) Type "interesting" and Enter

8) Type "Into" and Enter

9) Type "Label" and Enter

10) Type "lanyard" and Enter

11) Type "Do" and Enter

12) Type "hold" and Enter

13) Type "Hand" and Enter

14) Type "s space" and Enter

15) Type "n space" and Enter

16) Type "i space" and Enter

17) Type "l space" and Enter

18) Type "d space" and Enter

19) Type "S space" and Enter

20) Type "N space" and Enter

21) Type "I space" and Enter

22) Type "L space" and Enter

23) Type "D space" and Enter

24) Type "" and Enter

25) Type "d" and Enter to exit the program

POSTCONDITIONS: Every command entered other than that of execution step 25 shall result in the response, "What?". Program shall state user has lost and exit with error code 1.

ID: TEST-WIN

TEST CASE: Test the FUN-WIN by collecting all objects required to win

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "L" and Enter

2) Type "I" and Enter to confirm cream has been picked up

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "L" and Enter

6) Type "I" and Enter to confirm coffee has been picked up

7) Type "N" and Enter

8) Type "N" and Enter

9) Type "N" and Enter

10) Type "L" and Enter

11) Type "I" and Enter to confirm sugar has been picked up

12) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has won and exit with error code 0.

ID: TEST-ONE-ITEM-LOOK

TEST CASE: Test the FUN-LOOK by attempting to pick up an item already picked up

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "L" and Enter

2) Type "I" and Enter to confirm any object added to inventory shows up

3) Type "L" and Enter

4) Type "I" and Enter

2) Type "D" and enter to exit the program

POSTCONDITIONS: The collected item shall not be collected a second time. Program shall state user has lost and exit with error code 1.

ID: TEST-ONLY-CREAM

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having cream

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "L" and Enter

2) Type "I" and Enter to confirm cream has been picked up

3) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

ID: TEST-ONLY-COFFEE

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having coffee

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "N" and Enter

2) Type "N" and Enter

3) Type "L" and Enter

4) Type "I" and Enter to confirm coffee has been picked up

5) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

ID: TEST-ONLY-SUGAR

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having sugar

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "N" and Enter

2) Type "N" and Enter

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "N" and Enter

6) Type "L" and Enter

7) Type "I" and Enter to confirm sugar has been picked up

8) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

ID: TEST-CREAM-COFFEE

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having cream and coffee

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "L" and Enter

2) Type "I" and Enter to confirm cream has been picked up

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "L" and Enter

6) Type "I" and Enter to confirm coffee has been picked up

7) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

ID: TEST-COFFEE-SUGAR

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having coffee and sugar

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "N" and Enter

2) Type "N" and Enter

3) Type "L" and Enter

4) Type "I" and Enter to confirm coffee has been picked up

5) Type "N" and Enter

6) Type "N" and Enter

7) Type "N" and Enter

8) Type "L" and Enter

9) Type "I" and Enter to confirm sugar has been picked up

10) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

ID: TEST-CREAM-SUGAR

TEST CASE: Test the FUN-LOSE by attempting to Drink while only having cream and sugar

PRECONDITIONS: When `java -version` is run, system output says "java version "1.8.0\_231"". Program is running.

EXECUTION STEPS: 1) Type "L" and Enter

2) Type "I" and Enter to confirm cream has been picked up

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "N" and Enter

6) Type "N" and Enter

7) Type "N" and Enter

8) Type "L" and Enter

9) Type "I" and Enter to confirm sugar has been picked up

10) Type "D" and enter to exit the program

POSTCONDITIONS: Program shall state user has lost and exit with error code 1.

1. **Defects**

IDENTIFIER: BUG-SOUTH-NO-DOOR

SUMMARY: When selecting ‘S’ to go south when there is not a door an incorrect message is displayed

DESCRIPTION: Failure when ‘S’ is selected to go south when there is not a door to the south. The system replies with the incorrect message, “You are in a magical land! But you are returned to the beginning!” while it should be “A door in that direction does not exist”

REPRODUCTION STEPS: 1) Type "S" and Enter

2) Type "D" and enter to exit the program

EXPECTED BEHAVIOR: Display “A door in that direction does not exist” and then exit the program with error code 1.

OBSERVED BEHAVIOR: Displays “You are in a magical land! But you are returned to the beginning!” and then exits the program with error code 1.

IDENTIFIER: BUG-NORTH-NO-DOOR

SUMMARY: When selecting ‘N’ to go north when there is not a door an incorrect message is displayed

DESCRIPTION: Failure when ‘N’ is selected to go north when there is not a door to the north. The system replies with the incorrect message, “You are in a magical land! But you are returned to the beginning!” while it should be “A door in that direction does not exist”

REPRODUCTION STEPS: 1) Type "N" and Enter

2) Type "N" and Enter

3) Type "N" and Enter

4) Type "N" and Enter

5) Type "N" and Enter

6) Type "D" and enter to exit the program

EXPECTED BEHAVIOR: Display “A door in that direction does not exist” and then exit the program with error code 1.

OBSERVED BEHAVIOR: Displays “You are in a magical land! But you are returned to the beginning!” and then exits the program with error code 1.

IDENTIFIER: BUG-INPUT-UPPER-H

SUMMARY: When ‘H’ is entered for help, the program displays “What?”, instead of the help menu.

DESCRIPTION: Failure when ‘H’ is selected to open the help menu as the program instead displays ‘What?’, and no help menu is displayed.

REPRODUCTION STEPS: 1) Type "H" and Enter

6) Type "D" and enter to exit the program

EXPECTED BEHAVIOR: Display the help menu and then exit the program with error code 1.

OBSERVED BEHAVIOR: Displays “What?” and then exits the program with error code 1.

IDENTIFIER: BUG-INPUT-LOWER-H

SUMMARY: When ‘h’ is entered for help, the program displays “What?”, instead of the help menu.

DESCRIPTION: Failure when ‘h’ is selected to open the help menu as the program instead displays ‘What?’, and no help menu is displayed.

REPRODUCTION STEPS: 1) Type "h" and Enter

6) Type "D" and enter to exit the program

EXPECTED BEHAVIOR: Display the help menu and then exit the program with error code 1.

OBSERVED BEHAVIOR: Displays “What?” and then exits the program with error code 1.

IDENTIFIER: BUG-INPUT-LOWER-N

SUMMARY: When ‘n’ is entered to go north, the program displays “What?”, instead of going north

DESCRIPTION: Failure when ‘n’ is selected to go north in the house as the program instead displays ‘What?’, and no does not move north.

REPRODUCTION STEPS: 1) Type "n" and Enter

6) Type "D" and enter to exit the program

EXPECTED BEHAVIOR: Move north and then exit the program with error code 1.

OBSERVED BEHAVIOR: Displays “What?” and then exits the program with error code 1.