Rafid Sarowar

CIS-5

25SUM-CIS-5-46224

Project 1

What Is Ludo?

Ludo is a simple race-and-capture board game for 2 to 4 players. Each person rolls a single die to move their tokens from “yard” onto a cross-shaped track, aiming to bring all pieces safely home. Landing on an opponent’s token sends it back to its yard, combining luck and light strategy in every turn.

Origins and Evolution

Ludo traces its roots to the Indian game Pachisi (6th century AD), where cowrie shells and cloth boards defined royal pastimes. In 1896, British colonists formalized it as “Ludo” (Latin for “I play”), replacing shells with a cubical die and standardizing rules. From Britain, it spread worldwide, adapting to local materials and player styles.

Ludo in Bangladesh

In Bangladesh, লুডু boards—often plastic or chalk-drawn—are staples of monsoon nights and festive gatherings. Families and neighbors crowd verandahs or street corners, sipping tea as dice clatter and laughter fills the air. Impromptu matches bridge generations, blending friendly rivalry with casual storytelling.

A Core Family Memory

Growing up in Bangladesh, Ludo was all my brother, cousins, and I ever played. I must have whiled away thousands of monsoon-drenched evenings clustered around that battered plastic board in our courtyard. Grandparents would hover, gently guiding our tokens; cousins would erupt in laughter after each cunning “cut”; and my grandmother’s soft scolds came whenever I forgot to say “thank you” after rolling a six. Those countless games—just dice, tokens, and shared stories—wove a bond between us that still brings a smile whenever I think back.

How does the game work on V.6(Final Version)?

Welcome & Rules
At start, the game prints a friendly banner and all the rules—roll a six-sided die, choose one of your two tokens, watch for the triple-six penalty, cut opponents by landing on them, and never overshoot the finish (safe-stop clamps you at 50) Ludo_V6.

Choosing Players
You're asked how many are playing (2–4); the code validates this with a `do...while` loop before proceeding to Ludo_V6.

Seeding & Logging
The RNG is seeded via `time(nullptr)`, and an `ofstream` named `logF` opens `game_log_v6.txt`. Every roll, move, cut, and safe-stop is appended here, Ludo_V6.

Main Loop

- **Roll Phase**: Player presses Enter, the code does `rand() % 6 + 1`, prints "Rolled: X" and logs it.
- **Pip Description**: A `switch` echoes "One pip." … "Six pips." Ludo_V6.
- **Triple-Six Skip**: If you roll a 6 three times in a row, you immediately lose your turn (`cons6 >= 3` triggers a `continue`), reset `cons6`, advance to the next player, and log the penalty Ludo_V6.
- **Token Selection**: A `do...while` ensures you pick token 1 or 2. Pointers (`tokPtr`) pick the correct `pX_Y` variable based on player and choice Ludo_V6.
- **Safe-Stop**: The code computes `newPos = *tokPtr + dRoll`; if it exceeds `FN_POS` (50), it sets `*tokPtr = FN_POS` instead, prints/saves "safe-stopped at 50," else moves normally Ludo_V6.
- **Cut Rule**: After moving, it compares `movedPos` against every other token (explicit `if` chains for players 1–4). On a match, that opponent's token resets to 0, a message prints, and it logs "Cut P<…>_…." Ludo_V6.
- **Progress Report**: A `for(int p=1; p<=plCnt; ++p)` loop prints each token's completion percentage (`tokenPos/50*100`) formatted to one decimal. Internally, it calculates a dummy `pow(progress, PBS)` to satisfy the math library requirement

Ludo_V6.

- **Victory Check**: If both tokens of any player are≥50, it sets `gOver = true`, announces “Player N WINS!”, and breaks out.
- **Next Player**: If no winner, `curPl = (curPl % plCnt) + 1` advances turn order.

Exit
Once someone wins (or 1,000 turns elapse), the loop ends, the log file closes, and the program returns success.

