# War Card Game Simulation

Pseudocode Documentation

Owen Lindsey

September 28, 2025

# Contents

Problem Statement . . . . . 2
Summary Statement of Logic . . . . . 2
Input/Output/Processing Diagram . . . . . 3
Algorithm Pseudocode . . . . . 4
  Function Definition . . . . . 4
  Constants and Initialization . . . . . 4
  Main Simulation Loop - Round Processing . . . . . 5
  Match Winner Determination . . . . . 6
  Final Calculations and Output . . . . . 7
Expected Program Output . . . . . 8
  Sample Round-by-Round Output (Game 1) . . . . . 8
  Final Statistics Output . . . . . 10
  Analysis of Game Simulation Results . . . . . 11
Testing and Validation . . . . . 12
  Test Data Processing . . . . . 12
  Expected vs. Actual Results Validation . . . . . 12
  Testing Requirements . . . . . 12
Resources . . . . . 13

## Problem Statement

This program simulates a modified version of the classic War card game using two computer-controlled players. In this simulation, each player begins with 26 cards represented as integer values from 1 to 13, with each value appearing twice in each player's deck (representing a standard deck's suits). The game consists of 10 complete matches, where each match involves 26 rounds of card comparisons. Players draw one card per round, and the player with the higher card value wins that round. In case of ties, no player wins the round. After all 26 rounds are completed, the player who won the most rounds wins that match. The simulation tracks total wins for each player across all matches, total ties, and calculates win percentages for final statistical analysis.

## Summary Statement of Logic

**Input:** The program initializes two arrays of 26 integers each, with values from 1-13 appearing twice in each array to represent a standard card deck. Both arrays are then randomly shuffled to create unique starting configurations for each player.

**Processing:** The simulation executes 10 complete matches, with each match consisting of 26 rounds. In each round, both players draw one card from their respective decks, and the cards are compared using a standard comparison function. The player with the higher card value wins the round, while ties result in no winner for that round. After all 26 rounds in a match are completed, the player who won the most individual rounds is declared the winner of that match. The simulation tracks cumulative wins and ties across all matches.

**Output:** Upon completion of all 10 matches, the program displays comprehensive statistics including total wins for each player, total ties encountered, and win percentages calculated as (player_wins / 10) * 100. This provides a complete analysis of the simulation results.

## Input/Output/Processing Diagram

```
INPUT:
- Two arrays of 26 integers (values 1-13, each appearing twice)
- Random shuffle of both arrays

PROCESSING:
- 10 matches (games) total
- Each match: 26 rounds of card comparisons
- Compare cards using standard comparison function
- Track round wins, match wins, and ties
- Calculate win percentages

OUTPUT:
- Total wins for Player 1
- Total wins for Player 2
- Total ties
- Win percentage for Player 1
- Win percentage for Player 2
```

## Algorithm Pseudocode

### Function Definition

```
// FUNCTION simulate_war_card_game
// This function simulates a modified War card game between two computer
// over 10 matches, with each match consisting of 26 rounds of card compa
// The simulation tracks wins, ties, and calculates final statistics.
```

### Constants and Initialization

```
// SET deck values as integers from 1 to 13
SET deck_of_cards as [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]

// CREATE player decks by duplicating the base deck
SET player_1_deck as deck_of_cards + deck_of_cards
SET player_2_deck as deck_of_cards + deck_of_cards

// INITIALIZE tracking variables
SET player_1_round_wins as 0
SET player_2_round_wins as 0
SET player_1_match_wins as 0
SET player_2_match_wins as 0
SET total_ties as 0
SET player_1_win_percentage as 0.0
SET player_2_win_percentage as 0.0

// SHUFFLE both player decks randomly
RANDOM SHUFFLE player_1_deck
RANDOM SHUFFLE player_2_deck
```

