```java
package Project3;

/**********************************************************************
 * CIS 163 Section 01
 * Project 3: Chess Game
 * Rook Class
 *
 * This class represents a ChessPiece that is a rook.
 *
 * @author George Fayette
 * @version 3/23/2019
 *********************************************************************/
public class Rook extends ChessPiece {

    /**
     * Public boolean representing whether or not the rook has moved.
     */
    public boolean firstMove;

    /*****************************************************************
     * Public constructor sets player to parameter value.
     * @param player The player type.
     ****************************************************************/
    public Rook(Player player) {
        super(player);
        firstMove = true;
    }

    /*****************************************************************
     * Public String, returns the ChessPiece type.
     * @return A string representing the ChessPiece type.
     ****************************************************************/
    public String type() {
        return "Rook";
    }

    /*****************************************************************
     * Public boolean, returns true if the move is valid.
     * @param move The move that is being checked.
     * @param board The array of IChessPieces that is being checked.
     * @return True if the move is valid.
     ****************************************************************/
    public boolean isValidMove(Move move, IChessPiece[][] board) {
        boolean valid = true;

        if (!super.isValidMove(move, board)) {
            valid = false;
        } else {
            int vDistance = move.toRow - move.fromRow;
            int hDistance = move.toColumn - move.fromColumn;

            if (vDistance != 0 && hDistance != 0) {
                valid = false;
            } else {
                int absDistance = Math.abs(vDistance + hDistance);
                if (vDistance > 0) {
                    for (int i = 1; i < absDistance; ++i) {
                        if (board[move.fromRow + i][move.fromColumn] !=
                                null) {
                            valid = false;
```

```
61                          }
62                      }
63                  } else if (vDistance < 0) {
64                      for (int i = 1; i < absDistance; ++i) {
65                          if (board[move.fromRow - i][move.fromColumn] !=
66                                  null) {
67                              valid = false;
68                          }
69                      }
70                  } else if (hDistance > 0) {
71                      for (int i = 1; i < absDistance; ++i) {
72                          if (board[move.fromRow][move.fromColumn + i] !=
73                                  null) {
74                              valid = false;
75                          }
76                      }
77                  } else if (hDistance < 0) {
78                      for (int i = 1; i < absDistance; ++i) {
79                          if (board[move.fromRow][move.fromColumn - i] !=
80                                  null) {
81                              valid = false;
82                          }
83                      }
84                  }
85              }
86          }
87          return valid;
88      }
89  }
90
```