GCU CST-201

Battleship Game

Group Project

5/29/2025

Pseudocode

START PROGRAM

INITIALIZE:

BOARD\_SIZE ← 10

SHIP\_TYPES ← [Destroyer (2x2), Submarine (3-diagonal), Cruiser (3-line)]

PLAYER\_BOARD[10][10] ← EMPTY

COMPUTER\_BOARD[10][10] ← EMPTY

PLAYER\_VIEW[10][10] ← UNKNOWN // What the player sees of the computer's board

COMPUTER\_VIEW[10][10] ← UNKNOWN // For computer logic (to avoid repeats)

FUNCTION is\_valid\_placement(board, ship\_type, start\_position, orientation):

// Check bounds and ensure no overlap

RETURN TRUE if placement is valid, otherwise FALSE

FUNCTION place\_ship(board, ship\_type, start\_position, orientation):

DETERMINE shape\_offsets BASED ON ship\_type and orientation

FOR EACH offset IN shape\_offsets:

x ← start\_position.x + offset.x

y ← start\_position.y + offset.y

IF OUT OF BOUNDS OR board[x][y] ALREADY OCCUPIED:

RETURN ERROR

FOR EACH offset IN shape\_offsets:

PLACE ship cell at (x, y)

FUNCTION random\_place\_all\_ships(board):

FOR EACH ship\_type IN SHIP\_TYPES:

REPEAT UNTIL ship is placed:

GENERATE random position and orientation

TRY place\_ship(board, ship\_type, position, orientation)

FUNCTION manual\_place\_all\_ships(board):

FOR EACH ship\_type IN SHIP\_TYPES:

REPEAT UNTIL valid:

PROMPT user for position and orientation

TRY place\_ship(board, ship\_type, position, orientation)

FUNCTION is\_valid\_shot(x, y, view):

RETURN TRUE if coordinates are in bounds and cell not already shot

FUNCTION process\_shot(x, y, target\_board, view):

IF target\_board[x][y] contains SHIP:

MARK view[x][y] as HIT

MARK target\_board[x][y] as HIT

PRINT "Hit!"

RETURN TRUE

ELSE:

MARK view[x][y] as MISS

PRINT "Miss!"

RETURN FALSE

FUNCTION check\_all\_ships\_sunk(board):

RETURN TRUE if no SHIP cells remain

// ----------- MAIN GAME LOOP -------------

CALL manual\_place\_all\_ships(PLAYER\_BOARD)

CALL random\_place\_all\_ships(COMPUTER\_BOARD)

WHILE TRUE:

// Player's turn

WHILE TRUE:

PROMPT user for shot (x, y)

IF NOT is\_valid\_shot(x, y, PLAYER\_VIEW):

PRINT "Invalid shot. Try again."

CONTINUE

RESULT ← process\_shot(x, y, COMPUTER\_BOARD, PLAYER\_VIEW)

IF check\_all\_ships\_sunk(COMPUTER\_BOARD):

PRINT "You win!"

EXIT

IF RESULT == FALSE:

BREAK // switch to computer's turn

// Computer's turn

WHILE TRUE:

GENERATE random x, y

IF NOT is\_valid\_shot(x, y, COMPUTER\_VIEW):

CONTINUE

PRINT "Computer shoots at (x,y)"

RESULT ← process\_shot(x, y, PLAYER\_BOARD, COMPUTER\_VIEW)

IF check\_all\_ships\_sunk(PLAYER\_BOARD):

PRINT "Computer wins!"

EXIT

IF RESULT == FALSE:

BREAK // switch to player's turn

END PROGRAM