*Owen Lindsey*

*Professor Demland, David*

*CST-201 Battleship*

*10/20/2024*

**Pseudo code for battleship**

*Pseudo code solution for PlaceDestoryer*

**METHOD** PlaceDestroyer(board, row, column, destroyer)

// Validate initial placement

**IF** row < 0 OR row >= BoardSize **OR** column < 0 **OR** column >= BoardSize **THEN**

**RETURN** false

**END IF**

**IF** destroyer.Left **AND** column = 0 **THEN**

**RETURN** false

**END IF**

// Define the basic 2x2 shape of the destroyer

**SET** offsets to: (0,0), (0,1), (1,0), (1,1)

// Adjust offsets based on orientation

**IF** destroyer.Upwards **THEN**

**FOR EACH** offset in offsets **SET** offset.Row to -offset.Row

**END FOR**

**END IF**

**IF** destroyer.Left **THEN**

**FOR EACH** offset in offsets SET offset.Column to -offset.Column

**END FOR**

**END IF**

// Validate all required cells

**FOR EACH** offset in offsets **SET** newRow to row + offset.Row **SET** newColumn to column + offset.Column

**IF** newRow < 0 **OR** newRow >= BoardSize **OR** newColumn < 0 **OR** newColumn >= BoardSize **THEN**

**RETURN** false

**END IF**

**IF** board[newRow, newColumn] is not OpenCellChar **THEN**

**RETURN** false

**END IF**

**END FOR**

// Place the destroyer on the board

**FOR EACH** offset in offsets **SET** newRow to row + offset.Row **SET** newColumn to column + offset.Column

**SET** board[newRow, newColumn] to ShipCellTaken

**END FOR**

**RETURN** true

**END METHOD**

**Test Case for PlaceDestroyer:**  
*Summary of PlaceDestroyer Test Requirements:*

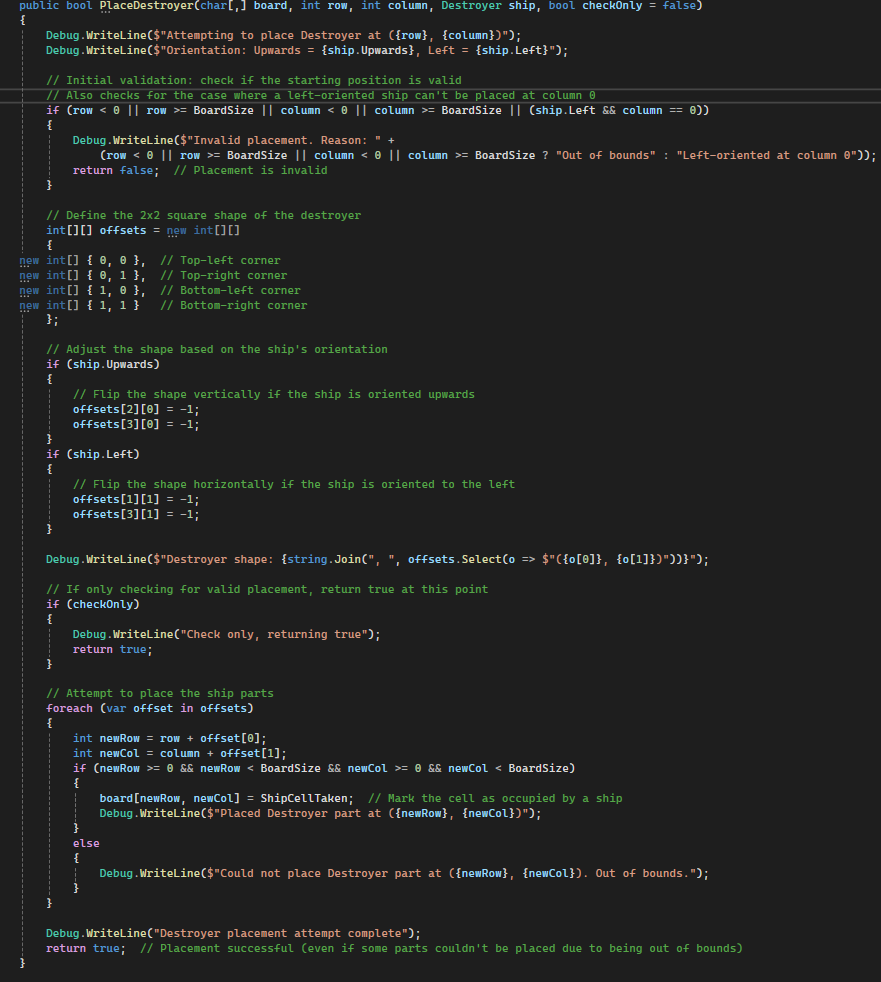
Summary of Updated PlaceDestroyer Test Requirements:

