Move function

Test Case Description	Input	Expected Output	Reasoning
Try to move a valid piece present	Current location coordinates and to coordinates	valid	Checks basic functionality of moving a valid player piece from current location and moves to a valid destination location following the standards of chess
Try to move an invalid piece	Current location coordinates	False	Checks if one is moving the correct piece (Color)and not the other player's piece or Color
Try to move a non existent piece in the given location to a destination	A location coordinates where no piece is occupying	False	Tests and Identifies the given location is valid for any piece.
Capture your own piece	Move to coordinates	False	Checks basic functionality, of capturing one's own pieces
Try moving piece off the grid	Move to coordinates	False	Checks basic functionality of moving of the grid.
Promote a pawn upon reaching the opposite end	A white or black pawn at the appropriate end row (7 for white, 0 for black)	True, promotes to Queen	Checks that the pawn promotion to Queen occurs upon reaching the last row.

Nodes and edges

1 -> 2: Start to assigning piece from from_coord

2 -> 3: After assigning piece, go to check if piece is None

3 -> 4: If piece is None, go to Return False

3 -> 5: If piece is not None, go to check piece.color

5 -> 6: If piece.color does not match color, go to Return False

5 -> 7: If piece.color matches color, assign to_piece

7 -> 8: After assigning to_piece, check if to_piece exists and has the same color

8 -> 9: If to_piece exists and has the same color, go to Return False

8 -> 10: If to_piece does not have the same color, check if to_coord is out of bounds

10 -> 11: If to_coord is out of bounds, go to Return False

10 -> 12: If to_coord is within bounds, call piece.move(to_coord) and assign result to valid

12 -> 13: If valid is True, proceed with updating the board

12 -> 21: If valid is False, return valid

13 -> 14: Set from_coord to None

14 -> 15: Set piece to to_coord

15 -> 16: Update piece.position

16 -> 17: Check if piece is a pawn

17 -> 18: If piece is a pawn, check if color is 'white' and to_coord.row is 7

17 -> 21: If piece is not a pawn, skip to return

18 -> 19: If color is 'white' and to_coord.row is 7, promote to Queen

18 -> 21: If color is not 'white' or row is not 7, go to Return

19 -> 21: If color is 'black' and to_coord.row is 0, promote to Queen

19 -> 21: If color is not 'black' or row is not 0, go to Return

21 -> End: Return valid

Cyclomatic Complexity;

26 edges – 21 nodes + 2p = 7

In Check function

Test Cases Description	Input	Expected Output	Reasoning
Try to see if color is valid	color	True	Checks basic functionality of valid color for the current player
Try and see if color is invalid	color	False	Checks basic functionality of invalid color
Check if king is on edge	Current coordinates of enemy player pieces	In_check true	Checks basic functionality of the kings when threatened by diagonal enemies when found on the edge of the game board.
Check if king is in corner 2 enemies check	Current position for player pieces	in_check true	Checks basic functionality of the kings 2 optional moves when cornered
Kings own piece covering	Kings is protected by its own piece	King not in check False	Checks if the king is in a safe place.
King is the only piece on board	Board game is empty	King is not in check False	Checks if king is the only piece on the board
King is within a valid attack position	An enemy can attack king	King In check	Checks if a enemy is within attack range.
King not in check if allied piece can move	King is within player attack and ally piece can move to block other player	King not in check False	Check if allied piece can block opposing player

Nodes

1. Start
2. Get King Position
3. Outer Loop (Row)
4. Inner Loop (Column)
5. Piece Assignment
6. Check Piece Existence

7. Check for Check
8. Return True
9. Return False back to outer loop

Edges

1 -> 2: Start to Get the position of the King
2-> 3: Get the position of the King to Start the outer loop
3-> 4: Start the outer loop to Start the Inner loop
4-> 5: Start the inner loop to Get the piece at the current position
5-> 6: Get the piece at the current position to Check if the piece is an enemy
6->7: Check if the piece is an enemy to if in check return True
6->4: Check if the piece is an enemy to continue the inner loop
4->3: After the inner loop ends, go back to check the next row
3-> End: if all rows are checked, return False(the king is not checked)

Cyclomatic Complexity;

9 edges – 8 nodes + 2p = 3