using System;

using System.Text;

namespace Chess

{

internal class Play

{

static void Main(string[] args)

{

Console.OutputEncoding = Encoding.Unicode;

int boardX = 0;

int boardY = 0;

bool noSymbols = false;

bool sound = true;

bool showMovement = false;

// Selections

string[] pieceSelect = new string[] { "pawn1", "pawn2", "pawn3", "pawn4", "pawn5", "pawn6", "pawn7", "pawn8", "rook1", "rook2", "knight1", "knight2", "bishop1", "bishop2", "king", "queen",

"castle", "shortcastle", "castleshort", "longcastle", "castlelong", "castleking", "castlequeen", "castlekingside", "castlequeenside", "castlek", "castleq", "enpassant", "ep", "promote", "promotion",

"exit", "resign", "forfeit", "draw", "grey", "gray", "green", "blue", "red", "yellow", "purple", "text", "letters", "symbols", "icons", "show", "mute", "unmute", "clear" };

char[] letterSelect = new char[] { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i' };

int[] numberSelect = new int[] { 1, 2, 3, 4, 5, 6, 7, 8 };

while (true)

{

// Initialize All Pieces

// White Pieces

King WhiteKing = new King(); WhiteKing.Y = 1;

Queen WhiteQueen = new Queen(); WhiteQueen.Y = 1;

Bishop WhiteBishop1 = new Bishop(); WhiteBishop1.X = 3; WhiteBishop1.Y = 1;

Bishop WhiteBishop2 = new Bishop(); WhiteBishop2.X = 6; WhiteBishop2.Y = 1;

Knight WhiteKnight1 = new Knight(); WhiteKnight1.X = 2; WhiteKnight1.Y = 1;

Knight WhiteKnight2 = new Knight(); WhiteKnight2.X = 7; WhiteKnight2.Y = 1;

Rook WhiteRook1 = new Rook(); WhiteRook1.X = 1; WhiteRook1.Y = 1;

Rook WhiteRook2 = new Rook(); WhiteRook2.X = 8; WhiteRook2.Y = 1;

// White Pawns

Pawn WhitePawn1 = new Pawn(); WhitePawn1.X = 1; WhitePawn1.Y = 2;

Pawn WhitePawn2 = new Pawn(); WhitePawn2.X = 2; WhitePawn2.Y = 2;

Pawn WhitePawn3 = new Pawn(); WhitePawn3.X = 3; WhitePawn3.Y = 2;

Pawn WhitePawn4 = new Pawn(); WhitePawn4.X = 4; WhitePawn4.Y = 2;

Pawn WhitePawn5 = new Pawn(); WhitePawn5.X = 5; WhitePawn5.Y = 2;

Pawn WhitePawn6 = new Pawn(); WhitePawn6.X = 6; WhitePawn6.Y = 2;

Pawn WhitePawn7 = new Pawn(); WhitePawn7.X = 7; WhitePawn7.Y = 2;

Pawn WhitePawn8 = new Pawn(); WhitePawn8.X = 8; WhitePawn8.Y = 2;

// Black Pieces

King BlackKing = new King(); BlackKing.Y = 8;

Queen BlackQueen = new Queen(); BlackQueen.Y = 8;

Bishop BlackBishop1 = new Bishop(); BlackBishop1.X = 6; BlackBishop1.Y = 8;

Bishop BlackBishop2 = new Bishop(); BlackBishop2.X = 3; BlackBishop2.Y = 8;

Knight BlackKnight1 = new Knight(); BlackKnight1.X = 7; BlackKnight1.Y = 8;

Knight BlackKnight2 = new Knight(); BlackKnight2.X = 2; BlackKnight2.Y = 8;

Rook BlackRook1 = new Rook(); BlackRook1.X = 8; BlackRook1.Y = 8;

Rook BlackRook2 = new Rook(); BlackRook2.X = 1; BlackRook2.Y = 8;

// Black Pawns

Pawn BlackPawn1 = new Pawn(); BlackPawn1.X = 8; BlackPawn1.Y = 7;

Pawn BlackPawn2 = new Pawn(); BlackPawn2.X = 7; BlackPawn2.Y = 7;

Pawn BlackPawn3 = new Pawn(); BlackPawn3.X = 6; BlackPawn3.Y = 7;

Pawn BlackPawn4 = new Pawn(); BlackPawn4.X = 5; BlackPawn4.Y = 7;

Pawn BlackPawn5 = new Pawn(); BlackPawn5.X = 4; BlackPawn5.Y = 7;

Pawn BlackPawn6 = new Pawn(); BlackPawn6.X = 3; BlackPawn6.Y = 7;

Pawn BlackPawn7 = new Pawn(); BlackPawn7.X = 2; BlackPawn7.Y = 7;

Pawn BlackPawn8 = new Pawn(); BlackPawn8.X = 1; BlackPawn8.Y = 7;

// New Pieces

NewPiece NewPiece1 = new NewPiece(); NewPiece NewPiece2 = new NewPiece(); NewPiece NewPiece3 = new NewPiece(); NewPiece NewPiece4 = new NewPiece();

NewPiece NewPiece5 = new NewPiece(); NewPiece NewPiece6 = new NewPiece(); NewPiece NewPiece7 = new NewPiece(); NewPiece NewPiece8 = new NewPiece();