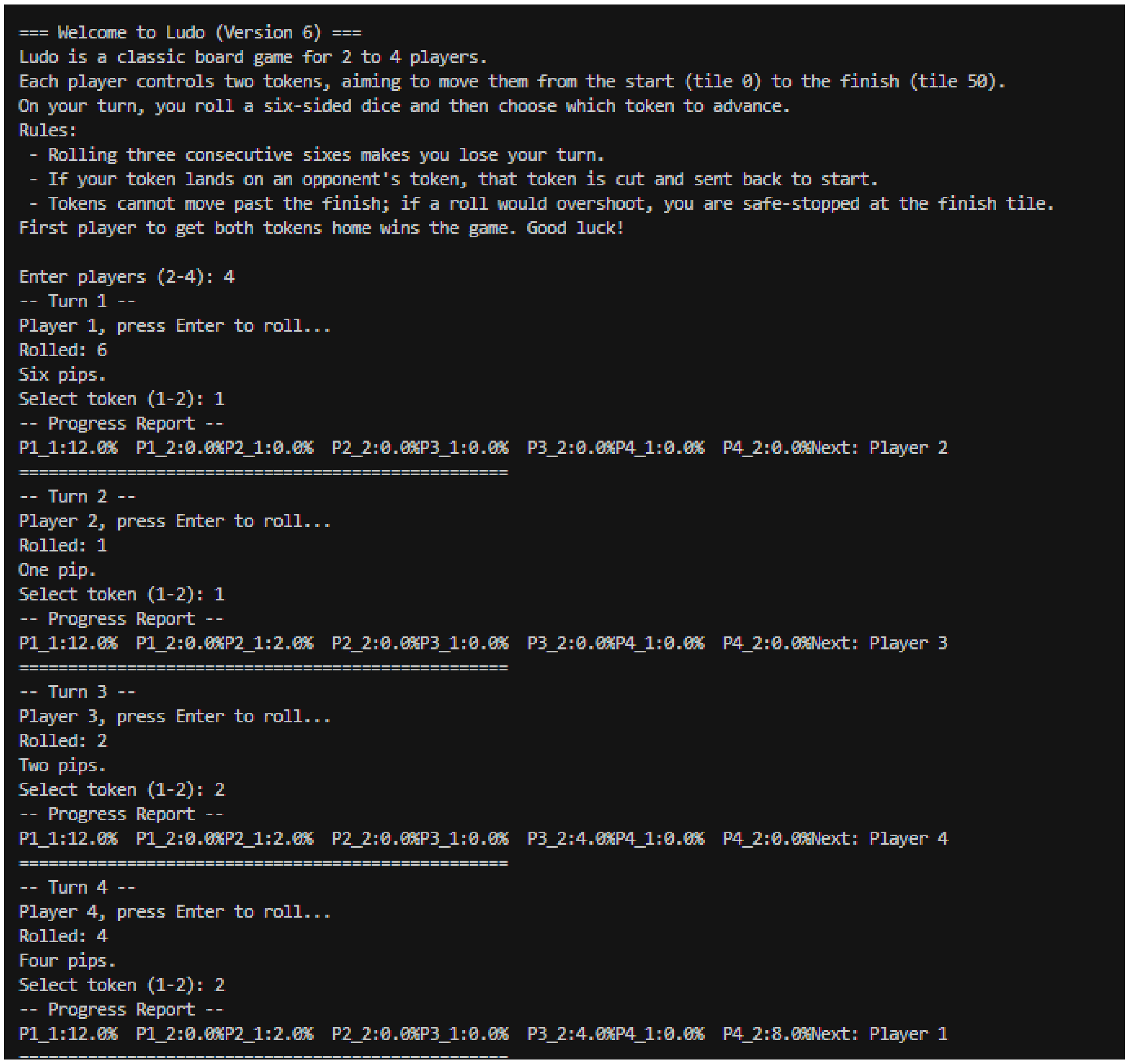
Game Overview (Version 6)

Version 6 delivers a full-featured, console-based Ludo for 2–4 players, each with two tokens racing from Start (0) to Finish (50). Key mechanics include:

- Three-six penalty: Rolling three consecutive sixes forfeits your turn.
- Safe-stop: Tokens never overshoot the finish—they clamp at 50.
- Cut rule: Landing on an opponent’s token sends it back to Start.
- File logging: Every roll, move, cut, and safe-stop is recorded in `game_log_v6.txt`.
- Formatted progress report: After each turn, a `for` loop prints each token’s completion percentage.
- Static local constants, no global variables; all numeric types are `float` or `int`.

```
/*
 * File:   Ludo_V6.cpp
 * Purpose: Complete Ludo for 2–4 players with all final rules and
logging.
 */
```

