# CS 1632 Software Quality Assurance

## Deliverable 1: Test Plan and Traceability Matrix

Member 1 Name: Tyler Lendon

Member 2 Name:

## 1. Introduction

I worry about thinking of test cases because I think of my test cases as being too simple for ones that I don't find errors for. Specifically, I think when writing test cases for requirements that I can't find a defect with causes me to think my test case is too simple and shouldn't be a test case.

I first went to all inputs I could think of as the start of the program. That's how I found the ctrl-z error exception. This is also how I found the defects for both "H" command and lower case "n" command. I'm surprised I didn't find any other error entering single characters or inputting strings. Though, an exhaustive search for defects using strings is impossible.

I considered the magical land as a room since it does not state what is considered a room. It even has a unique adjective to describe it since it is described as magical. Additionally, since its possible to walk into the magical land like you would a room was another reason, I decided it was a room.

I had assumed that not many edge cases exist. A few of the ones I could think of would be when no items are collected, all items are collected, and going through all of the rooms would be edges. There are probably more, but I had not thought of what they would be.

## 2. Traceability Matrix

FUN-ITERATION: WALK-NORTH-THROUGH-ALL-ROOMS

FUN-UNKNOWN-COMMAND: ENTER-CTRL-Z

FUN-INPUT-CAPS: ENTER-LOWER-CASE-N

FUN-MOVE: BEGINNING-MOVE-SOUTH

FUN-WIN: DRINK-AFTER-ALL-ITEMS

FUN-LOSE:  DRINK-BEFORE-ANY-ITEMS

FUN-INVENTORY: CHECK-INVENTORY-EMPTY, CHECK-INVENTORY-FULL

FUN-LOOK: LOOK-FIRST-ROOM

FUN-HELP: ENTER-H

FUN-UNIQ-ROOM: THIRD-ROOM-ADJECTIVE

FUN-UNIQ-ROOM-FURNISHING: ENTER-MAGIC-LAND

3. **Test Cases**

IDENTIFIER: WALK-NORTH-THROUGH-ALL-ROOMS

TEST CASE: FUN-ITERATION

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: During each iteration, when prompted for input, type “N” <enter> .

POSTCONDITIONS: You should move to each new room, and be able to input any known command in that room on that iteration.

IDENTIFIER: ENTER-CTRL-Z

TEST CASE: FUN-UNKNOWN-COMMAND

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type <ctrl>+”z” <enter>.

POSTCONDITIONS: The system shall respond with the phrase "What?" after the input of <ctrl>+”z” <enter> was entered.

IDENTIFIER: ENTER-LOWER-CASE-N

TEST CASE: FUN-INPUT-CAPS

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “n” <enter>.

POSTCONDITIONS: The game should be insensitive to the lower case and take that command the same as it would if you type “N” <enter> and move North.

IDENTIFIER: BEGINNING-MOVE-SOUTH

TEST CASE: FUN-MOVE

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “S” <enter>.

POSTCONDITIONS: The game should not allow you to move South since no door exists and display the message: “A door in that direction does not exist."

IDENTIFIER: DRINK-AFTER-ALL-ITEMS

TEST CASE: FUN-WIN

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type in this order “L <enter>”, “N <enter>”, “N <enter>”, “L <enter>”, “N <enter>”, “N <enter>”, “N <enter>”, “L <enter>”, “D <enter>”.

POSTCONDITIONS: The first look command will find creamer, two rooms north and look to find coffee, and another three rooms north and look to find sugar, and then drinking all three should win the game since the requirement to win is possible only if you have all three items.

IDENTIFIER: DRINK-BEFORE-ANY-ITEMS

TEST CASE: FUN-LOSE

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “D” <enter>.

POSTCONDITIONS: The game should end, and you should lose, since the start of the game begins with you having exactly zero of the items.

IDENTIFIER: CHECK-INVENTORY-EMPTY

TEST CASE: FUN-INVENTORY

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS:  As soon as the game begins, type “I” <enter>.

POSTCONDITIONS: Since at the start of the game, the player has no items. When the inventory is check, then no items should be shown to be collected.

IDENTIFIER: CHECK-INVENTORY-FULL

TEST CASE: FUN-INVENTORY

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type in this order “L <enter>”, “N <enter>”, “N <enter>”, “L <enter>”, “N <enter>”, “N <enter>”, “N <enter>”, “L <enter>”, “I <enter>”.

POSTCONDITIONS: The first look command will find creamer, two rooms north and look to find coffee, and another three rooms north and look to find sugar and checking your inventory should display that you have collected all of the items.

IDENTIFIER: LOOK-FIRST-ROOM

TEST CASE: FUN-LOOK

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “L” <enter>.

POSTCONDITIONS:

IDENTIFIER: ENTER-H

TEST CASE: FUN-HELP

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “H” <enter>.

