Project Summary

The project develops a formal logic model of the card game War, focusing on employing propositional logic to systematically represent and analyze the dynamics of card distribution, gameplay decisions, and winning strategies. By abstracting the game into logical propositions and constraints, the model aims to simulate various game scenarios and to identify key strategic factors that influence the outcome of games, thereby offering insights into optimal gameplay tactics.

Propositions

1. **Card(rank, suit)**: Represents a card in the deck with a specific rank and suit.
2. **Owns(player, card)**: Indicates that a particular player owns a specific card. Ownership is exclusive, ensuring that a card cannot belong to both players simultaneously.
3. **Plays(player, card, round_number)**: Represents a player playing a specific card in a given round. A player can only play cards they own.
4. **Wins(player, round_number)**: Indicates that a specified player wins a particular round based on the relative ranks of the cards played.
5. **Tie(round_number)**: Represents that a tie occurs in a specified round when both players play cards of the same rank.
6. **FinalTie(round_number)**: Indicates that no winner is found after recursive tie-breaking rounds, resulting in an unresolved tie for the specified round.
7. **HigherRank(card1, card2)**: Denotes that card1 has a higher rank than card2. This determines the winner of a round when two cards are compared.
8. **SameRank(card1, card2)**: Denotes that two cards have the same rank, leading to a tie if they are played in the same round.
9. **OverallWinner(player)**: Indicates that a specified player wins more rounds overall than the other player, based on the total number of rounds won across the game.

Constraints

- **Owns("Player A", c) → ¬Owns("Player B", c)**: A card can only be owned by one player. If Player A owns card c, then Player B doesn't and cannot own the same card.
- **Plays("Player A", c, r) ∨ Plays("Player B", c, r)**: In each round r, both players must play exactly one card, ensuring that every player makes a move.
- **Plays("Player A", c, r) → Owns("Player A", c)**: A player can only play a card they own. For example, if Player A plays card ccc in round r, they must own that card.
- **Plays("Player A", c, r) ∧ Plays("Player B", c_2, r) ∧ HigherRank(c_1, c_2) → Wins("Player A", r)**: Player A wins round r if they play a card c_1 that has a higher rank than the card c_2 played by Player B.
- **Plays("Player A", c_1, r) ∧ Plays("Player B", c_2, r) ∧ SameRank(c_1, c_2) → Tie(r)**: A round r ends in a tie if both players play cards of the same rank.
- **Wins("Player A", r) ∨ Wins("Player B", r) ∨ Tie(r)); ¬(Wins("Player A", r) ∧ Wins("Player B", r))**: Only one outcome can occur in each round: either Player A wins, Player B wins, or there is a tie. It is impossible for both players to win the same round.

- **Tie Resolution (Recursive)**: Ensures that if a tie occurs in round r, both players must play cards in the subsequent round:
  - Tie(r) → (Plays("Player A", c_1, r+1) ∧ Plays("Player B", c_2, r+1))

- **Final Tie Condition**: Limits the recursive tie-breaking process to three additional rounds. If no winner emerges, the round is declared a **FinalTie**:
  - Tie(r) ∧ Tie(r+1) ∧ Tie(r+2) ∧ Tie(r+3) → FinalTie(r)

- **Mutual Exclusivity of Outcomes**: Ensures that for any round r, only one of the three outcomes (win, loss, tie) occurs:
  - Wins("Player A", r) ∨ Wins("Player B", r) ∨ Tie(r)
    ¬(Wins("Player A", r) ∧ Wins("Player B", r))
  -
- **¬Wins("Player A", r) ∧ ¬Wins("Player B", r) → FinalTie(r)**: If neither player wins after the allowed tie-breakers, the round is declared a final tie.

Model Exploration

In developing a formal logic model for the War card game, we began by establishing the foundational constructs necessary for accurately simulating gameplay. The initial step involved defining propositions like Owns("Player A", x) and Owns("Player B", x), ensuring that each card in the deck was exclusively owned by one player. This representation allowed us to model the game's starting state, where the shuffled deck is evenly distributed between two players. The ownership setup was crucial for defining subsequent gameplay actions and enforcing game constraints.

