D1 Test Plan

By: Will Taylor & Josh Rodstein

Requirement 1: Test Cases

IDENTIFIER: TEST-INVALID-INPUT-ARG-TYPE

DESCRIPTION: Ensure that program does not accept anything other than non-negative integers as input args, and displays proper message when invalid input is entered.

PRECONDITIONS: User has necessary files to run program from CLI w/ arg and ruby > 2.5

EXECUTION STEPS: Enter a negative integer < 0 (i.e -1….. etc) or ASCII Character and press enter

POSTCONDITIONS: Program informs user of proper usage and exits

IDENTIFIER: TEST-INVALID-NUM-OF-ARGS

DESCRIPTION: Ensure that program only accepts a single valid argument

PRECONDITIONS: User has necessary files to run program from CLI w/ arg and ruby > 2.5

EXECUTION STEPS: Enter > 1 non-negative integers as input args and prss enter

POSTCONDITIONS: Program informs user of proper usage and exits

IDENTIFIER: TEST-INVALID-BLANK-SPACE-INPUT-ARG

DESCRIPTION: Ensure program does not accept null/blank space as input

PRECONDITIONS: User has necessary files to run program from CLI w/ argand ruby > 2.5

EXECUTION STEPS: Enter " " as input arg and press enter

POSTCONDITIONS: Program informs user of proper usage and exits

d
IDENTIFIER: TEST-INVALID-0-INPUT-ARG

DESCRIPTION: Ensure program does not accept null/blank space as input

PRECONDITIONS: User has necessary files to run program from CLI w/ arg and ruby > 2.5

EXECUTION STEPS: Enter "0" as input arg and press enter

POSTCONDITIONS: Program informs user of proper usage and exits

---

Requirement 2

IDENTIFIER: TEST-INT-VALID-INPUT-ARG

DESCRIPTION: Ensure the program accepts a single, < 100 non negative integer

PRECONDITIONS: User has necessary files to run program from CLI w/ arg and ruby > 2.5

EXECUTION STEPS: type `ruby connect_four.rb <int n < 99>` on cmd line and press enter

POSTCONDITIONS: Board displayed with # of columns entered as arg. All spots
on the board are blank, denoted by '.'

IDENTIFIER: TEST-LARGE-INT-VALID-INPUT-ARG

DESCRIPTION: EDGE CASE: Ensure the program accepts a single, Large > 99, non negative integer

PRECONDITIONS: User has necessary files to run program from CLI w/ argand ruby > 2.5

EXECUTION STEPS: type `ruby connect_four.rb <int n > 99>` on cmd line and press enter

POSTCONDITIONS: Board displayed with # of columns entered as arg. All spots
on the board are blank, denoted by '.'

IDENTIFIER: TEST-VERY-LARGE-INT-VALID-INPUT-ARG

DESCRIPTION: Ensure the program accepts a single, Very Large > 999, non negative integer

PRECONDITIONS: User has necessary files to run program from CLI w/ argand ruby > 2.5

EXECUTION STEPS: type `ruby connect_four.rb <int n > 999>` on cmd line and press enter

POSTCONDITIONS: Board displayed with # of columns entered as arg. All spots
on the board are blank, denoted by '.'

Requirement 3

IDENTIFIER: TEST-VALID-START

DESCRIPTION: Ensure the program starts properly on X's turn.

PRECONDITIONS: User has entered a valid input to pass through requirement 2.

EXECUTION STEPS: type `ruby connect_four.rb <int n < 99>` on cmd line and press enter

POSTCONDITIONS: Board displayed with # of columns entered as arg. The user who's turn is first is X.

IDENTIFIER: TEST-TURN-SWAP

DESCRIPTION: Ensure the program swaps correctly to O's turn after X's turn.

PRECONDITIONS: User has started the program correctly and gotten to X's turn.

EXECUTION STEPS: Enter any move for X's turn to advance to O's turn.

POSTCONDITIONS: Board displayed with whatever X had done, and the board says it is O's turn.

