**Console-based Chess Game Progress Report**

Ezequiel N. Buck Martínez

Texas Tech University

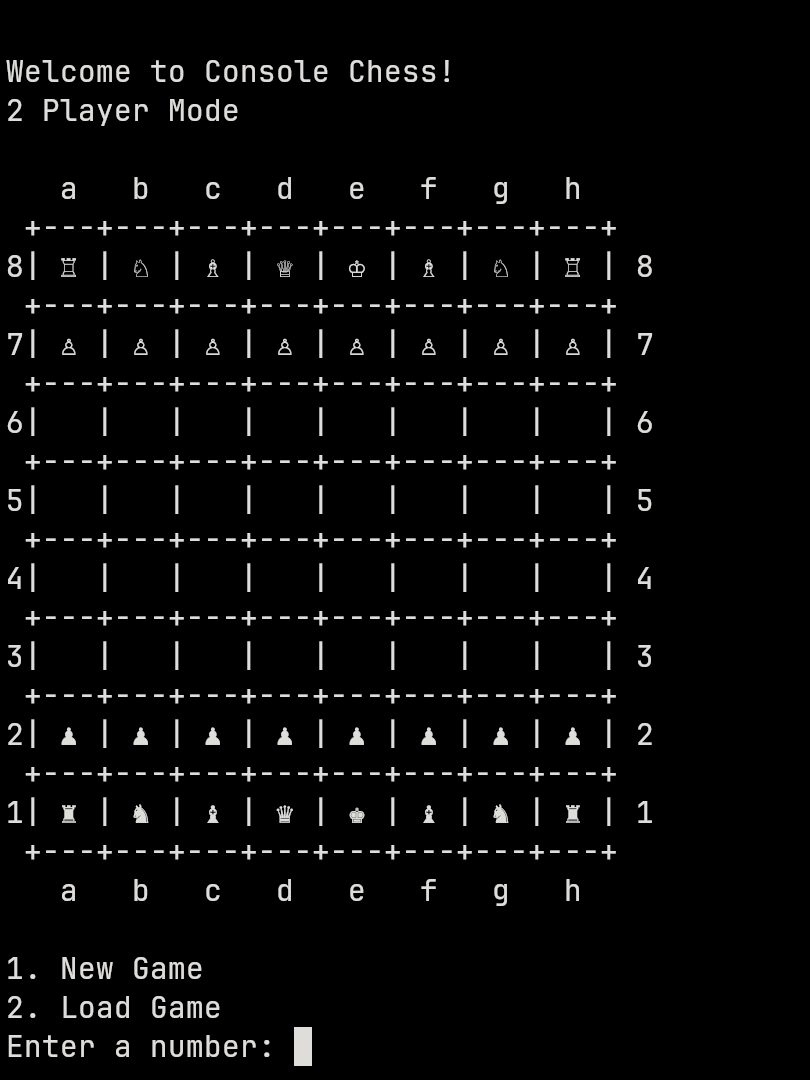
CS 1412: Programming Principles

Dr. Juan Carlos Rojas

November 24, 2024

**Refined Project Description**

The chess game project has evolved substantially from its initial proposal. The game begins by presenting players with a clean, Unicode-based chessboard display showing all pieces in their starting positions. The use of Unicode characters for chess pieces, rather than the initially proposed text notation, has significantly improved the visual clarity of the game. Each piece is distinctly visible on the board, making it easier for players to assess the game state at a glance. However, to ensure a complete visual experience, including the display of icons, your terminal must have a Nerd Font installed. Nerd Fonts provide the necessary glyphs to render game icons correctly. Without them, certain symbols may not appear as intended, potentially affecting the user interface.



The movement system has been implemented as a straightforward two-step process. Players specify their moves by entering the coordinates of the piece they wish to move (e.g., "e2") followed by their desired destination (e.g., "e4"). This system, while slightly different from the notation system proposed initially, has proven more user-friendly and intuitive, especially for players who might not be familiar with chess notation.

A significant enhancement to the original design is the implementation of comprehensive move validation. The game now includes specific movement rules for each piece type, ensuring that pawns can only move forward and capture diagonally, knights maintain their L-shaped movement pattern, bishops move diagonally, rooks move in straight lines, queens combine both rook and bishop movements, and kings move one square in any direction. The validation system also prevents pieces from moving through other pieces and ensures players can't capture their own pieces.

The game state management has been implemented using a combination of efficient data structures. The board state is maintained in a two-dimensional array, while captured pieces and move history are tracked using linked lists. This allows for easy access to the game state while maintaining a complete record of the game's progression. The system also alternates turns between players, displaying the board from each player's perspective to maintain a natural playing experience.

The program successfully handles piece movement, captures, and turn management. The board display automatically rotates based on the current player's turn, showing white's perspective during white's turn and black's perspective during black's turn. This feature, while not part of the original proposal, has proven to be a valuable addition for player orientation.

The game also maintains a history of moves and displays captured pieces for both players, adding a strategic element as players can track the material advantage of each side. The move history system uses a linked list structure, allowing for potential future features such as move undoing or game replay.

The save/load game functionality outlined in the original proposal has been structured but not fully implemented. The groundwork has been laid through the game state management system, making it straightforward to add this feature in the next development phase.



**Future Development Plans**

While the current implementation provides a solid foundation, several enhancements are planned for the final version. The most significant planned addition is the implementation of finishing the game when a king is captured.

The save/load game functionality will be completed, allowing players to save their progress and resume games later. This will involve writing the game state to a binary file and implementing the corresponding load functionality. Additionally, the initial print will include instructions for the game, detailing the menu options and the use of special keys. These special keys will allow players to print the move history and save the game state to a binary file during gameplay.

Another critical enhancement will be improving input validation for the piece and the desired position a player wants to move their piece. Currently, unexpected inputs can lead to segmentation faults. The improved validation will ensure that inputs are checked and handled properly to prevent such issues, enhancing the program's robustness and user experience.

**Structure Chart**

The structure chart attached provides a visual representation of the current function hierarchy, demonstrating how the various components of the game interact. This structure has proven effective in maintaining code organization while allowing for future expansions of functionality.

Diagrama

Descripción generada automáticamente