NewPiece NewPiece9 = new NewPiece(); NewPiece NewPiece10 = new NewPiece(); NewPiece NewPiece11 = new NewPiece(); NewPiece NewPiece12 = new NewPiece();

NewPiece NewPiece13 = new NewPiece(); NewPiece NewPiece14 = new NewPiece(); NewPiece NewPiece15 = new NewPiece(); NewPiece NewPiece16 = new NewPiece();

NewPiece[] WhitePromo = new NewPiece[8] { NewPiece1, NewPiece2, NewPiece3, NewPiece4, NewPiece5, NewPiece6, NewPiece7, NewPiece8 };

NewPiece[] BlackPromo = new NewPiece[8] { NewPiece9, NewPiece10, NewPiece11, NewPiece12, NewPiece13, NewPiece14, NewPiece15, NewPiece16 };

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.White;

Console.Clear();

Console.WriteLine("\n♚ Chess♚\n"); // Title

Console.WriteLine("by Kyle Furey\n");

Console.WriteLine("Press any key.");

Console.ReadKey();

if (sound == true)

{

Console.Beep(); // Funny beep

}

Console.BackgroundColor = ConsoleColor.DarkGray; // Default board color

Console.ForegroundColor = ConsoleColor.White; // Text color

string input = "";

string piece = "";

char letter = ' ';

int number = 0;

bool badinput = true;

int turn = 1;

int showTurn = 1;

string previousTurn = "";

string pieceTaken = "";

int enPassantWhite = 0;

int enPassantBlack = 0;

int castleValue = 0;

// Counts are always what is currently on the board plus one.

int queenCountWhite = 2;

int bishopCountWhite = 3;

int knightCountWhite = 3;

int rookCountWhite = 3;

int queenCountBlack = 2;

int bishopCountBlack = 3;

int knightCountBlack = 3;

int rookCountBlack = 3;

string promotedPiece = "";

// Inputs

while (true)

{

if (piece == "resign" || piece == "forfeit") // Resignation

{

if (IsEven(turn) == false) // White loses

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\nBLACK wins by resignation after " + (showTurn - 1) + " turns!\nPress enter to return to the start screen.\n");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadLine();

}

else // Black loses

{

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("\nWHITE wins by resignation after " + (showTurn - 1) + " turns!\nPress enter to return to the start screen.\n");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadLine();

}

break;

}

if (piece == "draw") // Draw

{

if (IsEven(turn) == false) // White loses

{

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("\nThe game was a DRAW after " + showTurn + " turns.\nPress enter to return to the start screen.\n");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadLine();

}

break;

}

if (piece == "exit") // Exit check

{

break;

}

// Render the board

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

badinput = true;

if (Checkmate(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Checkmate test

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White loses

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\nBLACK wins by checkmate after " + showTurn + " turns!\nPress enter to return to the start screen.\n");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadLine();

}

else // Black loses

{

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("\nWHITE wins by checkmate after " + showTurn + " turns!\nPress enter to return to the start screen.\n");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadLine();

}

break;

}

if (IsEven(turn) == false) // White's Turn

{

enPassantWhite = 0; // Reset En Passant for White

}

else // Black's Turn

{

enPassantBlack = 0; // Reset En Passant for Black

}

while (badinput == true)

{

if (piece == "exit") // Exit check

{

break;

}

if (piece == "resign" || piece == "forfeit") // Resignation

{

break;

}

if (piece == "draw") // Draw

{

break;

}

input = "";

piece = "";

letter = ' ';

number = 0;

Console.WriteLine();

piece = SelectPiece(input, pieceSelect, turn, WhitePromo, BlackPromo, // Piece Selection

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

Console.WriteLine();

if (piece.Length == 7) // Algebraic Notation (Pawn, Rook, and Queen)

{

if (piece.Contains("pawn") || piece.Contains("rook") || piece.Contains("queen"))

{

char[] pieceArray = new char[piece.Length];

for (int i = 0; i < pieceArray.Length; i++) // Check each character

{

pieceArray[i] = piece[i];

}

piece = piece.Remove(5);

letter = pieceArray[5];

number = (int)pieceArray[6] - 48; // Note: I do not know why it adds 48.

}

}

if (piece.Length == 8) // Algebraic Notation (Queen with number)

{

if (piece.Contains("queen"))

{

char[] pieceArray = new char[piece.Length];

for (int i = 0; i < pieceArray.Length; i++) // Check each character

{

pieceArray[i] = piece[i];

}

piece = piece.Remove(6);

letter = pieceArray[6];

number = (int)pieceArray[7] - 48; // Note: I do not know why it adds 48.

}

}

if (piece.Length == 9) // Algebraic Notation (Knight and Bishop)

{

if (piece.Contains("knight") || piece.Contains("bishop"))

{

char[] pieceArray = new char[piece.Length];

for (int i = 0; i < pieceArray.Length; i++) // Check each character

{

pieceArray[i] = piece[i];

}

piece = piece.Remove(7);

letter = pieceArray[7];

number = (int)pieceArray[8] - 48; // Note: I do not know why it adds 48.

}

}

if (piece.Length == 6) // Algebraic Notation (King)

{

if (piece.Contains("king"))

{

char[] pieceArray = new char[piece.Length];

for (int i = 0; i < pieceArray.Length; i++) // Check each character

{

pieceArray[i] = piece[i];

}

piece = piece.Remove(4);

letter = pieceArray[4];

number = (int)pieceArray[5] - 48; // Note: I do not know why it adds 48.