IDENTIFIER: TEST-TURN-MULTI-SWAP

DESCRIPTION: EDGE CASE: Ensure the program continues to swap turns properly indefinitely.

PRECONDITIONS: User has started the program correctly and gone through both user's first turns.

EXECUTION STEPS: Enter any move for the current turn to advance to the other players turn.

POSTCONDITIONS: When you were on X's turn, the next move is O's, and vice versa.

Requirement 4

IDENTIFIER: TEST-NEG-INT-INPUT

DESCRIPTION: CORNER CASE: Ensure the program doesn't accept a negative int to drop the piece.

PRECONDITIONS: User has started the program properly.

EXECUTION STEPS: Enter any negative int in the field when asked which column to drop the checker in.

POSTCONDITIONS: Game states that the number is an invalid move, and repeats the turn.

IDENTIFIER: TEST-CHAR-INPUT

DESCRIPTION: Ensure the program doesn't accept a character as a turn input.

PRECONDITIONS: User has started the program correctly and gotten to X's turn.

EXECUTION STEPS: Enter any character as your choice for the move.

POSTCONDITIONS: Game states that the character is an invalid move, and repeats the turn.

IDENTIFIER: TEST-WRONG-STRING-INPUT

DESCRIPTION: Ensure the program doesn't accept a string that isn't flip, or rot, ignoring case.

PRECONDITIONS: User has started the program correctly and gotten to X's turn.

EXECUTION STEPS: Enter a string that isn't flip or rot to see the response.

POSTCONDITIONS: Game states the string is an invalid move, and repeats the turn.

Requirement 5

IDENTIFIER: TEST-NORMAL-DROP

DESCRIPTION: Ensure that when you drop a checker in a column, it falls to the lowest available space.

PRECONDITIONS: User has started the program properly.

EXECUTION STEPS: Enter a number for a non-full column.

POSTCONDITIONS: Checker drops to the lowest available space and moves to the next player's turn.

IDENTIFIER: TEST-FULL-DROP

DESCRIPTION: Ensure that when attempting to drop into a full column, the turn repeats.

PRECONDITIONS: User has gotten through enough turns to fill a column.

EXECUTION STEPS: Enter the number for a full column during a turn.

POSTCONDITIONS: Game states that the number is an invalid move, and repeats the turn.

Requirement 6

IDENTIFIER: TEST-ROT-COMMAND-FLIPS-BOARD

DESCRIPTION: Enter Flip in lowercase & Uppercase and combo l/u

PRECONDITIONS: Board has been initialized and tokens have been placed in any position on the board in a non full column

EXECUTION STEPS: type `flip`, `FLIP`, and `FlIp` at command prompt and press enter

POSTCONDITIONS: command shall cause the board to be flipped "upside down" and all pieces falling to the "floor" of the game board, game is NOT won, turn is alternated

IDENTIFIER: TEST-FLIP-COMMAND-TOKEN-GRAVITY

DESCRIPTION: test that all pieces not in new floor after flip,
fall to new floor

PRECONDITIONS: Board has been initialized and tokens have been placed in any position in non full column

EXECUTION STEPS: type `flip` at command prompt and press enter

POSTCONDITIONS: all pieces not in full column fall to new floor.

IDENTIFIER: TEST-FLIP-COMMAND-WINS-GAME

DESCRIPTION: test that all pieces not in new floor after rotation,
fall to new floor

PRECONDITIONS: Board has been initialized and tokens have been placed in
any position not in final column

EXECUTION STEPS: type `rot` at command prompt and press enter

POSTCONDITIONS: all pieces not in previous last column fall to new floor or fall to next token in row

---

Requirement 7

IDENTIFIER: TEST-ROT-COMMAND-ROTES-BOARD

DESCRIPTION: Enter Rotate in lowercase & Uppercase and combo l/u

PRECONDITIONS: Board has been initialized and tokens have been placed in any position not in final column

