**Manual Test Plan**

Identifier: TEST-NUM-PLAYERS-NULL

Test case: Ensure that number of players must be specified

Preconditions: Open command prompt to run program, gradle run

Execution steps: Close the dialog by hitting the X in the upper right-hand corner, press ok

Postconditions: Check that an exception was thrown stating the game only supports 2-4 players

Identifier: TEST-NUM-PLAYERS-FOUR

Test case: Ensure that the game can support four players

Preconditions: Open command prompt to run program, gradle run, press 4, press ok, press normal player, press pink, press ok, press normal player, press purple, press ok, press normal player, press yellow, press ok, press normal player, press white and press classic

Execution steps: Observe the number of player displayed on the bottom status bar and their associated tokens on the start

Postconditions: Check that there are four players with four tokens on the starting square

Identifier: TEST-PLAYER-NAME-EMPTY

Test case: Ensure that if the player leaves name field blank, it defaults to player numbers

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Delete the default name for player 1 and hit enter. Delete the default name for player 2 and hit enter.

Postconditions: Check that the names defaulted back to player 1 and 2.

Identifier: TEST-PLAYER-NAME-NULL

Test case: Ensure that if the player closes the enter name field, it defaults to player numbers

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Hit the X in the upper right corner of the popup dialog. Repeat for the next dialog.

Postconditions: Check that the names defaulted to player 1 and 2 on the bottom status bar.

Identifier: TEST-PLAYER-NAME-CUSTOM

Test case: Ensure that the player can enter personalized names

Preconditions: Open command prompt to run program, press 2, type “S@r@h “, press ok, press normal player, press pink, type “RentACat!!!”, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Observe that the player names S@r@h and RentACat!!! are displayed on bottom status bar.

Identifier: TEST-MINT-SQUARE

Test case: Ensure the game board has a distinct middle space

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Check that the first pink square has a mint on it

Identifier: TEST-ICECREAM-SQUARE

Test case: Ensure the game board has a distinct middle space

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Check that the second pink square has an ice cream on it

Identifier: TEST-COTTONCANDY-SQUARE

Test case: Ensure the game board has a distinct middle space

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Check that the third pink square has a cotton candy on it

Identifier: TEST-CANDTCANE-SQUARE

Test case: Ensure the game board has a distinct middle space

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Check that the forth pink square has a candy cane on it

Identifier: TEST-LOLLIPOP-SQUARE

Test case: Ensure the game board has a distinct middle space

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Look at screen

Postconditions: Check that the fifth pink square has a lollipop on it

Identifier: TEST-TOKEN-MOVE

Test case: Ensure the player’s token moves

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press the deck

Postconditions: The first player’s token should have moved, given the first card did not have an X

Identifier: TEST-STARTING-PLAYER

Test case: Ensure that player 1 always starts

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press the deck, observe which player moves first, then restart the program and repeat again

Postconditions: Check that the same player moved first both times

Identifier: TEST-TIMER-START-NEWGAME

Test case: Ensure that a new game timer displays on the bottom bar when beginning a new game

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Observe the bottom status bar for the game time.

Postconditions: The time should display 00:00:00 and begin count by 1 every second

Identifier: TEST-TIMER-SAVE-GAME

Test case: Ensure that the game timer pauses while trying to save the game

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Hit the deck twice. Then hit the save game button. Observe the time on the status bar. Then hit the x on the dialog and observe the time again.

Postconditions: The time should not be counting while the dialog is up. Once the save game dialog is closed, the time should resume.

Identifier: TEST-LOAD-GAME

Test case: Ensure that the game timer pauses while trying to save the game

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Hit the deck twice. Observe the state of the game (where the tokens are and the time) and hit the save game button. Enter a name to save the game and hit save. Hit the deck again to move the tokens. Then select the load game button and choose the game you just saved. Observe the game state again.

Postconditions: The window should have restarted with the saved game state from before. The timer should resume from time just before the save game dialog appeared. The tokens should be in their old positions.

Identifier: TEST-PLAYER-CONSECUTIVE-TURNS

Test case: Ensure the turns alternate between players with no player going twice in a row

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Observe which player’s turn is it and then press the deck, observe which player’s turn is next and press the deck, then repeat once more

Postconditions: Check that player 1’s turn initially, then player 2’s turn, then back to player 1’s turn

Identifier: TEST-END-GAMEPLAY

Test case: Ensure that gameplay ends when one player reaches Grandma’s house

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Keep pressing the deck until you observe a player’s token reach the last square. Then try to press the deck again.

Postconditions: You should not be able to press the deck anymore since the game has ended.

Identifier: TEST-DISPLAY-WINNER-BOTTOMBAR

Test case: Ensure that winner is displayed on the bottom bar

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Keep pressing the deck until you observe a player’s token reach the last square. Then observe the text on the bottom right where the current player’s turn was usually displayed.

Postconditions: You should which player won the game.

Identifier: TEST-ENDGAME-NOTIFICATION

Test case: Ensure that when someone has won, there is a notification of the results

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Keep pressing the deck until you observe a player’s token reach the last square.