}

}

if (piece == "enpassant" || piece == "ep") // En Passant

{

string inputEP;

inputEP = FindEnPassant(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

piece = "pawn" + inputEP[0];

letter = inputEP[1];

number = inputEP[2] - 48; // Note: I do not know why it adds 48.

}

castleValue = 0; // Castling

if (piece == "castle") // Castle Kingside (unless Queenside is only available)

{

piece = "king";

letter = 'c';

if (IsEven(turn) == false) // White's turn

{

number = 1;

}

else // Black's turn

{

number = 8;

}

if (CastlingTest(turn, enPassantWhite, enPassantBlack, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

letter = 'g';

}

}

if (piece == "o-o" || piece == "0-0" || piece == "oo" || piece == "00" ||

piece == "shortcastle" || piece == "castleshort" || piece == "castleking" || piece == "castlekingside" || piece == "castlek")

{ // Castle Kingside

piece = "king";

letter = 'g';

if (IsEven(turn) == false) // White's turn

{

number = 1;

}

else // Black's turn

{

number = 8;

}

}

if (piece == "o-o-o" || piece == "0-0-0" || piece == "ooo" || piece == "000" ||

piece == "longcastle" || piece == "castlelong" || piece == "castlequeen" || piece == "castlequeenside" || piece == "castleq")

{ // Castle Queenside

piece = "king";

letter = 'c';

if (IsEven(turn) == false) // White's turn

{

number = 1;

}

else // Black's turn

{

number = 8;

}

}

if (piece == "promote" || piece == "promotion") // Promotion

{

string promote = FindPromotion(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

piece = "pawn" + promote[0];

letter = promote[1];

number = (int)promote[2] - 48; // Note: I do not know why it adds 48.

}

while (letter == ' ') // Other inputs

{

if (piece == "exit") // Exit check

{

break;

}

if (piece == "resign" || piece == "forfeit") // Resignation

{

break;

}

if (piece == "draw") // Draw

{

break;

}

if (piece == "gray" || piece == "grey") // Gray board

{

Console.BackgroundColor = ConsoleColor.DarkGray;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "green") // Green board

{

Console.BackgroundColor = ConsoleColor.DarkGreen;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "blue") // Blue board

{

Console.BackgroundColor = ConsoleColor.DarkCyan;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "red") // Red board

{

Console.BackgroundColor = ConsoleColor.Red;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "yellow") // Yellow board

{

Console.BackgroundColor = ConsoleColor.DarkYellow;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "purple") // Purple board

{

Console.BackgroundColor = ConsoleColor.DarkMagenta;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "text" || piece == "letters")

{ // Type

noSymbols = true;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "symbols" || piece == "icons") // Symbols

{

noSymbols = false;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "mute") // Beep off

{

sound = false;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "unmute") // Beep on

{

sound = true;

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "clear") // Clear board

{

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (piece == "show") // Show movements

{

if (showMovement)

{

showMovement = false;

}

else

{

showMovement = true;

}

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

if (showMovement) // If enabled, highlight the selected piece's possible movement spaces.

{

RenderMovement(boardX, boardY, turn, showTurn, previousTurn, noSymbols, piece, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

else

{

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

Console.Write("Selected Piece: ");

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine(piece);

string letterCheck;

letterCheck = SelectLetter(input, letterSelect);

letter = letterCheck[0]; // Letter selection

Console.WriteLine();

if (letterCheck.Length == 2)

{

number = letterCheck[1];

number = number - 48; // Note: I do not know why it adds 48.

}

if (letter == 'π') // Go back check

{

piece = " ";

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

while (number == 0) // Check a number isn't assigned.

{

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

Console.Write("Selected Letter: ");

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine(char.ToUpper(letter));

number = SelectNumber(input, numberSelect); // Number selection

Console.WriteLine();

if (number == 9) // Go back check

{

piece = " ";

RenderBoard(boardX, boardY, turn, showTurn, previousTurn, noSymbols, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

break;

}

}

}

// Movement

int destinationX = 0;

switch (letter) // X value

{

case 'a':

destinationX = 1;

break;

case 'b':

destinationX = 2;

break;

case 'c':

destinationX = 3;

break;

case 'd':

destinationX = 4;

break;

case 'e':

destinationX = 5;

break;

case 'f':

destinationX = 6;

break;

case 'g':

destinationX = 7;

break;

case 'h':

destinationX = 8;

break;

}

int destinationY = number; // Y Value

badinput = true; // The input must be valid.

string epTest = FindEnPassant(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Store en passant code

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

int occupiedCheck = 0; // Same color occupation check.

if (IsEven(turn) == false)

{

occupiedCheck = 1;

}

else

{

occupiedCheck = 2;

}

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == occupiedCheck)

{

destinationX = 99;

destinationY = -100;

}

int checkX = 0;

int checkY = 0;

switch (piece) // Piece

{

case "king": // White King

if (IsEven(turn) == false)

{

if (WhiteKing.HasMoved == false) // Castling check

{

if (destinationY == 1)

{

if (destinationX == 3)

{

if (CastlingTest(turn, enPassantWhite, enPassantBlack, 2, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

castleValue = 2;

}

}

if (destinationX == 7)

