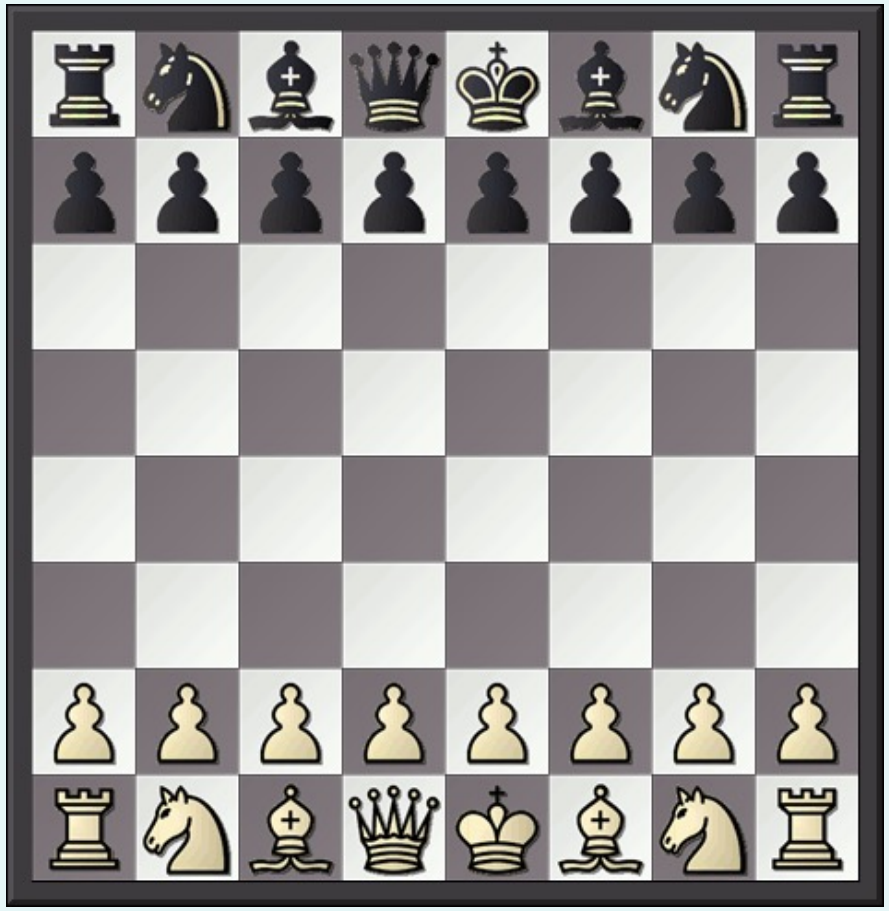
Northeastern University

Chess Game

Zheng Qiao

CS 5004

Rules



Rook

A rook can move horizontally or vertically. It can kill any opponent's piece if it can move to its place.

Knight

A knight can move only in an L pattern: two cells horizontally and one vertically or vice versa. It can kill any opponent's piece if it can move to its place.

Bishop

A bishop can only move diagonally, and kill any opponent's piece if it can move to its place.

Queen

A queen can move horizontally, vertically and diagonally. It can kill any opponent's piece if it can move to its place.

King

A king can move one square in any direction: horizontally, vertically, or diagonally. It can capture an opponent's piece if it can move to its place

Pawn

A pawn can move only "ahead", not backwards towards where its color started. It can move only one place forward in its own column.
However to kill, it must move one place forward diagonally (it cannot kill by moving straight).

Key aspects of code(sample)

Northeastern University

```
// judgement
if (chessBoard2.board[x1][y1].canMove(x2, y2)) {

    // judgement for King
    if (chessBoard2.board[x1][y1].toString().equals("King  ")) {
        finalWhiteRow = x2;
        finalWhiteCol = y2;
    }

    chessBoard2.board[x1][y1].setCol(y2);
    chessBoard2.board[x1][y1].setRow(x2);
    chessBoard2.board[x2][y2] = chessBoard2.board[x1][y1];
    chessBoard2.board[x1][y1] = null;
    isPrimePieceWhite = true;
} else {
    System.out.println("cannot move to this location! ! ! ");
}
```

```
// judgement
if (chessBoard2.board[x1][y1].canKill(chessBoard2.board[x2][y2])) {

    // judgement for King
    if (chessBoard2.board[x1][y1].toString().equals("King  ")) {
        finalWhiteRow = x2;
        finalWhiteCol = y2;
    }

    // If the opposing king is killed, white wins and the game ends
    if (x2 == finalBlackRow && y2 == finalBlackCol) {
        System.out.println("White wins, game over! ");
        isPrime = true;
        break;
    }

    chessBoard2.board[x1][y1].setCol(y2);
    ChessBoard2.board[x1][y1].setRow(x2);
    chessBoard2.board[x2][y2] = chessBoard2.board[x1][y1];
    chessBoard2.board[x1][y1] = null;
    isPrimePieceWhite = true;
} else {
    System.out.println("cannot kill this piece");
}
```

Highlight a few big ideas of OOD

Northeastern University

For each chess piece, I implemented some important ideas of OOD. Here I only take the Rook class as an example:

Encapsulation: The Rook class has private instance variables (row, col, color) and public getter and setter methods to access and modify them.

Abstraction: The interface ChessPiece declare some methods that any class representing a chess piece should implement. The Rook class provides its own implementation for these methods.


CHESS CLASH

Findings

Northeastern University

"Everybody should learn how to program because it teaches you how to think" --Steve Jobs

Lessons Learned

- Learn to balance challenge and interest versus realism
- Debugging is important

Limitations

- Javadocs and unit test are not sufficient
- There are still some bugs
- Redundant code

Future Extensions

- Add more rules
- Add UI parts, deploy it on the website

Resources and Citations

https://en.wikipedia.org/wiki/Chess

https://www.w3schools.com/

Northeastern University

Thank you!

Zheng Qiao

CS 5004