# Battleship Game Implementation Challenge

## About Battleship: A Classic Strategy Game

### Game Overview

Battleship is a classic two-player guessing game that simulates naval combat. Originating as a paper and pencil game in the early 20th century, it has since become a popular board game and digital experience that tests players' strategic thinking, deduction skills, and luck.

### Game Concept

In Battleship, two players each command a fleet of ships on a hidden grid. The objective is to strategically place your ships and then systematically search for and destroy your opponent's fleet by guessing their ship locations.

### Game Components

- A 10x10 grid for each player
- A fleet of 5 ships of varying sizes:
    - Carrier (5 squares)
    - Battleship (4 squares)
    - Cruiser (3 squares)
    - Submarine (3 squares)
    - Destroyer (2 squares)

### Gameplay Mechanics

#### Ship Placement

- Each player secretly arranges their fleet on their grid
- Ships can be placed horizontally or vertically
- Ships cannot overlap
- Ships must fit completely within the grid boundaries

#### Turns and Targeting

- Players take turns calling out coordinates to attack
- After each shot, the opponent reveals whether it was a "hit" or "miss"
- Hits are marked when a coordinate corresponds to an occupied square of a ship
- Misses are marked when a shot lands on an empty sea square

### Scoring and Winning

- The goal is to sink all of the opponent's ships
- A ship is considered "sunk" when all of its squares have been hit
- The first player to sink all enemy ships wins the game

### Strategic Elements

- Players must use deduction and probability to guess ship locations
- Tracking previous shots helps narrow down potential ship positions
- Varying ship sizes provide different strategic challenges

# Challenge Overview

Create a console-based Battleship game where a player competes against a computer opponent.

# Technical Requirements

- Command-line interface
- Single player vs computer opponent
- Language choice: Any programming language
- No external libraries except for standard library
- No GUI frameworks

# Features and Scoring (100 points total)

## 1. Game Setup and Board Management (30 points)

- 10x10 grid implementation (10 pts)
- Ship placement logic:
  - Random computer placement (5 pts)
  - Player placement (manual or random) (5 pts)
  - Ship overlap prevention (5 pts)
- Basic board state tracking (5 pts)

## 2. Game Logic (40 points)

- Turn management (5 pts)
- Shot validation and processing (10 pts)
- Hit/Miss detection (10 pts)
- Ship status tracking (5 pts)
- Win condition checking (5 pts)
- Computer opponent moves (5 pts)

## 3. Display and User Interface (30 points)

- Clear board visualization (15 pts)
- Game status messages (5 pts)

- Input handling:
  - Coordinate validation (5 pts)
  - Error messages (5 pts)

# Game Specifications

## Ships

| Ship Type | Size | Symbol |
|---|---|---|
| Carrier | 5 | C |
| Battleship | 4 | B |
| Cruiser | 3 | R |
| Submarine | 3 | S |
| Destroyer | 2 | D |

## Display Format

```
==== PLAYER'S BOARD ====     ==== TARGET BOARD ====
   A B C D E F G H I J          A B C D E F G H I J
1  S ~ ~ ~ ~ ~ ~ ~ ~ ~       1  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
2  S ~ ~ B B B B ~ ~ ~       2  ~ ~ O ~ ~ ~ ~ ~ ~ ~
3  S ~ ~ ~ ~ ~ ~ ~ ~ ~       3  ~ ~ ~ X ~ ~ ~ ~ ~ ~
4  ~ ~ C C C C C ~ ~ ~       4  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
5  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       5  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
6  ~ D D ~ ~ ~ ~ ~ ~ ~       6  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
7  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       7  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
8  ~ ~ ~ ~ R R R ~ ~ ~       8  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
9  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       9  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
Legend:
Your ships:    Shots:
C: Carrier     ~: Unknown
B: Battleship  O: Miss
R: Cruiser     X: Hit
S: Submarine
D: Destroyer

# Player's Turn
Enter your shot coordinates (e.g., A5): B3
Result: Hit! You struck an enemy ship.

# Computer's Turn
Computer fires at coordinate F7...
Result: Miss! Your ships are safe.
```

```
==== PLAYER'S BOARD ====      ==== TARGET BOARD ====
   A B C D E F G H I J           A B C D E F G H I J
1  S ~ ~ ~ ~ ~ ~ ~ ~ ~       1  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
2  S ~ ~ B B B B ~ ~ ~       2  ~ ~ O ~ ~ ~ ~ ~ ~ ~
3  S ~ ~ ~ ~ ~ ~ ~ ~ ~       3  ~ X ~ X ~ ~ ~ ~ ~ ~
4  ~ ~ C C C C C ~ ~ ~       4  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
5  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       5  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
6  ~ D D ~ ~ ~ ~ ~ ~ ~       6  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
7  ~ ~ ~ ~ ~ O ~ ~ ~ ~       7  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
8  ~ ~ ~ ~ R R R ~ ~ ~       8  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
9  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       9  ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~       10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
Legend:
Your ships:     Shots:
C: Carrier      ~: Unknown
B: Battleship   O: Miss
R: Cruiser      X: Hit
S: Submarine
D: Destroyer

# Player's Turn
Enter your shot coordinates (e.g., A5):
```

## Game Flow

1. **Start Game**
    - Show welcome message
    - Choose random or manual ship placement
    - Place computer ships
    - Show initial board
2. **Each Turn**
    - Show both boards
    - Show legend
    - Get player input
    - Show hit/miss result
    - Computer takes turn
    - Show computer's result
    - Check if game is won
3. **Input Format**
    - Coordinates: Letter + Number (e.g., "A5", "J10")
    - Case insensitive
    - Show error for invalid input
4. **Computer Opponent**
    - Makes random valid moves
    - Shows its moves on screen

### Hints

- Focus on getting basic game mechanics working first
- Start with random ship placement before implementing manual placement
- Test your coordinate input handling early
- Keep your display code separate from game logic
- Make sure you can detect when the game is over

---

# [OPTIONAL] Advanced Features

**Important**: Only attempt these features AFTER completing the main requirements. These additional challenges are designed to showcase extra skills and are not required for a passing submission. Your advanced features submission WILL NOT be evaluated if the main requirements are missing out.

## 1. Testing Suite (15 points)

Implement a comprehensive testing suite for your battleship game implementation. Consider various types of tests and aim for good coverage of your codebase.

## 2. AI Opponent Strategies (5 points)

Enhance the computer opponent with smart targeting strategies that go beyond random selection. Your implementation should demonstrate improved efficiency over the basic random targeting approach.

## 3. Code Quality Enhancements (5 points)

Demonstrate your understanding of software engineering best practices through improved architecture, documentation, and optimizations.

Remember: These features are entirely optional and will only be considered after evaluating the main requirements. Focus on completing the core game mechanics first.