For each round, we introduced dynamic propositions such as Plays("Player A", x, r) and Plays("Player B", x, r) to track the players' card selections. These propositions represented a central aspect of the game's mechanics: the interaction between the players through their choices of cards. To model winning conditions, we incorporated propositions like HigherRank(x, y) to establish when one card outranked another and determined the round's winner. In addition, the proposition SameRank(x, y) was used to handle ties, enabling the simulation to accurately reflect scenarios where tied ranks required further rounds to resolve.

Our model included logic to handle up to three consecutive tie-breaker rounds. These tie scenarios were managed by recursively assigning face-down cards and resolving ties through higher-ranked cards revealed later. This design added realism to the gameplay, reflecting the complexity of real-world War rules. Rigorous testing was conducted by simulating forced ties in various rounds to ensure that the tie-breaking logic triggered correctly and resolved all persistent ties. This recursive approach also handled the unlikely scenario of final unresolved ties, where neither player gained an advantage.

Handling Advanced Scenarios

To deepen the model's capabilities, we explored **stacked deck scenarios**, where a player's victory was guaranteed. For example, we simulated a game where Player A held the highest-ranked cards in every round, resulting in deterministic outcomes. This allowed us to validate the model's ability to handle extreme cases and reinforced its capacity to simulate the influence of card distribution and rank advantages on gameplay. The deterministic outcomes in these scenarios highlighted how strategic deck setups could completely alter the game's dynamics, emphasizing the importance of randomness in typical gameplay.

In addition to stacked decks, we introduced **biased shuffling** to investigate the effects of altered initial card distributions. By strategically skewing the card assignment process, we simulated conditions where one player held an inherent advantage—akin to real-world scenarios where prior successes or external factors might grant a strategic edge. Various levels of bias, from mild to significant, were tested to observe their impact on game outcomes and strategies. Our findings revealed that even slight biases in initial conditions could dramatically shift the trajectory of the game. These results underscored the critical role of fairness in the initial setup and provided insights into how perceived advantages or disadvantages might influence player behavior and strategy.

Probabilistic Outcomes and Strategic Analysis

As part of our iterative refinement process, we implemented **partial assignments** to simulate scenarios where a designated player, typically Player A, was set to win a predefined percentage of the rounds (e.g., 70%). This technique involved calculating the number of rounds Player A needed to win (e.g., 18 out of 26 rounds) and ensuring they were assigned higher-ranked cards in those rounds. By selectively applying winning conditions using logical constraints, we tested the model's flexibility under controlled biases. This enabled us to study the impact of strategic advantages on overall outcomes, shedding light on how subtle changes in game setup can influence player strategies.

Recursive Tie Resolution and Quantifiers

A key feature of the model was the use of **quantifiers** to resolve ties dynamically. For instance, in tie scenarios, additional face-up and face-down cards were recursively assigned to simulate realistic tie-breaking rules. Quantifiers ensured that the constraints dynamically evolved with the game's state, accurately modeling the rules governing tie resolution. This recursive logic extended the model's capability to handle intricate game states, such as successive ties requiring multiple rounds for resolution.

The tie-breaking mechanism was particularly useful in stress-testing the model's robustness. Simulating tie-heavy scenarios demonstrated that the model could efficiently handle edge cases and adhere to the rules governing fair play and game progression. The use of quantifiers provided a structured method for evaluating increasingly complex gameplay interactions, further validating the model's adaptability.

Final Refinements and Advanced Game Theory

The transition from our draft model to the final submission involved iterative enhancements to account for probabilistic outcomes, strategic manipulations, and fairness in gameplay. By introducing deterministic scenarios, biased shuffling, and partial assignments, we expanded the scope of the model beyond simple gameplay replication to a tool for exploring advanced game-theoretical concepts. These refinements allowed us to investigate how initial conditions, randomness, and strategy interplay to shape the game's trajectory.

The final model is a comprehensive representation of the War card game, capable of simulating not only standard gameplay but also advanced scenarios reflecting deterministic, biased, or strategic elements. This robust framework serves as both a practical simulation tool and a platform for deeper analysis of probabilistic and strategic variations in competitive settings.

Jape Proof Ideas

1. *If for every card x that Player A owns, Player B cannot own the same card, then it logically follows that there is no card x that both Player A and Player B own simultaneously.*

A→¬B ⊢ ¬(A∧B)

1:	A→¬B	premise
2:	A∧B	assumption
3:	B	∧ elim 2
4:	A	∧ elim 2
5:	¬B	→ elim 1,4
6:	⊥	¬ elim 3,5
7:	¬(A∧B)	¬ intro 2-6

Premise: (OwnsA(x) → ¬OwnsB(x))

Conclusion: ¬ (OwnsA(x) ∧ OwnsB(x))

2. Premise:

Premise 1: PlaysA(c,r) → OwnsA(c)– If Player A plays card c in round r, then Player A owns that card.

