Saenz Exceiver, Soto Brandon, Cheng Bungein

Lab 3 Chess Game Report:

Objective:

The main goal of this Java project is to improve the development of a Chess game, implementing classes and inheritance for better performance and practice of the concepts. Allowing users to choose and place various chess pieces on a chessboard. The implementation demonstrates key features to ensure a functional demonstration of the progress of the program.

Key Features:

Piece Selection: The program utilizes Enums (ChessPiece, Color, Column, and Row) to enhance code readability and maintainability. Users are prompted to select chess pieces from a predefined set, including pawn, rook, knight, bishop, queen, and king. The program has an array (Piece) to store instances of the selected ChessPieces classes. In this way, the multiple options of the program are more coordinated with the display of data.

Error Handling: To prevent logical errors, the code incorporates Boolean flags (visitedPawn, visitedRook, etc.) to track whether a particular type of chess piece has been selected already. Users receive informative error messages when attempting to choose the same piece more than once. The program validates initial positions to ensure they fall within the valid range of the chessboard or when they try to introduce an invalid character.

Piece Validation: Users are prompted to input a target position for each piece after selecting the chess pieces. The program validates the target positions to check if they are within the chessboard and conform to the rules of movement for each specific piece. It then reports whether the selected piece can move to the target position or not, providing valuable feedback to the user. If the user input is not correct, the program will display a message error indicating to try again.

Code Structure: Enums enhance code clarity by replacing strings with symbolic names for chess piece types, columns, and colors. The switch statement is employed for handling different types of chess pieces, ensuring clean and readable code. Boolean flags are used to efficiently track visited chess pieces, preventing duplicates.

Classes implemented:

Chess piece: This class is responsible for the inheritance of the attributes of the pieces such as Name, Color, Column, and Row. This class includes a print statement of the description of the piece and a Boolean method to let know the user if the piece selected is in the same position as the other pieces.

Bishop, Pawn, Rook, and Knight: This piece class is inherited from the Chess Piece class, each piece overrides a method that specifies its movements including an extra method that overrides if the desired new spot is a valid move or not.

Queen: Queen class inheritance from Bishop class and includes the same method of verifying move as Rook. This class includes all attributes from the bishop through the Chess Piece class.

King: King class inheritance from Queen, this class has very similar movement as Queen, the only detail is this class only can move one unit.

Part name	Purpose	Pseudocode
Enumerated Types	Define types of chess pieces and colors	Define chess piece type and chess color enumerations
Chess piece class	Represent a chess piece with properties and movement verification methods.	Define chess piece class with piece name, color, column, and row fields. Implement methods for getting and setting values. Implement verify target method based on chess piece type.
Chess board class	Represent the chessboard and check if a coordinate is within boundaries.	Define chessboard class with max row and min row. Implement a method to check bounds
User interaction loop	Manage the main loop and user input.	Use a while loop for continuous user interaction. Prompt user for chess piece selection and input. Check if the user wants to continue or terminate the game.
Prompt the user for input	Prompt the user for the chess piece and position.	Print a message to select a chess piece type, color, column, and row.
Check chess boundary	Verify if the position is within bounds.	Check if the column and row are within a valid range. Display an appropriate message for an invalid position.
Prompt user for target	Prompt the user for the target position.	Print a message to enter a target position. Get user input for the target column and target row.
Check target validity	Check if the target position is valid.	Verify if the target position is within the chessboard. Check if the target position is different from the current position. Display appropriate messages for invalid targets.
Check chess piece rules	Verify if the target position is valid based on chess piece rules and the original position.	Use chess piece rules to validate the move. Display the result of the movement.
Ask the user for another target position	Ask the user if they want to verify another target position	Print a message asking if the user wants to move the piece

	using the same original position, move a different piece, or end the program.	again from the same original position, move a different piece, or end the program.

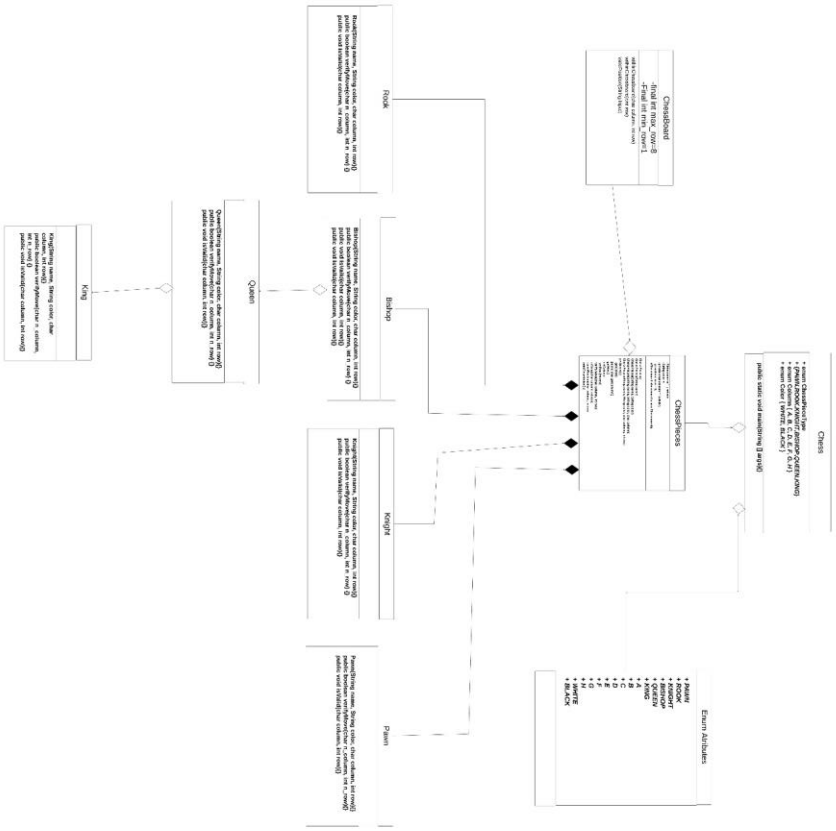
Design Model

<<Create your class diagram for LAB 3 and include attributes, methods, access levels, associations, and multiplicity. Describe your model.>>

a) **Model description**

in this Program we have the class chess which is the class where the whole program is going to execute. We have the Chesspieces class which will serve as the parent class for all our pieces. They will inherit their methods from it. We have the enum class which will allow us to use column color and piece as enum types in our main class. We have the chessboard class which servers a way to verify that the position in fact exits in our chessboard. All of other classes are the methods for the pieces PAWN, ROOK, KNIGHT, BISHOP, QUEEN, KING. The queen inherits from the bishop which helps simplify the behavior of the queen. The king inherits from the queen so we will have a hierarchy king<queen<bishop<Chessboard. Using this way we can save methods reusing our super classes. We can create in our main an array of chesspieces so we can user methods which are overriding their parent classes.

b.


Chess
ChessBoard
ChessPieces
Enum Attributes
Rook
Bishop
Queen
King
Knight
Pawn