EXECUTION STEPS: type `rot`, `ROT`, and `RoT` at command prompt and press enter

POSTCONDITIONS: command shall cause the board to be rotated 90 degrees to the right (i.e. clockwise) and all pieces falling to the new "floor" of the game board

IDENTIFIER: TEST-ROT-COMMAND-TOKEN-GRAVITY

DESCRIPTION: test that all pieces not in new floor after rotation,
fall to new floor

PRECONDITIONS: Board has been initialized and tokens have been placed in
any position not in final column

EXECUTION STEPS: type `rot` at command prompt and press enter

POSTCONDITIONS: all pieces not in previous last column fall to new floor or fall to next token in row

IDENTIFIER: TEST-ROT-COMMAND-WINS-GAME

DESCRIPTION: Flip of board causes four pieces to align

PRECONDITIONS: Board has been initialized and players tokens have been placed like so

```
0123456789
..........
..........
..........
..........
..........
..........
.........o
xxx......o
xoxx....oo
```

EXECUTION STEPS: Enter `flip` at command prompt and press enter

POSTCONDITIONS: Player initiating flip wins game after player x aligns four tokens in bottom row

---

Requirement 8

IDENTIFIER: TEST-VERTICAL-CONNECT-FOUR

DESCRIPTION: Test that aligning four of the same token vertically wins that game for that player

PRECONDITIONS: Game board has been generated

EXECUTION STEPS: Place 4 x tokens in column 0 and 3 x tokens in column 1

POSTCONDITIONS: Correct player wins with message "Player # won!" printed to console

IDENTIFIER: TEST-HORIZONTAL-CONNECT-FOUR

DESCRIPTION: Test that aligning four of the same token horizontally wins that game for that player

PRECONDITIONS: Game board has been generated

EXECUTION STEPS: Place 4 x tokens in column 0,1,2,3 in the final row and 3 x tokens in column 0,1,2 in seocnd ot final row | repeat for rows 0 and 1

POSTCONDITIONS: Correct player wins with message "Player # won!" printed to console

IDENTIFIER: TEST-DIAGONAL-RIGHT-CONNECT-FOUR

DESCRIPTION: Test that aligning four of the same token Rising diagonal to the right wins that game for that player

PRECONDITIONS: Game board has been generated

EXECUTION STEPS: place tokens like so:

```
    .....
    ...X0
    ..X0X
    .X000
    X00XX
```

POSTCONDITIONS: Correct player wins with message "Player # won!" printed to console

IDENTIFIER: TEST-DIAGONAL-LEFT-CONNECT-FOUR

DESCRIPTION: Test that aligning four fo the same token falling diagonal to the left wins the game for that player

PRECONDITIONS: game board ahs been generated

EXECUTION STEPS: Place tokens like so

```
    .....
    .X...
    .XX..
    .0XX0
    .000X
```

POSTCONDITIONS: Correct Player wins with message "Player # won!" printed to console.

Requirement 9

IDENTIFIER: TEST-BOARD-BIGGER-THAN-9

DESCRIPTION: Test that creating a board bigger than 9x9 prints properly.

PRECONDITIONS: Code is available and terminal is opened.

EXECUTION STEPS: 1) Run "ruby connect_four.rb xx" where xx is a number greater than 9.
POSTCONDITIONS: Board prints out a grid the size that was chosen, with only the 1's place value representing columns >9.

IDENTIFIER: TEST-DROP-AFTER-9

DESCRIPTION: Test that dropping a token into a column number over 9 works.

PRECONDITIONS: Board is created and is bigger than 9x9.

EXECUTION STEPS: 1) Pick a number greater than 9 to drop a token into.
POSTCONDITIONS: The token drops into the correct column number chosen.

IDENTIFIER: TEST-DROP-BEFORE-9

DESCRIPTION: Test that dropping a token into a column number under 9 works for a greater than 9x9 board.

PRECONDITIONS: Board is created and is bigger than 9x9.

EXECUTION STEPS: 1) Pick a number less than 9 to drop a token into.
POSTCONDITIONS: The token drops into the correct column number chosen.