Ludo_V6



```
=== Welcome to Ludo (Version 6) ===
Ludo is a classic board game for 2 to 4 players.
Each player controls two tokens, aiming to move them from the start (tile 0) to the finish (tile 50).
On your turn, you roll a six-sided dice and then choose which token to advance.
Rules:
 - Rolling three consecutive sixes makes you lose your turn.
 - If your token lands on an opponent's token, that token is cut and sent back to start.
 - Tokens cannot move past the finish; if a roll would overshoot, you are safe-stopped at the finish tile.
First player to get both tokens home wins the game. Good luck!

Enter players (2-4): 4
-- Turn 1 --
Player 1, press Enter to roll...
Rolled: 6
Six pips.
Select token (1-2): 1
-- Progress Report --
P1_1:12.0%  P1_2:0.0%P2_1:0.0%  P2_2:0.0%P3_1:0.0%  P3_2:0.0%P4_1:0.0%  P4_2:0.0%Next: Player 2
==================================================
-- Turn 2 --
Player 2, press Enter to roll...
Rolled: 1
One pip.
Select token (1-2): 1
-- Progress Report --
P1_1:12.0%  P1_2:0.0%P2_1:2.0%  P2_2:0.0%P3_1:0.0%  P3_2:0.0%P4_1:0.0%  P4_2:0.0%Next: Player 3
==================================================
-- Turn 3 --
Player 3, press Enter to roll...
Rolled: 2
Two pips.
Select token (1-2): 2
-- Progress Report --
P1_1:12.0%  P1_2:0.0%P2_1:2.0%  P2_2:0.0%P3_1:0.0%  P3_2:4.0%P4_1:0.0%  P4_2:0.0%Next: Player 4
==================================================
-- Turn 4 --
Player 4, press Enter to roll...
Rolled: 4
Four pips.
Select token (1-2): 2
-- Progress Report --
P1_1:12.0%  P1_2:0.0%P2_1:2.0%  P2_2:0.0%P3_1:0.0%  P3_2:4.0%P4_1:0.0%  P4_2:8.0%Next: Player 1
==================================================
```

Iteration 1: Version 1 – Basic Two-Player Roll & Move

- Players: 2
- Tokens: Each has 2, explicit `int` variables (no arrays).
- Mechanics:
    1. Roll a 6-sided die.
    2. Choose which of your two tokens to advance.
    3. First to get both tokens ≥50 wins.

```
// Named constants and token positions
static const int ST = 0, FN = 50, DS = 6, MAXT = 500;
int p1_1 = ST, p1_2 = ST, p2_1 = ST, p2_2 = ST;
```

Ludo_V1

V1 Input/Output Example

- Turn 1: Player 1 rolls 4, moves token 2 to tile 4.
- Turn 2: Player 2 rolls 6, moves token 1 to tile 6.

Iteration 2: Version 2 – Adding the Cut Rule

- New feature: If your token lands exactly on an opponent’s token, that opponent’s token resets to Start.
- Players: Still 2
- Implementation: After moving, compare positions and apply the cut.

```
// Cutting rule: if activeToken == opponentToken, reset opponentToken
if (activeToken == opponentToken) {
    opponentToken = ST_POS;
    cout << "Cut! Opponent's token returns to start.\n";
}
```

Ludo_V2

V2 Scenario

- Player 1 moves to tile 12; Player 2’s token was there → Player 2's token reset.

Iteration 3: Version 3 – Support for 2–4 Players

- Player count: Prompted (2–4).
- Token variables: Extended to `p3_1`, `p3_2`, `p4_1`, `p4_2`.
- Turn order: `currentPlayer = (currentPlayer % plCnt) + 1`.

```
// Prompt for number of players (2–4)
do {
    cout << "Enter number of players (2–4): ";
    cin >> plCnt;
} while (plCnt < 2 || plCnt > 4);
```

Ludo_V3

V3 Gameplay

- In a 3-player game, turns cycle P1→P2→P3→P1…

Iteration 4: Version 4 – Triple-Six “Lose Turn” Rule

- New feature: Rolling three consecutive sixes causes you to lose your turn immediately, with a notification.
- Implementation: A `cons6` counter resets on non-six rolls.

```
// Triple-six penalty
if (dRoll == 6) {
    cons6++;
    if (cons6 == 3) {
        cout << "Three sixes! You lose your turn.";
        cons6 = 0;
        curPl = (curPl % plCnt) + 1;
        continue;  // skip movement
    }
} else {
    cons6 = 0;
}
```

Ludo_V4

- P2 rolls 6-6-6 → instantly skips to P3.

Iteration 5: Version 5 – Safe-Stop, File Logging & Progress Display

- Safe-stop: Rolls that would move a token beyond tile 50 instead set it exactly to 50.
- File logging: All events written to `game_log_v5c.txt`.
- Progress display: Manual percentage report for the two-player case.

```
// Safe-stop logic
int newPos = *tokenPtr + roll;
if (newPos > FINISH) {
    *tokenPtr = FINISH;
    cout << "Safe-stop at finish!\n";
    logFile << "safe at " << FINISH << "\n";
} else {
    *tokenPtr = newPos;
    logFile << "->" << newPos << "\n";
}
```