{

if (CastlingTest(turn, enPassantWhite, enPassantBlack, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

castleValue = 1;

}

}

}

}

if (castleValue == 0)

{

if (KingMove(destinationX, destinationY, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteKing.X;

checkY = WhiteKing.Y;

WhiteKing.X = destinationX;

WhiteKing.Y = destinationY;

WhiteKing.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

}

else // Black King

{

if (BlackKing.HasMoved == false) // Castling check

{

if (destinationY == 8)

{

if (destinationX == 3)

{

if (CastlingTest(turn, enPassantWhite, enPassantBlack, 2, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

castleValue = 2;

}

}

if (destinationX == 7)

{

if (CastlingTest(turn, enPassantWhite, enPassantBlack, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

castleValue = 1;

}

}

}

}

if (castleValue == 0)

{

if (KingMove(destinationX, destinationY, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackKing.X;

checkY = BlackKing.Y;

BlackKing.X = destinationX;

BlackKing.Y = destinationY;

BlackKing.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

}

break;

case "queen": // White Queen

if (IsEven(turn) == false)

{

if (QueenMove(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteQueen.X;

checkY = WhiteQueen.Y;

WhiteQueen.X = destinationX;

WhiteQueen.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Queen

{

if (QueenMove(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackQueen.X;

checkY = BlackQueen.Y;

BlackQueen.X = destinationX;

BlackQueen.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "bishop1": // White Bishop1

if (IsEven(turn) == false)

{

if (BishopMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteBishop1.X;

checkY = WhiteBishop1.Y;

WhiteBishop1.X = destinationX;

WhiteBishop1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Bishop1

{

if (BishopMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackBishop1.X;

checkY = BlackBishop1.Y;

BlackBishop1.X = destinationX;

BlackBishop1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "bishop2": // White Bishop2

if (IsEven(turn) == false)

{

if (BishopMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteBishop2.X;

checkY = WhiteBishop2.Y;

WhiteBishop2.X = destinationX;

WhiteBishop2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Bishop2

{

if (BishopMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackBishop2.X;

checkY = BlackBishop2.Y;

BlackBishop2.X = destinationX;

BlackBishop2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "knight1": // White Knight1

if (IsEven(turn) == false)

{

if (KnightMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteKnight1.X;

checkY = WhiteKnight1.Y;

WhiteKnight1.X = destinationX;

WhiteKnight1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Knight1

{

if (KnightMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackKnight1.X;

checkY = BlackKnight1.Y;

BlackKnight1.X = destinationX;

BlackKnight1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "knight2": // White Knight2

if (IsEven(turn) == false)

{

if (KnightMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteKnight2.X;

checkY = WhiteKnight2.Y;

WhiteKnight2.X = destinationX;

WhiteKnight2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Knight2

{

if (KnightMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackKnight2.X;

checkY = BlackKnight2.Y;

BlackKnight2.X = destinationX;

BlackKnight2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "rook1": // White Rook1

if (IsEven(turn) == false)

{

if (RookMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteRook1.X;

checkY = WhiteRook1.Y;

WhiteRook1.X = destinationX;

WhiteRook1.Y = destinationY;

WhiteRook1.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Rook1

{

if (RookMove1(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackRook1.X;

checkY = BlackRook1.Y;

BlackRook1.X = destinationX;

BlackRook1.Y = destinationY;

BlackRook1.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "rook2": // White Rook2

if (IsEven(turn) == false)

{

if (RookMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhiteRook2.X;

checkY = WhiteRook2.Y;

WhiteRook2.X = destinationX;

WhiteRook2.Y = destinationY;

WhiteRook2.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Rook2

{

if (RookMove2(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackRook2.X;

checkY = BlackRook2.Y;

BlackRook2.X = destinationX;

BlackRook2.Y = destinationY;

BlackRook2.HasMoved = true;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn1": // White Pawn1

if (IsEven(turn) == false)

{

if (PawnMove1(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn1.Y)

{

enPassantWhite = 1;

}

checkX = WhitePawn1.X;

checkY = WhitePawn1.Y;

WhitePawn1.X = destinationX;

WhitePawn1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn1

{

if (PawnMove1(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn1.Y - 2)

{

enPassantBlack = 1;

}

checkX = BlackPawn1.X;

checkY = BlackPawn1.Y;

BlackPawn1.X = destinationX;

BlackPawn1.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn2": // White Pawn2

if (IsEven(turn) == false)

{

if (PawnMove2(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn2.Y)

{

enPassantWhite = 2;

}

checkX = WhitePawn2.X;

checkY = WhitePawn2.Y;

WhitePawn2.X = destinationX;

WhitePawn2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn2

{

if (PawnMove2(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn2.Y - 2)

{

enPassantBlack = 2;

}

checkX = BlackPawn2.X;

checkY = BlackPawn2.Y;

BlackPawn2.X = destinationX;

BlackPawn2.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn3": // White Pawn3

if (IsEven(turn) == false)

{

if (PawnMove3(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn3.Y)

{

enPassantWhite = 3;

}

checkX = WhitePawn3.X;

checkY = WhitePawn3.Y;

WhitePawn3.X = destinationX;

WhitePawn3.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn3

{

if (PawnMove3(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn3.Y - 2)

{

enPassantBlack = 3;

}

checkX = BlackPawn3.X;

checkY = BlackPawn3.Y;