**Main Simulation Loop - Round Processing**

```
// PLAY 10 complete matches
FOR match FROM 1 TO 10

    // RESET round counters for each new match
    SET player_1_round_wins as 0
    SET player_2_round_wins as 0

    // PLAY 26 rounds within this match
    FOR round FROM 1 TO 26

        // DRAW cards from current position in deck
        SET player_1_card as player_1_deck[round - 1]
        SET player_2_card as player_2_deck[round - 1]

        // COMPARE cards and determine round winner
        IF player_1_card GREATER THAN player_2_card THEN
            SET player_1_round_wins as player_1_round_wins + 1
            PRINT "Player 1 wins round count:", player_1_round_wins

        ELSE IF player_2_card GREATER THAN player_1_card THEN
            SET player_2_round_wins as player_2_round_wins + 1
            PRINT "Player 2 wins round count:", player_2_round_wins

        ELSE
            SET total_ties as total_ties + 1
            PRINT "Tie - no one wins this round"
        ENDIF
    ENDFOR
```

**Match Winner Determination**

```
    // DETERMINE match winner after 26 rounds
    IF player_1_round_wins GREATER THAN player_2_round_wins THEN
        SET player_1_match_wins as player_1_match_wins + 1
        PRINT "Player 1 wins game!"

    ELSE IF player_2_round_wins GREATER THAN player_1_round_wins THEN
        SET player_2_match_wins as player_2_match_wins + 1
        PRINT "Player 2 wins game!"

    ELSE
        SET total_ties as total_ties + 1
        PRINT "Game is a tie!"
    ENDIF
ENDFOR
```

### Final Calculations and Output

```
// CALCULATE final win percentages
SET player_1_win_percentage as (player_1_match_wins / 10) * 100
SET player_2_win_percentage as (player_2_match_wins / 10) * 100

// DISPLAY final results
PRINT "Final Results:"
PRINT "Player 1 total wins:", player_1_match_wins
PRINT "Player 2 total wins:", player_2_match_wins
PRINT "Total ties:", total_ties
PRINT "Player 1 win percentage:", player_1_win_percentage, "%"
PRINT "Player 2 win percentage:", player_2_win_percentage, "%"
```

## Expected Program Output

### Sample Round-by-Round Output (Game 1)

| Round | Player 1 Card | Player 2 Card | Winner | P1 Score | P2 Score |
|---|---|---|---|---|---|
| 1 | 8 | 5 | P1 | 1 | 0 |
| 2 | 3 | 9 | P2 | 1 | 1 |
| 3 | 12 | 7 | P1 | 2 | 1 |
| 4 | 6 | 6 | Tie | 2 | 1 |
| 5 | 11 | 4 | P2 | 2 | 2 |
| 6 | 2 | 13 | P1 | 3 | 2 |
| 7 | 10 | 1 | P2 | 3 | 3 |
| 8 | 9 | 8 | P1 | 4 | 3 |
| 9 | 7 | 7 | Tie | 4 | 3 |
| 10 | 5 | 12 | P2 | 4 | 4 |
| 11 | 13 | 2 | P1 | 5 | 4 |
| 12 | 4 | 11 | P2 | 5 | 5 |
| 13 | 1 | 10 | P1 | 6 | 5 |
| 14 | 3 | 9 | P2 | 6 | 6 |
| 15 | 8 | 6 | P1 | 7 | 6 |
| 16 | 12 | 5 | P2 | 7 | 7 |
| 17 | 2 | 11 | P1 | 8 | 7 |
| 18 | 10 | 4 | P2 | 8 | 8 |
| 19 | 7 | 13 | P1 | 9 | 8 |
| 20 | 9 | 3 | P2 | 9 | 9 |
| 21 | 6 | 8 | P1 | 10 | 9 |

| Round | Player 1 Card | Player 2 Card | Winner | P1 Score | P2 Score |
|---|---|---|---|---|---|
| 22 | 11 | 2 | P2 | 10 | 10 |
| 23 | 5 | 12 | P1 | 11 | 10 |
| 24 | 1 | 9 | P2 | 11 | 11 |
| 25 | 13 | 7 | P1 | 12 | 11 |
| 26 | 4 | 10 | P2 | 12 | 12 |

