Coffee Maker Quest Test Plan

Introduction

We knew right off the bat that we cannot fully test this program using only the “happy path” because of the fact that every iteration (or room) of the game will take different inputs and might give different outputs. This will most likely result in having to make a test case for the different input commands (N, S, L etc.) for each different room. So, for a program that has 6 rooms for example, we’ll have to make 6 different test cases with the same inputs and just change the pre-conditions (the user’s room) for each test case to test for that single requirement. One problem we came across was trying to understand the difference between the “Test cases” and “Tested implicitly” fields in the traceability matrix.

Requirements

REQ 1 - FUN-ITERATION - At each iteration of the game, the user shall be able enter one of six commands - "N" to go North, "S" to go South, "L" to Look for items, "I" for Inventory, "H" for Help, or "D" to Drink.

REQ 2 - FUN-UNKNOWN-COMMAND - If a player enters a command not specified by FUN-ITERATION, the system shall respond with the phrase "What?".

REQ 3 - FUN-INPUT-CAPS - The system shall be case-insensitive in regards to input values; that is, it shall accept capital and lower-case letters and treat them as equivalent.

REQ 4 - FUN-MOVE - The system shall allow a player to move North only if a door exists going North, and South only if a door exists going South.

REQ 5 - FUN-WIN - The player shall win the game if and only if Coffee, Sugar, and Cream have been collected by the player and then drunk.

REQ 6 - FUN-LOSE - The player shall lose the game if and only if the player Drinks but has not collected all of the items (Coffee, Sugar, and Cream).

REQ 7 - FUN-INVENTORY - Upon entering "I" for inventory, the player shall be informed of the items that he/she has collected (consisting of Coffee, Sugar, and Cream).

REQ 8 - FUN-LOOK - Upon entering "L" for Look, the player shall collect any items in the room and those items will be added to the player's inventory.

REQ 9 - FUN-HELP - Upon entering "H" for Help, the player shall be shown a listing of possible commands and what their effects are.

REQ 10 - FUN-UNIQ-ROOM - Each room in the house shall have a unique adjective describing it.

REQ 11 - FUN-UNIQ-ROOM-FURNISHING - Each room in the house shall have one and only one unique furnishing visible to the user upon entering the room.

Test Cases

Test1Req1

Ensure that when the user is in the first room, the user can input all the commands: “N”, “S”, “L”, “I”, “H”, and “D”.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Small room”.

Input Values: “N”, “S”, “L”, “I”, “H”, and “D”.

Execution Steps: Enter “H”, enter “I”, enter “L”, enter “S”, enter “N”, and then enter “D”

Output Values: Strings.

Post-Conditions: The system allows for the user’s input commands and terminates after the “D” command.

Test2Req1

Ensure that when the user is in the second room, the user can input all the commands: “N”, “S”, “L”, “I”, “H”, and “D”.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Funny room”.

Input Values: “N”, “S”, “L”, “I”, “H”, and “D”.

Execution Steps: Enter “H”, enter “I”, enter “L”, enter “S”, enter “N”, and then enter “D”

Output Values: Strings.

Post-Conditions: The system allows for the user’s input commands and terminates after the “D” command.

Test3Req1

Ensure that when the user is in the third room, the user can input all the commands: “N”, “S”, “L”, “I”, “H”, and “D”.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Refinanced room”.

Input Values: “N”, “S”, “L”, “I”, “H”, and “D”.

Execution Steps: Enter “H”, enter “I”, enter “L”, enter “S”, enter “N”, and then enter “D”

Output Values: Strings.

Post-Conditions: The system allows for the user’s input commands and terminates after the “D” command.

Test1Req2

Ensure that the system responds with the string “What?” when the user inputs “X” while in any room.

Pre-Conditions: None.

Input Values: “X”.

Execution Steps: Enter “X”.

Output Values: The string “What?”

Post-Conditions: None.

Test2Req2

Ensure that the system responds with the string “What?” when the user inputs “Y” while in any room.

Pre-Conditions: None.

Input Values: “Y”.

Execution Steps: Enter “Y”.

Output Values: The string “What?”

Post-Conditions: None.

Test3Req2

Ensure that the system responds with the string “What?” when the user inputs “Z” while in any room.

Pre-Conditions: None.

Input Values: “Z”.

Execution Steps: Enter “Z”.

Output Values: The string “What?”

Post-Conditions: None.

Test1Req3

While in any room, ensure that the system gives the same response to upper and lower case “L”.

Pre-Conditions: None.

Input Values: “L” and “l”.

Execution Steps: Enter “L” then enter “l”.

Output Values: One or two lines of strings.

Post-Conditions: None.

Test2Req3

While in any room, ensure that the system gives the same response to upper and lower case “I”.

Pre-Conditions: None.

Input Values: “I” and “i”.

Execution Steps: Enter “I” then enter “i”.

Output Values: Three lines of strings.

Post-Conditions: None.

Test1Req4

While in the first room, ensure that the user cannot move South.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Small room”.

Input Values: “S”.

Execution Steps: Enter “S”.