BlackPawn3.X = destinationX;

BlackPawn3.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn4": // White Pawn4

if (IsEven(turn) == false)

{

if (PawnMove4(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn4.Y)

{

enPassantWhite = 4;

}

checkX = WhitePawn4.X;

checkY = WhitePawn4.Y;

WhitePawn4.X = destinationX;

WhitePawn4.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn4

{

if (PawnMove4(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn4.Y - 2)

{

enPassantBlack = 4;

}

checkX = BlackPawn4.X;

checkY = BlackPawn4.Y;

BlackPawn4.X = destinationX;

BlackPawn4.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn5": // White Pawn5

if (IsEven(turn) == false)

{

if (PawnMove5(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn5.Y)

{

enPassantWhite = 5;

}

checkX = WhitePawn5.X;

checkY = WhitePawn5.Y;

WhitePawn5.X = destinationX;

WhitePawn5.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn5

{

if (PawnMove5(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn5.Y - 2)

{

enPassantBlack = 5;

}

checkX = BlackPawn5.X;

checkY = BlackPawn5.Y;

BlackPawn5.X = destinationX;

BlackPawn5.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn6": // White Pawn6

if (IsEven(turn) == false)

{

if (PawnMove6(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn6.Y)

{

enPassantWhite = 6;

}

checkX = WhitePawn6.X;

checkY = WhitePawn6.Y;

WhitePawn6.X = destinationX;

WhitePawn6.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn6

{

if (PawnMove6(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn6.Y - 2)

{

enPassantBlack = 6;

}

checkX = BlackPawn6.X;

checkY = BlackPawn6.Y;

BlackPawn6.X = destinationX;

BlackPawn6.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn7": // White Pawn7

if (IsEven(turn) == false)

{

if (PawnMove7(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn7.Y)

{

enPassantWhite = 7;

}

checkX = WhitePawn7.X;

checkY = WhitePawn7.Y;

WhitePawn7.X = destinationX;

WhitePawn7.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn7

{

if (PawnMove7(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn7.Y - 2)

{

enPassantBlack = 7;

}

checkX = BlackPawn7.X;

checkY = BlackPawn7.Y;

BlackPawn7.X = destinationX;

BlackPawn7.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

case "pawn8": // White Pawn8

if (IsEven(turn) == false)

{

if (PawnMove8(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == 2 + WhitePawn8.Y)

{

enPassantWhite = 8;

}

checkX = WhitePawn8.X;

checkY = WhitePawn8.Y;

WhitePawn8.X = destinationX;

WhitePawn8.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

else // Black Pawn8

{

if (PawnMove8(destinationX, destinationY, turn, true, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (destinationY == BlackPawn8.Y - 2)

{

enPassantBlack = 8;

}

checkX = BlackPawn8.X;

checkY = BlackPawn8.Y;

BlackPawn8.X = destinationX;

BlackPawn8.Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

break;

}

if (IsEven(turn) == false) // White Promoted Piece

{

for (int i = 0; i < WhitePromo.Length; i++)

{

if (piece == WhitePromo[i].Tag)

{

if (NewPieceMove(destinationX, destinationY, turn, WhitePromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = WhitePromo[i].X;

checkY = WhitePromo[i].Y;

WhitePromo[i].X = destinationX;

WhitePromo[i].Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

}

}

else // Black Promoted Piece

{

for (int i = 0; i < BlackPromo.Length; i++)

{

if (piece == BlackPromo[i].Tag)

{

if (NewPieceMove(destinationX, destinationY, turn, BlackPromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

checkX = BlackPromo[i].X;

checkY = BlackPromo[i].Y;

BlackPromo[i].X = destinationX;

BlackPromo[i].Y = destinationY;

badinput = false;

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That is not a valid movement. Type 'help' for the key.\n");

Console.ForegroundColor = ConsoleColor.Black;

badinput = true;

}

}

}

}

if (badinput == false) // Is the King in check after this movement?

{

int checkedX = 0;

int checkedY = 0;

if (IsEven(turn) == false) // Which color King are we checking?

{

checkedX = WhiteKing.X;

checkedY = WhiteKing.Y;

}

else

{

checkedX = BlackKing.X;

checkedY = BlackKing.Y;

}

// If the King is now in check, put the piece back.

if (InCheck(turn, enPassantWhite, enPassantBlack, checkedX, checkedY, true, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

badinput = true;

switch (piece)

{

case "king":

if (IsEven(turn) == false)

{

WhiteKing.X = checkX;

WhiteKing.Y = checkY;

}

else

{

BlackKing.X = checkX;

BlackKing.Y = checkY;

}

break;

case "queen":

if (IsEven(turn) == false)

{

WhiteQueen.X = checkX;

WhiteQueen.Y = checkY;

}

else

{

BlackQueen.X = checkX;

BlackQueen.Y = checkY;

}

break;

case "bishop1":

if (IsEven(turn) == false)

{

WhiteBishop1.X = checkX;

WhiteBishop1.Y = checkY;

}

else

{

BlackBishop1.X = checkX;

BlackBishop1.Y = checkY;

}

break;

case "bishop2":

if (IsEven(turn) == false)

{

WhiteBishop2.X = checkX;

WhiteBishop2.Y = checkY;

}

else

{

BlackBishop2.X = checkX;

BlackBishop2.Y = checkY;