**Final Game Result:** Tie (12-12) - No winner for this game

**Final Statistics Output**

| Game | Winner | Player 1 Wins | Player 2 Wins | Ties |
|---|---|---|---|---|
| 1 | Tie | 0 | 0 | 1 |
| 2 | Player 1 | 1 | 0 | 0 |
| 3 | Player 2 | 1 | 1 | 0 |
| 4 | Player 1 | 2 | 1 | 0 |
| 5 | Player 1 | 3 | 1 | 0 |
| 6 | Player 2 | 3 | 2 | 0 |
| 7 | Player 1 | 4 | 2 | 0 |
| 8 | Player 1 | 5 | 2 | 0 |
| 9 | Player 2 | 5 | 3 | 0 |
| 10 | Player 1 | 6 | 3 | 0 |

**Final Results Summary:** - Player 1 total wins: **6** - Player 2 total wins: **3** - Total ties: **1** - Player 1 win percentage: **60.0%** - Player 2 win percentage: **30.0%**

*Note: Games 2-10 would show similar round-by-round patterns with different outcomes.*

**Analysis of Game Simulation Results**

- **Balanced Gameplay:** The simulation shows realistic alternating wins between players, with occasional ties adding unpredictability to the game outcomes.
- **Match Determination:** Each match is decided by the player who wins the majority of the 26 rounds, creating a fair system where consistent performance over multiple rounds determines the winner.
- **Statistical Tracking:** The final output provides comprehensive statistics including total wins, ties, and win percentages, allowing for analysis of player performance across all 10 matches.
- **Random Element:** The shuffled decks ensure that each simulation run produces different results, making the game simulation realistic and unpredictable.

## Testing and Validation

### Test Data Processing

The algorithm has been designed to handle the following test scenarios:

**Test Case 1: Standard Game Flow** - **Expected:** 10 matches with 26 rounds each - **Validation:** Each match should complete exactly 26 card comparisons - **Verification:** Total rounds played = 260 (10 matches × 26 rounds)

**Test Case 2: Tie Handling** - **Expected:** Ties are properly counted and tracked - **Validation:** When `player_1_card == player_2_card`, increment tie counter - **Verification:** Total ties should be accurately reported in final statistics

**Test Case 3: Win Percentage Calculation** - **Expected:** Percentages calculated as (wins / 10) × 100 - **Validation:** Player with 6 wins should show 60.0% win rate - **Verification:** Percentages should sum to 100% (excluding ties)

**Test Case 4: Array Indexing** - **Expected:** Cards drawn sequentially from deck positions 0-25 - **Validation:** `player_1_deck[round - 1]` and `player_2_deck[round - 1]` - **Verification:** No array index out of bounds errors

### Expected vs. Actual Results Validation

The simulation output must match the following criteria: 1. **Match Structure:** Each of the 10 matches shows exactly 26 rounds 2. **Winner Determination:** Match winner has > 13 round wins out of 26 3. **Statistical Accuracy:** Final totals equal sum of individual match results 4. **Percentage Precision:** Win percentages calculated to one decimal place

### Testing Requirements

Following the professor’s recommendations: - **Initialize all variables to 0** - Ensures clean starting state - **Test every loop/decision** - Verify 10-match outer loop and 26-round inner loop - **Validate decision points** - Confirm card comparison logic and tie handling - **Match expected outcomes** - Compare actual results with predicted statistics

## Resources

Dalbey, J. (2003). *Pseudocode Standard*. Retrieved from https://users.csc.calpoly.edu/~jdalbey/SWE/pdl_std.

*This document follows the structured pseudocode conventions outlined in the Pseudocode Standard, utilizing appropriate keywords such as SET, FOR, IF-THEN-ELSE, ENDFOR, and ENDIF to describe the algorithm logic in a clear, implementation-independent manner.*