Premise 2: WinsA(r) → PlaysA(c1,r) ∧ PlaysB(c2,r) ∧ HigherRank(c1, c2 – Player A wins round r if Player A plays c1, Player B plays c2, and c1 is higher-ranked than c2.

Premise 3: HigherRank(c,c')→ ¬ SameRank(c,c') – If a card c has a higher rank than another card c', then they cannot have the same rank.

P→A, Q→(P∧B∧H), H→¬S ⊢ Q→(A∧¬S)

1:	P→A, Q→(P∧B∧H), H→¬S	premises
2:	Q	assumption
3:	P∧B∧H	→ elim 1.2,2
4:	H	∧ elim 3
5:	¬S	→ elim 1.3,4
6:	P∧B	∧ elim 3
7:	P	∧ elim 6
8:	A	→ elim 1.1,7
9:	A∧¬S	∧ intro 8,5
10:	Q→(A∧¬S)	→ intro 2-9

Conclusion: If Player A wins round r, then Player A owns a card that is not of the same rank as Player B's card

3. Premises:

Premise 1: PlaysA(c,r)→OwnsA(c): If Player A plays card c in round r, then Player A must own that card.

Premise 2: OwnsA(c)→¬OwnsB(c): If Player A owns card c, then Player B cannot own that card.

Pa→A, A→¬B ⊢ Pa→¬B

1:	Pa→A, A→¬B	premises
2:	Pa	assumption
3:	A	→ elim 1.1,2
4:	¬B	→ elim 1.2,3
5:	Pa→¬B	→ intro 2-4

Conclusion: if Player A plays card c in round r, then Player B does not own that card.

First-Order Extension

To

To enhance the War game model with a first-order logic extension, we integrate parameterized predicates and quantifiers to provide a more dynamic and scalable system. This transition from propositional to first-order logic allows for a generalized representation of the game's rules and scenarios, avoiding the limitations of enumerating each case individually.

In our original model, specific propositions such as Owns(player, card) and Plays(player, card, round_number)were defined discretely for each instance. Transitioning to a first-order logic framework, we employ predicates that parameterize these relations, encompassing all possible instances within a unified structure. For instance:

- The ownership relation is expressed as $\forall c\ (\text{Owns}(\text{"Player A"}, c) \rightarrow \neg\text{Owns}(\text{"Player B"}, c))$. This universally quantified statement ensures that if Player A owns any card c, then Player B cannot own that same card, thereby enforcing exclusive ownership without repeating constraints for each card.

- The action of playing a card is captured by $\forall r\ \exists! c\ (\text{Plays}(\text{"Player A"}, c, r)) \land \exists! c\ (\text{Plays}(\text{"Player B"}, c, r))$, asserting that in every round r, exactly one card c is played by each player. The use of the uniqueness quantifier $\exists!$ clarifies that each player can play only one card per round, simplifying the modeling of turns and ensuring the game progresses orderly.

Handling more complex scenarios, such as multiple tie-breaking rounds, becomes streamlined with first-order logic. We define:

- Tie Conditions: $\forall r\ (\text{Tie}(r) \rightarrow (\exists c1, c2\ (\text{Plays}(\text{"Player A"}, c1, r) \land \text{Plays}(\text{"Player B"}, c2, r) \land \text{SameRank}(c1, c2))))$ encapsulates the condition for a tie in any round r, where both players play cards of the same rank. This definition naturally extends to handle multiple consecutive tie-breaking rounds by recursively applying the tie condition to subsequent rounds until a winner is determined or a final tie is declared.

Implementing recursive tie resolution involves defining a procedure within our logical framework that checks subsequent rounds for a higher-ranked card when a tie occurs. This procedure would iteratively apply the tie condition and check for winning conditions until the tie is resolved or confirmed as unresolved after a set number of rounds, enhancing the model's capability to deal with extended game dynamics.

By employing first-order logic, the War game model not only achieves a reduction in complexity but also gains the ability to flexibly adapt to varied game scenarios and player strategies.