}

break;

case "knight1":

if (IsEven(turn) == false)

{

WhiteKnight1.X = checkX;

WhiteKnight1.Y = checkY;

}

else

{

BlackKnight1.X = checkX;

BlackKnight1.Y = checkY;

}

break;

case "knight2":

if (IsEven(turn) == false)

{

WhiteKnight2.X = checkX;

WhiteKnight2.Y = checkY;

}

else

{

BlackKnight2.X = checkX;

BlackKnight2.Y = checkY;

}

break;

case "rook1":

if (IsEven(turn) == false)

{

WhiteRook1.X = checkX;

WhiteRook1.Y = checkY;

}

else

{

BlackRook1.X = checkX;

BlackRook1.Y = checkY;

}

break;

case "rook2":

if (IsEven(turn) == false)

{

WhiteRook2.X = checkX;

WhiteRook2.Y = checkY;

}

else

{

BlackRook2.X = checkX;

BlackRook2.Y = checkY;

}

break;

case "pawn1":

if (IsEven(turn) == false)

{

WhitePawn1.X = checkX;

WhitePawn1.Y = checkY;

}

else

{

BlackPawn1.X = checkX;

BlackPawn1.Y = checkY;

}

break;

case "pawn2":

if (IsEven(turn) == false)

{

WhitePawn2.X = checkX;

WhitePawn2.Y = checkY;

}

else

{

BlackPawn2.X = checkX;

BlackPawn2.Y = checkY;

}

break;

case "pawn3":

if (IsEven(turn) == false)

{

WhitePawn3.X = checkX;

WhitePawn3.Y = checkY;

}

else

{

BlackPawn3.X = checkX;

BlackPawn3.Y = checkY;

}

break;

case "pawn4":

if (IsEven(turn) == false)

{

WhitePawn4.X = checkX;

WhitePawn4.Y = checkY;

}

else

{

BlackPawn4.X = checkX;

BlackPawn4.Y = checkY;

}

break;

case "pawn5":

if (IsEven(turn) == false)

{

WhitePawn5.X = checkX;

WhitePawn5.Y = checkY;

}

else

{

BlackPawn5.X = checkX;

BlackPawn5.Y = checkY;

}

break;

case "pawn6":

if (IsEven(turn) == false)

{

WhitePawn6.X = checkX;

WhitePawn6.Y = checkY;

}

else

{

BlackPawn6.X = checkX;

BlackPawn6.Y = checkY;

}

break;

case "pawn7":

if (IsEven(turn) == false)

{

WhitePawn7.X = checkX;

WhitePawn7.Y = checkY;

}

else

{

BlackPawn7.X = checkX;

BlackPawn7.Y = checkY;

}

break;

case "pawn8":

if (IsEven(turn) == false)

{

WhitePawn8.X = checkX;

WhitePawn8.Y = checkY;

}

else

{

BlackPawn8.X = checkX;

BlackPawn8.Y = checkY;

}

break;

}

if (IsEven(turn) == false)

{

for (int i = 0; i < WhitePromo.Length; i++)

{

if (piece == WhitePromo[i].Tag)

{

WhitePromo[i].X = checkX;

WhitePromo[i].Y = checkY;

}

}

}

else

{

for (int i = 0; i < BlackPromo.Length; i++)

{

if (piece == BlackPromo[i].Tag)

{

BlackPromo[i].X = checkX;

BlackPromo[i].Y = checkY;

}

}

}

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("That move puts your King in check.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

if (badinput == false) // Capturing a piece

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

pieceTaken = CapturePiece(destinationX, destinationY, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

}

string epTest2 = ""; // En passant capture test (capturing pawn's number, destination letter coordinate, destination number coordinate)

if (piece.Length == 5)

{

epTest2 = epTest2 + piece[4];

epTest2 = epTest2 + letter;

epTest2 = epTest2 + number.ToString();

}

if (badinput == true)

{

epTest2 = "";

}

if (epTest == epTest2)

{

if (IsEven(turn) == false) // White's turn

{

pieceTaken = " captures pawn" + enPassantBlack + " (en passant)";

}

else // Black's turn

{

pieceTaken = " captures pawn" + enPassantWhite + " (en passant)";

}

}

if (castleValue == 1) // Castling

{

if (IsEven(turn) == false) // White's turn

{

badinput = false;

WhiteKing.X = 7;

WhiteKing.HasMoved = true;

WhiteRook2.X = 6;

WhiteRook2.HasMoved = true;

}

else // Black's turn

{

badinput = false;

BlackKing.X = 7;

BlackKing.HasMoved = true;

BlackRook1.X = 6;

BlackRook1.HasMoved = true;

}

}

if (castleValue == 2)

{

if (IsEven(turn) == false) // White's turn

{

badinput = false;

WhiteKing.X = 3;

WhiteKing.HasMoved = true;

WhiteRook1.X = 4;

WhiteRook1.HasMoved = true;

}

else // Black's turn

{

badinput = false;

BlackKing.X = 3;

BlackKing.HasMoved = true;

BlackRook2.X = 4;

BlackRook2.HasMoved = true;

}

}

if (Promotion(turn, // Check for promotions

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.WriteLine();

promotedPiece = CreateNewPiece(WhitePromo, BlackPromo,

queenCountWhite, bishopCountWhite, knightCountWhite, rookCountWhite,

queenCountBlack, bishopCountBlack, knightCountBlack, rookCountBlack,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

if (IsEven(turn) == false)