POSTCONDITIONS:

IDENTIFIER: THIRD-ROOM-ADJECTIVE

TEST CASE: FUN-UNIQ-ROOM

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type in this order exactly “N” <enter>, “N” <enter>, “N” <enter>.

POSTCONDITIONS: According to the requirement for FUN-UNIQ-ROOM, each room should have a unique adjective describing it. So, the third room should have a unique adjective that describes it.

IDENTIFIER: ENTER-MAGIC-LAND

TEST CASE: FUN-UNIQ-ROOM-FURNISHING

PRECONDITIONS: Running on java version "1.8.0_221", and coffeemaker.jar is in your working directory

EXECUTION STEPS: As soon as the game begins, type “S” <enter>.

POSTCONDITIONS: According to the requirement for FUN-UNIQ-ROOM-FURNISHING that each room should have one and only one unique furnishing visible to the user upon entering the room. By requirement, upon entering the magical land room then a unique furnishing should be visible to the user.

## 4. Defects

IDENTIFIER: CTRL-Z-EXCEPTION

SUMMARY: Entering ctrl-“z” when prompted for an input produces an exception.

DESCRIPTION: This was found in ENTER-CTRL-Z test case. If ctrl-“z” is entered, then an exception will be thrown.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type ctrl+"z” <enter>.

EXPECTED BEHAVIOR: The program should display “What?” after typing ctrl-”z” <enter>, since ctrl-“z” is not a known command.

OBSERVED BEHAVIOR: The program crashes and gives a no such element exception.

IDENTIFIER: LOWER-CASE-N

SUMMARY: Entering lower case “n” when prompted for is considered unknown.

DESCRIPTION: This was found in ENTER-LOWER-CASE-N test case. A lower case “n” should be considered the same as the upper case “N” command, however it is not recognized as a command.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type exactly the lowercase “n” <enter>.

EXPECTED BEHAVIOR: Entering “n” should be case-insensitive and run the North command since “n” and “N” should be the same when the case is insensitive.

OBSERVED BEHAVIOR: The program does not recognize lowercase “n” as a command and prints “What?” after it is entered.

IDENTIFIER: SOUTH-THROUGH-NO-DOOR

SUMMARY: Entering “S” to move south from the first room, enters a magic land while no south door exists.

DESCRIPTION: This was found in BEGINNING-MOVE-SOUTH test case. The requirement states that you should only be able to move when a door is present, and no door is present that allows the player to move into the magical land.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type “S” <enter>.

EXPECTED BEHAVIOR: The game should not allow you to move South since no door exists and print the following message: “A door in that direction does not exist."

OBSERVED BEHAVIOR: The game allows you to travel South into a magical land, ignoring the requirement for a door.

IDENTIFIER: NO-HELP-PRESENTED

SUMMARY: Entering “H” does no produce any help text and is considered an unknown command.

DESCRIPTION: This was found in ENTER-H test case. The requirement states that “H” should be a help command that displays a list of commands and their use. The program does not recognize “H” as a command.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type “H” <enter>.

EXPECTED BEHAVIOR: Upon entering "H" for Help, the player should be shown a listing of possible commands and what their effects are.

OBSERVED BEHAVIOR: The command is considered unknown, and no listing of possible commands and what their effects are not shown to the player. Instead, “What?” is printed to the program after the command.

IDENTIFIER: NON-ADJECTIVE-THIRD-ROOM

SUMMARY: The word describing the third room is refinanced which is a verb.

DESCRIPTION: This was found in THIRD-ROOM-ADJECTIVE test case. A unique adjective should be given to each room that is shown to the player upon entering that room. The third room has a verb instead of an adjective to describe the room.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type in this order “N” <enter>, “N” <enter>, “N” <enter>.

EXPECTED BEHAVIOR: The room should have an adjective that describes the third room.

OBSERVED BEHAVIOR: The word being used to describe the room is “refinanced” which is a verb and not an adjective.

IDENTIFIER: MAGIC-LAND-ROOM-NO-UNIQUE-FURNISHING

SUMMARY: The magic land despite being able to be entered, has no unique furnishing shown to the player.

DESCRIPTION: This was found in ENTER-MAGIC-LAND test case. The requirements state that each room should have a unique furnishing that is displayed to the user. In the case of the magical land, which does not state it isn’t a room, does not display a unique furnishing.

REPRODUCTION STEPS: Enter java -jar coffeemaker.jar. As soon as the game begins and prompts an input, then type “S” <enter>.

EXPECTED BEHAVIOR: When the player enters a room, then a single unique furnishing should be shown to the player that is in that room.

OBSERVED BEHAVIOR: When entering the magic land, no furnishing is displayed, and the user is sent back to the beginning room of the house.