Test Plan for Battleship

Our testing involves unit testing with OUnit and manual testing. This ensures the correctness and robustness of our program.

We used both black-box and glass-box testing in order to verify our program from an external perspective while also covering expected edge case scenarios.

OUnit Tests

- Tests all possible grid cell types to confirm accurate string representation of ship status.
- Verifies correct conversion from string representation of coordinates to (row, column) tuples.
- Checks for proper creation of square boards of different sizes, confirming all cells are initialized to the “water” state before players position their ships.
- Confirms that the ship distribution for different board sizes is valid.
- Confirms that the states of grid cells changes correctly when an opponent tries to attack, from “water” to “miss” and “ship” to “hit” or “sunk”.
- Verifies correct placement of ships on the board, with proper lengths and no overlaps.
- Tests that hitting the last cell of a ship converts all cells of that ship to the “sunk” state rather than only “hit”.
- Tests that ship placements are within bounds, not overlapping, and correctly oriented.

Manual tests

- Verifies the correct visual representation of the board: ship placements and cell states.
- Checks that the opponent's view of the board hides the correct information.
- Verifies the correct count of sunk ships after various hit scenarios.