{

switch (promotedPiece[0])

{

case 'q':

queenCountWhite++;

break;

case 'b':

bishopCountWhite++;

break;

case 'k':

knightCountWhite++;

break;

case 'r':

rookCountWhite++;

break;

}

}

else

{

switch (promotedPiece[0])

{

case 'q':

queenCountBlack++;

break;

case 'b':

bishopCountBlack++;

break;

case 'k':

knightCountBlack++;

break;

case 'r':

rookCountBlack++;

break;

}

}

}

}

// Write Previous Turn + Next Turn

if (badinput == false)

{

if (piece != "exit" && piece != "resign" && piece != "forfeit" && piece != "draw" && piece != "green" && piece != "blue" && piece != "gray" && piece != "grey" && piece != "text" &&

piece != "type" && piece != "letter" && piece != "symbol" && piece != "mute" && piece != "unmute" && piece != "clear" && piece != "show" && (letter != 'π') && (number != 9))

{

previousTurn = showTurn + ". " + piece + " " + letter + number + pieceTaken;

// Special moves

if (castleValue == 1)

{

previousTurn = showTurn + ". O-O (castle kingside)";

}

if (castleValue == 2)

{

previousTurn = showTurn + ". O-O-O (castle queenside)";

}

if (piece == "pawn1" || piece == "pawn2" || piece == "pawn3" || piece == "pawn4" || piece == "pawn5" || piece == "pawn6" || piece == "pawn7" || piece == "pawn8")

{

if (piece[0] == 'p' && (number == 1 || number == 8))

{

previousTurn = showTurn + ". " + piece + " " + letter + number + pieceTaken + " promotes to " + promotedPiece;

}

}

if (sound == true)

{

Console.Beep(); // Funny beep

}

turn++; // Next turn

if (IsEven(turn) == false) // Turn shown

{

showTurn++;

}

}

}

}

}

}

// Render Board and Pieces

static bool IsEven(int input) // Even Integer Check

{

if ((input - (input / 2)) == (input / 2)) // Is the integer subtracted by itself in half is equal to the integer in half?

{

return true; // The integer is even

}

else

{

return false; // The integer is odd

}

}

static void RenderBoard(int boardX, int boardY, int turn, int showTurn, string previousTurn, bool type, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo,

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{ // Render the Current Board

Console.Clear(); // Clear text

if (turn > 1) // Write previous turn

{

string points = " +0"; // Calculate point advantage

int points1 = Points(turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

int points2 = Points(turn + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

if (points1 >= points2)

{

points = " +" + (points1 - points2).ToString();

}

else

{

points = " " + (points1 - points2).ToString();

}

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("\nPrevious Turn\n" + previousTurn);

Console.ForegroundColor = ConsoleColor.DarkRed;

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write(" check");

}

if (Checkmate(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write("mate");

}

Console.ForegroundColor = ConsoleColor.White;

Console.Write(points);

Console.WriteLine("\n");

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("\nPrevious Turn\n" + previousTurn);

Console.ForegroundColor = ConsoleColor.DarkRed;

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write(" check");

}

if (Checkmate(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write("mate");

}

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(points);

Console.WriteLine("\n");

}

}

else

{

Console.WriteLine("\n\n\n");

}

if (IsEven(turn) == false) // White's Turn

{

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine(" Turn " + (showTurn));

for (boardY = 8; boardY > 0; boardY--) // For each row

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(boardY); // Write row number (y value)

Console.Write(" "); // Indent

for (boardX = 1; boardX < 9; boardX++) // For each column

{

if (RenderPieces(boardX, boardY, type, Console.BackgroundColor, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8)

== true) // Is there a piece present?

{

// Render that piece

}

else

{

if (IsEven(boardY) == true) // Check if the row is even or odd

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■ "); // Empty black space

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■ "); // Empty white space

}

}

else

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■ "); // Empty white space

}

else

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■ "); // Empty black space

}

}

}

}

Console.WriteLine();

}

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\n A B C D E F G H\n"); // Write column numbers (x values)

}

else // Black's Turn

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine(" Turn " + showTurn);

for (boardY = 1; boardY < 9; boardY++) // For each row

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(boardY); // Write row number (y value)

Console.Write(" "); // Indent

for (boardX = 8; boardX > 0; boardX--) // For each column

{

if (RenderPieces(boardX, boardY, type, Console.BackgroundColor, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8)

== true) // Is there a piece present?

{

// Render that piece

}

else

{

if (IsEven(boardY) == true) // Check if the row is even or odd

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■ "); // Empty black space

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■ "); // Empty white space

}

}

else

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■ "); // Empty white space

}

else

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■ "); // Empty black space

}

}

}

}

Console.WriteLine();

}

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\n H G F E D C B A\n"); // Write column numbers (x values)

}

}

static bool RenderPieces(int boardX, int boardY, bool type, ConsoleColor BackgroundColor, NewPiece[] WhitePromo, NewPiece[] BlackPromo,

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

// Render the Current Pieces

if (boardX == WhiteKing.X && boardY == WhiteKing.Y) // Render White King

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♚");

}

else

{

Console.Write("K");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (boardX == WhiteQueen.X && boardY == WhiteQueen.Y) // Render White Queen

