**Super Connect Four**

Mustafa AL Azzawi

Chris

IDENTIFIER: CHECKING\_NON\_NEG\_INTEGER\_NORMAL\_INT

DESCRIPTION: The program shall accept a single non-negative integer as an argument and create an n x n board where n is the non-negative integer.

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: run the program with 4 as the board size. At the command line, type ruby connect\_four.rb 4

POST-CONDITIONS: The program built 4 X 4 grid with 0 to 3 above the grid point and ready to the user to make move

**STATUS: PASS**

IDENTIFIER: CHECKING\_NON\_NEG\_INTEGER\_INT\_FLOAT\_OR\_DOUBLE

DESCRIPTION: The program shall ONLY accept a single non-negative integer as an argument and create an n x n board where n is the non-negative integer.

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: run the program with 4.9999 input. At the command line, typed ruby connect\_four.rb 4.9999

POST-CONDITIONS: The program built 4 X 4 grid with 0 to 3 above the grid points and was ready to the user to make a move

**STATUS: FAIL**

IDENTIFIER: CHECKING\_NON\_NEG\_INTEGER\_INT\_CONCATENATE\_WITH\_STRING\_OR\_SYMBOL

DESCRIPTION: The program shall ONLY accept a single non-negative integer as an argument and create an n x n board where n is the non-negative integer.

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: run the program with 4.Ab# input. At the command line, typed ruby connect\_four.rb 4.Ab#

POST-CONDITIONS: The program built 4 X 4 grid with 0 to 3 above the grid points and ready for the user to make move

**STATUS: FAIL**

IDENTIFIER: CHECKING\_NON\_NEG\_INTEGER\_INT\_BIG\_NUMBER

DESCRIPTION: The program shall ONLY accept a single non-negative integer as an argument and create an n x n board where n is the non-negative integer.

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: At the command line, typed ruby connect\_four.rb 99999999999999999

POST-CONDITIONS: The program froze for a few seconds and there crushed with error. “in `block (2 levels) in generate\_board': index 268435455 too big (IndexError)”

**STATUS: FAIL**

IDENTIFIER: CHECKING\_BOARD\_WITH\_VALID\_INT

DESCRIPTION: The program shall build n rows of “.” and n columns of “.” Where n is valid integer

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: run the program with 4 as the board size. At the command line, type ruby connect\_four.rb 4

POST-CONDITIONS: The program built 4 X 4 blank ( . character) grid with 0 to 3 at the top and

**STATUS: PASS**

IDENTIFIER: CHECKING\_BOARD\_WITH\_VALID\_INT

DESCRIPTION: The program shall build n rows of “.” and n columns of “.” If the input n, where n is valid integer

PRECONDITIONS: the program has not been started yet

EXECUTION STEPS: run the program with 4 as the board size. At the command line, type ruby connect\_four.rb 4

POST-CONDITIONS: The program built 4 X 4 blank (. character) grid with 0 to 3 at the top

**STATUS: PASS**

IDENTIFIER: CHECKING\_ALTERNATE\_TURNS\_BETWEEN\_PLAYER\_X\_AND\_Y

DESCRIPTION: The program should start with player x and alternate with player y

PRECONDITIONS: 4 X 4 board size contain blank char “.” with (0-3) grid indexes above it and asking player x to play

EXECUTION STEPS: type 1 as a player x move

POST-CONDITIONS: Player O enter move

**STATUS: PASS**

IDENTIFIER: CHECKING\_ALTERNATE\_TURNS\_BETWEEN\_PLAYER\_X\_AND\_Y

DESCRIPTION: The program should start with player x and alternate with player y

PRECONDITIONS: 4 X 4 board size contain blank char “.” with (0-3) grid indexes above it and asking player x to play

EXECUTION STEPS: type 1 as a player x move

POST-CONDITIONS: Player O enter move

**STATUS: PASS**

IDENTIFIER: CHECKING\_ALTERNATE\_TURNS\_BETWEEN\_PLAYER\_X\_AND\_Y

DESCRIPTION: The program should start with player x and alternate with player y

PRECONDITIONS: 4 X 4 board size contain blank char “.” with (0-3) grid indexes above it and asking player x to play

EXECUTION STEPS: At command line player x type 1

POST-CONDITIONS: print x at the bottom of col 1 and let Player O enter move

**STATUS: PASS**

IDENTIFIER: CHECKING\_FLIP\_COMMAND\_NON\_SENSTVE\_CASE

DESCRIPTION: The program should flip board upside down when user types “flip” non-case sensitive.

PRECONDITIONS: The program has 4 empty columns except one has X at the bottom and O above it

EXECUTION STEPS: At the command line typed Flip

POST-CONDITIONS: The board flipped. O at the bottom and x above it

**STATUS: PASS**

IDENTIFIER: CHECKING\_ROT\_COMMAND\_NON\_SENSTVE\_CASE

DESCRIPTION: The program should rotate the board to the right when the user types “rot” non-case sensitive.

PRECONDITIONS: The board has one col at zero index. X at the bottom and O above it.

EXECUTION STEPS: At the command line player X typed Rot

POST-CONDITIONS: The board didn’t rotate, instead added X above O

**STATUS: FAIL**

IDENTIFIER: CHECKING\_INVALID\_MOVE\_COMMAND

DESCRIPTION: The program should output invalid move message if the user type invalid command

PRECONDITIONS: The board has one col at zero index. X at the bottom and O above it.

EXECUTION STEPS: At the command line player X typed Rotate

POST-CONDITIONS: The program printed the invalid move message and ask player x to play again

**STATUS: PASS**

IDENTIFIER: CHECKING\_INVALID\_MOV\_IF\_USER\_DROPS\_AT\_FULL\_COLM

DESCRIPTION: The program should output invalid move message if the user tries to add X or O on a full Column

PRECONDITIONS: The 4X4 grid board has one full column at index 1.

EXECUTION STEPS: At the command line player X typed 1

POST-CONDITIONS: The program printed the invalid move message and asks player x to play again

**STATUS: PASS**

IDENTIFIER: CHECKING\_IF\_SELECTING\_COL\_WILL\_DROP\_X\_OR\_O\_TO\_THE\_END\_IF\_EMPTY

DESCRIPTION: The program should out

PRECONDITIONS: The 4X4 grid board is empty.

EXECUTION STEPS: At the command line player X typed 1

POST-CONDITIONS: The letter X dropped to the end of column index 1

**STATUS: PASS**