Postconditions: There should be a game over dialog displayer, which player won with their token.

Identifier: TEST-ENDGAME-EXIT

Test case: Ensure that after the game over notification is closed, the game exits

Preconditions: Open command prompt to run program, gradle run, , press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Keep pressing the deck until you observe a player’s token reach the last square. Then press “Bye!”.

Postconditions: The game should exit upon clicking “Bye!”

Identifier: TEST-SKIP-CARDS

Test case: Ensure there are special skip and middle cards in the deck

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press through the entire deck

Postconditions: Ensure a card with a star appeared and a card with an X appeared

Identifier: TEST-X-CARDS

Test case: Ensure there are special cards in the deck

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press through the entire deck

Postconditions: Ensure there are cards, one of each mint, candy cane, lollipop, ice cream and cotton candy, that appeared

Identifier: TEST-X-SPACES

Test case: Ensure there are special spaces on the board

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Observe the board

Postconditions: Ensure that there are 5 pink squares.

Identifier: TEST-UNIQUE-SQUARES

Test case: Ensure there are special cards in the deck

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press through the entire deck and watch the tokens move

Postconditions: Ensure that the tokens do not move on to the special squares without the special squares.

Identifier: TEST-SPECIAL-CARDS-RATIO

Test case: Ensure that special cards are less than half the deck

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic

Execution steps: Press through the deck, count the number of cards and count the number of special cards

Postconditions: The number of special cards should be less than half the deck

Identifier: TEST-CLASSIC-MODE

Test case: Ensure that the setup of classic mode is correct

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press classic Execution steps: Observe the board

Postconditions: Ensure that no boomerangs appear

Identifier: TEST-STRATEGIC-MODE

Test case: Ensure that the setup of strategic mode is correct

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press strategic

Execution steps: Observe the board

Postconditions: Ensure that boomerangs appear at the bottom of the screen for both player

Identifier: TEST-BOOMERANG-NUMBER

Test case: Ensure that the number of boomerangs is correct

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press strategic

Execution steps: Observe the board

Postconditions: Ensure that 3 boomerangs appear at the bottom of the screen for both player

Identifier: TEST-BOOMERANG-TURN-1

Test case: Ensure that the use boomerang button works

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press strategic

Execution steps: Press boomerang button, press player two button, press deck, observe the board

Postconditions: Ensure that 2 boomerangs appear at the bottom of the screen for bellow Player 1

Identifier: TEST-BOOMERARNG-TURN-2

Test case: Ensure that the boomerang buttons works multiple times

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press strategic

Execution steps: Press boomerang button, press player two button, press deck twice, press boomerang button, press player two button, observe the board

Postconditions: Ensure that 1 boomerangs appear at the bottom of the screen for bellow Player 1

Identifier: TEST-BOOMERARNG-MOVEMENT -1

Test case: Ensure that the boomerang buttons works once

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press strategic, press deck four times

Execution steps: Press boomerang button, press player two button, press deck twice, press boomerang button, press player two button, observe the board

Postconditions: Ensure that player two was moved back to the appropriate space

Identifier: TEST-BOOMERARNG-MOVEMENT -2

Test case: Ensure that the boomerang buttons works multiple times

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press normal player, press purple, and press strategic, press deck two times

Execution steps: Press boomerang button, press player two button, press deck twice, press boomerang button, press player two button, observe the board

Postconditions: Ensure that player two was moved back both times to the appropriate space

Identifier: TEST-AI-PLAYERS-CLASSIC

Test case: Ensure that AI players work in classic mode

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press ai player, press purple, and press classic

Execution steps: Click the deck for every time it is player one’s turn

Postconditions: Ensure that player two is drawing a new card and moving to the correct space

Identifier: TEST-AI-PLAYERS-STRATEGIC

Test case: Ensure that AI players work in classic mode

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press normal player, press pink, press ok, press ai player, press purple, and press strategic

Execution steps: Click the deck for every time it is player one’s turn

Postconditions: Ensure that player two is drawing a new card or using a boomerang every time it is player two’s turn and moving to the correct space

Identifier: TEST-AI-DISABLED-BUTTONS

Test case: Ensure that when an AI player is moving the deck cannot be clicked for that AI player or boomeranged for that player

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press AI player, press pink, press ok, press AI player, press purple, and press strategic mode

Execution steps: Try to click the deck, try to click boomerang button

Postconditions: Ensure that no boomerang dialog appears and that game play is automatic since all players are AI

Identifier: TEST-AI-PAUSE

Test case: Ensure that when an AI player movement pauses when trying to load or save a game

Preconditions: Open command prompt to run program, gradle run, press 2, press ok, press AI player, press pink, press ok, press AI player, press purple, and press classic mode

Execution steps: Click on the save game button, wait and observe the board. Close the save game dialog. Allow some game play and then click on the load game button and observe the board.

Postconditions: Ensure that after the load or save dialogs come up that game play pauses.