1. Standard Placement (2x2 square):
   1. Place at (3,3), occupying (3,3), (3,4), (4,3), (4,4)
2. Left-oriented Placement:
   1. Place at (2,7), occupying (2,7), (2,6), (3,7), (3,6)
3. Upward Placement:
   1. Place at (7,3), occupying (6,3), (6,4), (7,3), (7,4)
4. Upward and Left-oriented Placement:
   1. Place at (7,6), occupying (7,5), (7,6), (6,5), (6,6)
5. Invalid Placements:
   1. Out of bounds: Attempt to place at (10,10)
   2. Out of bounds: Attempt to place at (10,1) left-oriented
   3. Out of bounds: Attempt to place at (1,10) upward-oriented
   4. Out of bounds: Attempt to place at (0,0) upward and left-oriented.

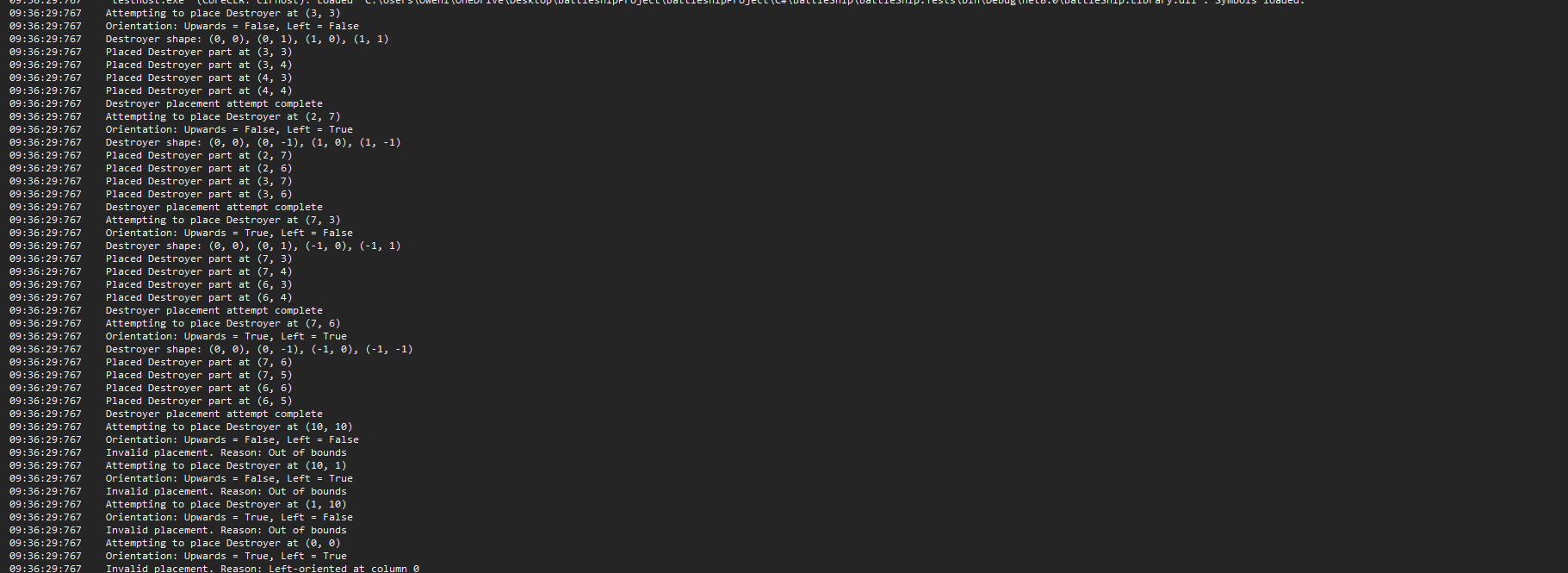
6. Why PlaceDestroyer passes all test cases:

* Checks for initial out-of-bounds conditions and the special case of left-oriented placement at column 0.
* Defines the basic 2x2 shape of the destroyer.
* Adjusts the shape based on the destroyer's orientation (Upwards and/or Left).
* Validates all cells before placement, checking for out-of-bounds conditions and existing ships (overlaps).
* Only places the ship if all checks pass, marking all four cells as occupied.
* Returns true for successful placement and false otherwise.

**Test Case for PlaceDestroyer:**  
*Screenshot of method PlaceDestroyer:*



**Test Case for PlaceDestroyer:**  
*Screenshot of debug output of PlaceDestroyer test case:*



**Test Case for PlaceDestroyer:**  
*Screenshot of success output of PlaceDestroyer test case:*

