BATTLESHIP- KAZOO  
TEST CASES  
  
  
**1. deploySmokeScreenOverShip**

Case 1: Deploy smoke when multiple unsunk ships are present.

Expected Outcome: Smoke is deployed near one of the unsunk ships randomly.

Case 2: Deploy smoke when only one ship remains unsunk.

Expected Outcome: Smoke is deployed around the last ship.

Case 3: Attempt to deploy smoke when all ships are sunk.

Expected Outcome: No smoke is deployed, and an appropriate message is shown.

Case 4: Call the function when no ships are placed on the grid.

Expected Outcome: No action occurs, or an appropriate error is handled.

Case 5: Deploy smoke on a ship located at the grid boundary (e.g., J10).

Expected Outcome: Smoke is only deployed in valid cells.

**2. Fire**

Case 1: Fire at an empty cell.

Expected Outcome: Cell changes to o, and "Miss!" is printed.

Case 2: Fire at a cell with a ship.

Expected Outcome: Cell changes to \*, and "Hit!" is printed.

Case 3: Fire at a cell that has already been hit.

Expected Outcome: No changes occur, and an appropriate message is displayed.

Case 4: Fire at an invalid coordinate (e.g., Z15).

Expected Outcome: Function gracefully handles invalid input.

Case 5: Fire at the grid boundary (e.g., A10).

Expected Outcome: Cell state updates if the input is valid.

**3. RadarSweep**

Case 1: Perform a radar sweep on a grid with multiple ships.

Expected Outcome: Detects ships in the sweep range and outputs their presence.

Case 2: Perform a radar sweep on an empty grid.

Expected Outcome: Outputs "No enemy ships found in the area."

Case 3: Perform a radar sweep at the grid boundary (e.g., J10).

Expected Outcome: Sweep range adjusts to stay within grid limits.

Case 4: Perform a radar sweep when radar sweeps are unavailable.

Expected Outcome: Displays "No radar sweeps remaining."

**4. SmokeScreen**

Case 1: Deploy a smoke screen in the center of the grid.

Expected Outcome: A 2x2 area around the target is marked with smoke.

Case 2: Deploy a smoke screen at the grid boundary (e.g., J10).

Expected Outcome: Smoke is applied only to valid grid cells.

Case 3: Deploy a smoke screen when no smoke is available.

Expected Outcome: Displays "No smoke screens remaining."

Case 4: Deploy smoke on an opponent's grid (invalid action).

Expected Outcome: No changes occur, and the action is rejected.

**5. Artillery**

Case 1: Use artillery on a 2x2 grid containing a ship.

Expected Outcome: Multiple hits are registered.

Case 2: Use artillery on a 2x2 grid without ships.

Expected Outcome: All cells are marked as misses.

Case 3: Use artillery on a single cell at the grid boundary.

Expected Outcome: Only valid grid cells are affected.

Case 4: Attempt to use artillery when it's locked.

Expected Outcome: Action is rejected, and an appropriate message is displayed.

Case 5: Use artillery on a grid with all ships already sunk.

Expected Outcome: No significant effect occurs.

**6. Torpedo**

Case 1: Use a torpedo on a column containing a ship.

Expected Outcome: Hits are registered across the column.

Case 2: Use a torpedo on an empty column.

Expected Outcome: Misses are registered for all cells in the column.

Case 3: Use a torpedo on the last remaining column with ships.

Expected Outcome: Hits are registered, and the ship may sink.

Case 4: Attempt to use a torpedo when it's locked.

Expected Outcome: Action is rejected.

Case 5: Use a torpedo on invalid coordinates (e.g., Z).

Expected Outcome: Function handles invalid input gracefully.

**7. CreateGrid**

Case 1: Initialize an uninitialized grid.

Expected Outcome: All cells are set to ~.

Case 2: Reinitialize a partially initialized grid.

Expected Outcome: All cells are reset to ~.

**8. compare\_strings**

Case 1: Compare identical strings (e.g., "Fire" vs. "Fire").

Expected Outcome: Returns 1.

Case 2: Compare strings with different cases (e.g., "Fire" vs. "FIRE").

Expected Outcome: Returns 0.

Case 3: Compare completely different strings (e.g., "Fire" vs. "Radar").

Expected Outcome: Returns 0.

Case 4: Compare a string with an empty string.

Expected Outcome: Returns 0.

**9. handle\_move**

Case 1: Execute a valid move (e.g., "Fire A1").

Expected Outcome: Corresponding action is performed.

Case 2: Input an invalid command (e.g., "Attack X3").

Expected Outcome: Displays "Invalid command."

Case 3: Input an empty command.

Expected Outcome: No action occurs.

**10. displayGrid**

Case 1: Display a grid with hits, misses, and unmarked cells.

Expected Outcome: Correct symbols (\*, o, ~) are shown.

Case 2: Display an empty grid.

Expected Outcome: All cells show ~.

**11. isValidPlacement**

Case 1: Test valid horizontal placement.

Expected Outcome: Returns 1.

Case 2: Test valid vertical placement.

Expected Outcome: Returns 1.

Case 3: Test placement overlapping another ship.

Expected Outcome: Returns 0.

Case 4: Test placement out of bounds.

Expected Outcome: Returns 0.

**12. UpdateProbabilityMap**

Case 1: Update the map when some opponent ships are unsunk.

Expected Outcome: Probabilities reflect possible ship positions.

Case 2: Update the map after all ships are sunk.

Expected Outcome: Probabilities are all zero.

Case 3: Update the map when the grid is partially explored.

Expected Outcome: Probabilities favor unexplored areas.

**13. BotMakeMove**

Case 1: Bot makes a valid move based on probability.

Expected Outcome: A valid target is selected.

Case 2: Bot makes a move with no valid targets left.

Expected Outcome: No move is made.

**14. PlaceShip**

Case 1: Place a ship without overlap.

Expected Outcome: Ship is placed successfully.

Case 2: Place a ship overlapping another.

Expected Outcome: Placement is rejected.

Case 3: Attempt to place a ship out of bounds.

Expected Outcome: Placement is rejected.

**15. InitializePlayer**

Case 1: Initialize a new player.

Expected Outcome: All grids and attributes are reset.

Case 2: Reinitialize a player mid-game.

Expected Outcome: All progress is lost.

**16. selectHighestProbabilityCell**

Case 1: Choose the cell with the highest probability.

Expected Outcome: Returns the correct cell.

Case 2: Choose when probabilities are tied.

Expected Outcome: Returns any of the tied cells.

**17. checkForSunkShips**

Case 1: Check after sinking a ship.

Expected Outcome: Ship count decreases.

Case 2: Check when all ships are sunk.

Expected Outcome: No changes occur.

**18. selectHighestProbabilityArea**

Case 1: Select a 2x2 area with the highest probability.

Expected Outcome: Correct area is selected.

Case 2: Select when no area stands out.

Expected Outcome: Any valid area is selected.

**19. decideRowOrColumn**

Case 1: Choose the row or column with the higher sum.

Expected Outcome: Correct choice is made.

Case 2: Handle ties in sums.

Expected Outcome: Tie-breaking logic is applied.