Ludo_V5

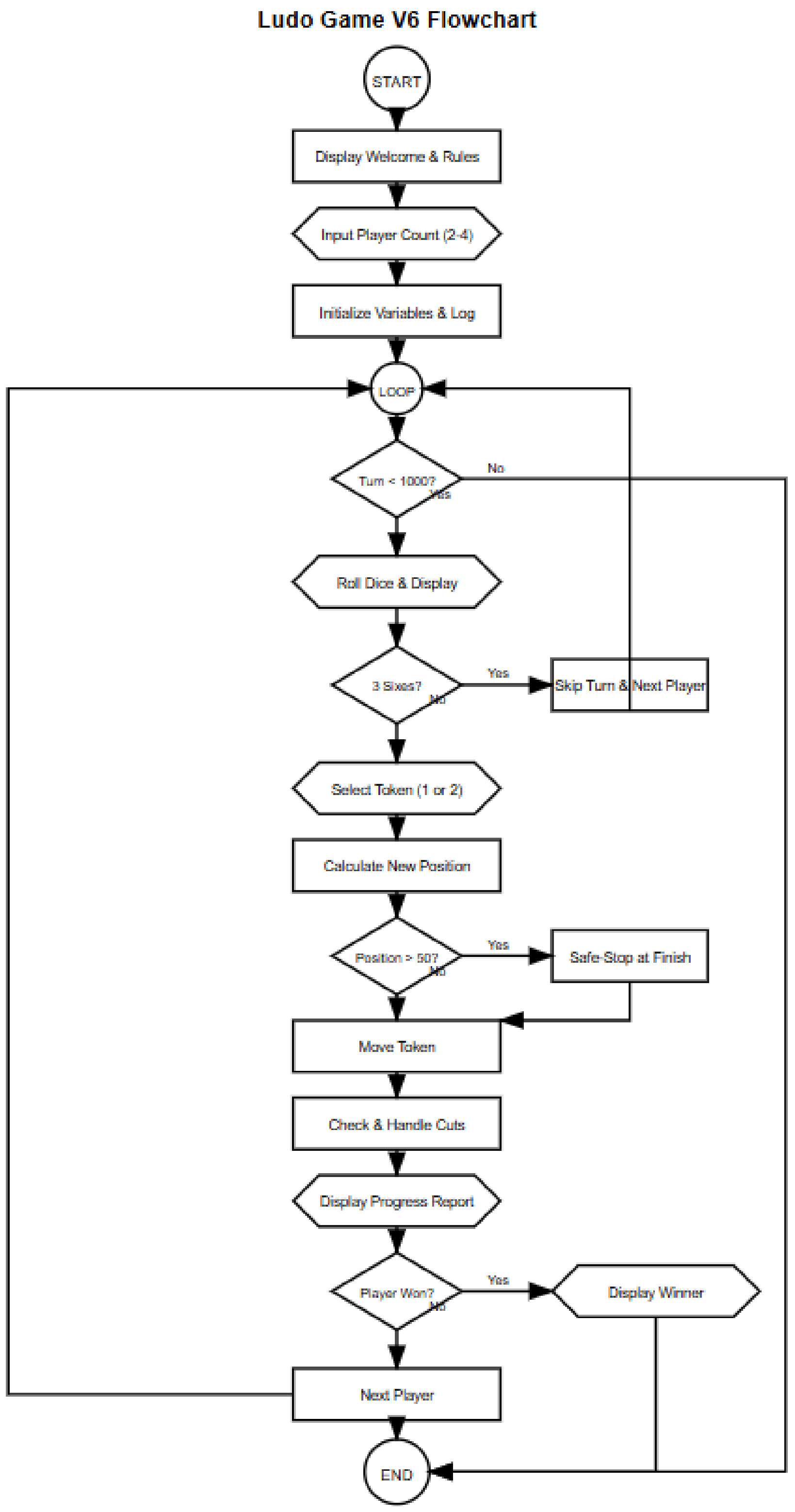
```
Progress P1: 60%, 45%  P2: 30%, 10%
```

Conclusion

Across these six versions, the Ludo project illustrates incremental application of core programming constructs:

1. Version 1 introduced primitives, loops, and input/output.
2. Version 2 added branching and relational checks for the cut rule.
3. Version 3 scaled to support variable player counts.
4. Version 4 demonstrated loop control with `continue` for special penalties.
5. Version 5 integrated file I/O, math functions, and formatted output.
6. Version 6 finalized with full feature set—including safe-stop, logging, progress loops, static constants, and clean code style.

Flowchart


Ludo Game V6 Flowchart
START
Display Welcome & Rules
Input Player Count (2-4)
Initialize Variables & Log
LOOP
Turn < 1000?
No
Yes
Roll Dice & Display
3 Sixes?
Yes
No
Skip Turn & Next Player
Select Token (1 or 2)
Calculate New Position
Position > 50?
Yes
No
Safe Stop at Finish
Move Token
Check & Handle Cuts
Display Progress Report
Player Won?
Yes
No
Display Winner
Next Player
END