Output Values: A string.

Post-Conditions: The user is moved to the first room.

Test2Req4

While in the last room, ensure that the user cannot move North.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Rough room”.

Input Values: “N”

Execution Steps: Enter “N”.

Output Values: A string.

Post-Conditions: The user is moved to the first room.

Test1Req5

Ensure that the user wins the game if they have collected coffee, cream, and sugar then entered the drink command.

Pre-Conditions: Coffee, cream, and sugar must be present in the user inventory.

Input Values: “D”.

Execution Steps: Enter “D”.

Output Values: A bunch of lines of strings.

Post-Conditions: The user wins the game and the system terminates.

Test1Req6

Ensure that the user loses the game if they have not collected coffee, cream, and sugar then entered the drink command.

Pre-Conditions: The user inventory must not contain coffee, cream, and sugar together.

Input Values: “D”.

Execution Steps: Enter “D”.

Output Values: A bunch of lines of strings.

Post-Conditions: The user loses the game and the system terminates.

Test1Req7

Ensure that entering the inventory command “I” will inform the user of what items (coffee, cream, sugar) they have collected.

Pre-Conditions: None.

Input Values: “I”.

Execution Steps: Enter “I”.

Output Values: Three lines of strings describing which items the user has and which he doesn’t.

Post-Conditions: None.

Test1Req8

While the user is in a room that has an item, ensure that entering the look command “L” will look for and add any items in the current room to the user’s inventory.

Pre-Conditions: The user must be in a room where the first line of text provided by the system reads “Small room”, “Refinanced room”, or “Rough room”.

Input Values: “L”.

Execution Steps: Enter “L”.

Output Values: Two lines of string describing the item found in the room.

Post-Conditions: The item found is added to the user’s inventory.

Test2Req8

While the user is in a room that does not have an item, ensure that entering the look command “L” informs the user that there are no items in the room and doesn’t add anything to the user’s inventory.

Pre-Conditions: The user must be in a room where the first line of text provided by the system reads “Funny room”, “Dumb room”, or “Bloodthirsty room”.

Input Values: “L”.

Execution Steps: Enter “L”.

Output Values: One line of string telling the user there are no items found.

Post-Conditions: None.

Test1Req9

Ensure that entering the help command “H” provides the user with a list of possible commands and their effects.

Pre-Conditions: None.

Input Values: “H”.

Execution Steps: While in any room, enter “H”.

Output Values: The string “What?”.

Post-Conditions: None.

Test1Req10

While the user is in the first room, keep moving North by entering “N” ensuring that the descriptive word that is preceding the word “room” in the first line of text provided by the system is different for each different room.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Small room”.

Input Values: “N”.

Execution Steps: Enter “N”. Do this six times.

Output Values: Strings describing each room.

Post-Conditions: None.

Test1Req11

While the user is in the first room, keep moving North by entering “N” ensuring that the second line of text provided by the system describing a furnishing visible in the room is unique for every room.

Pre-Conditions: The user must be in the room where the first line of text provided by the system reads “Small room”.

Input Values: “N”.

Execution Steps: Enter “N”. Do this six times.

Output Values: Strings describing each room.

Post-Conditions: None.

Traceability Matrix

Required Identifiers	Reqs Tested	REQ 1	REQ 2	REQ 3	REQ 4	REQ 5	REQ 6	REQ 7	REQ 8	REQ 9	REQ 10	REQ 11
Test Cases	18	3	3	2	2	1	1	1	2	1	1	1
Tested Implicitly	16	0	0	0	2	0	3	4	4	3	0	0
Test1Req1	1	X										
Test2Req1	1	X										
Test3Req1	1	X										
Test1Req2	1		X									
Test2Req2	1		X									
Test3Req2	1		X									
Test1Req3	1			X								
Test2Req3	1			X								
Test1Req4	1				X							
Test2Req4	1				X							
Test1Req5	1					X						
Test1Req6	1						X					
Test1Req7	1							X				
Test1Req8	1								X			
Test2Req8	1								X			
Test1Req9	1									X		
Test1Req10	1										X	
Test1Req11	1											X

Defects Found

North key

Description: Lower case **n** does not work correctly

Reproductions Steps: Once the game has started, in any room enter lowercase **n**

Expected Behavior: It should put you in the next room located north

Observed Behavior: It displays the invalid key message – “What?”

Help key

Description: Help key **H** does not work correctly, it does not display the what each key does

Reproductions Steps: Once the game has started, in any room enter **H**

Expected Behavior: It should display all the key options along with what each key does

Observed Behavior: It displays the invalid key message – “What?”

No door movement

Description: Can still move north or south even if there is no door

Reproductions Steps: When in the first room enter **S** or when in the last room enter **N**

Expected Behavior: The game should not move the player

Observed Behavior: The game takes the player back to the first room

Who Did What

Introduction – Phillip Schuster and Ali Abu Alizz

Requirements – Phillip Schuster

Test Cases – Ali Abu Alizz

Traceability Matrix – Ali Abu Alizz

Defects Found – Phillip Schuster