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♛");

}

else

{

Console.Write("Q");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == WhiteBishop1.X && boardY == WhiteBishop1.Y) || (boardX == WhiteBishop2.X && boardY == WhiteBishop2.Y))

{ // Render White Bishops

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♝");

}

else

{

Console.Write("B");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == WhiteKnight1.X && boardY == WhiteKnight1.Y) || (boardX == WhiteKnight2.X && boardY == WhiteKnight2.Y))

{ // Render White Knights

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♞");

}

else

{

Console.Write("N");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == WhiteRook1.X && boardY == WhiteRook1.Y) || (boardX == WhiteRook2.X && boardY == WhiteRook2.Y))

{ // Render White Rooks

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♜");

}

else

{

Console.Write("R");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == WhitePawn1.X && boardY == WhitePawn1.Y) || (boardX == WhitePawn2.X && boardY == WhitePawn2.Y) || (boardX == WhitePawn3.X && boardY == WhitePawn3.Y) ||

(boardX == WhitePawn4.X && boardY == WhitePawn4.Y) || (boardX == WhitePawn5.X && boardY == WhitePawn5.Y) || (boardX == WhitePawn6.X && boardY == WhitePawn6.Y) ||

(boardX == WhitePawn7.X && boardY == WhitePawn7.Y) || (boardX == WhitePawn8.X && boardY == WhitePawn8.Y))

{ // Render White Pawns

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♙");

}

else

{

Console.Write("P");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (boardX == BlackKing.X && boardY == BlackKing.Y) // Render Black King

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♚");

}

else

{

Console.Write("K");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (boardX == BlackQueen.X && boardY == BlackQueen.Y) // Render Black Queen

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♛");

}

else

{

Console.Write("Q");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == BlackBishop1.X && boardY == BlackBishop1.Y) || (boardX == BlackBishop2.X && boardY == BlackBishop2.Y))

{ // Render Black Bishops

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♝");

}

else

{

Console.Write("B");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == BlackKnight1.X && boardY == BlackKnight1.Y) || (boardX == BlackKnight2.X && boardY == BlackKnight2.Y))

{ // Render Black Knights

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♞");

}

else

{

Console.Write("N");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == BlackRook1.X && boardY == BlackRook1.Y) || (boardX == BlackRook2.X && boardY == BlackRook2.Y))

{ // Render Black Rooks

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♜");

}

else

{

Console.Write("R");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if ((boardX == BlackPawn1.X && boardY == BlackPawn1.Y) || (boardX == BlackPawn2.X && boardY == BlackPawn2.Y) || (boardX == BlackPawn3.X && boardY == BlackPawn3.Y) ||

(boardX == BlackPawn4.X && boardY == BlackPawn4.Y) || (boardX == BlackPawn5.X && boardY == BlackPawn5.Y) || (boardX == BlackPawn6.X && boardY == BlackPawn6.Y) ||

(boardX == BlackPawn7.X && boardY == BlackPawn7.Y) || (boardX == BlackPawn8.X && boardY == BlackPawn8.Y))

{ // Render Black Pawns

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♙");

}

else

{

Console.Write("P");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

for (int i = 0; i < 8; i++)

{

if (boardX == WhitePromo[i].X && boardY == WhitePromo[i].Y) // Render New White Pieces

{

if (WhitePromo[i].Type == "queen")

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♛");

}

else

{

Console.Write("Q");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (WhitePromo[i].Type == "bishop")

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♝");

}

else

{

Console.Write("B");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (WhitePromo[i].Type == "knight")

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♞");

}

else

{

Console.Write("N");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (WhitePromo[i].Type == "rook")

{

Console.ForegroundColor = ConsoleColor.White;

if (type == false)

{

Console.Write("♜");

}

else

{

Console.Write("R");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

}

if (boardX == BlackPromo[i].X && boardY == BlackPromo[i].Y) // Render New Black Pieces

{

if (BlackPromo[i].Type == "queen")

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♛");

}

else

{

Console.Write("Q");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (BlackPromo[i].Type == "bishop")

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♝");

}

else

{

Console.Write("B");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (BlackPromo[i].Type == "knight")

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♞");

}

else

{

Console.Write("N");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

if (BlackPromo[i].Type == "rook")

{

Console.ForegroundColor = ConsoleColor.Black;

if (type == false)

{

Console.Write("♜");

}

else

{

Console.Write("R");

}

Console.BackgroundColor = BackgroundColor;

Console.Write(" ");

return true;

}

}

}

return false; // Render Empty

}

static void RenderMovement(int boardX, int boardY, int turn, int showTurn, string previousTurn, bool type, string piece, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo,

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{ // Highlight Where Pieces Can Move

Console.Clear(); // Clear text

ConsoleColor BackgroundColor = Console.BackgroundColor; // Background Color

if (turn > 1) // Write previous turn

{

string points = " +0"; // Calculate point advantage

int points1 = Points(turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

int points2 = Points(turn + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

if (points1 >= points2)

{

points = " +" + (points1 - points2).ToString();

}

else

{

points = " " + (points1 - points2).ToString();

}

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("\nPrevious Turn\n" + previousTurn);

Console.ForegroundColor = ConsoleColor.DarkRed;

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write(" check");

}

if (Checkmate(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write("mate");

}

Console.ForegroundColor = ConsoleColor.White;

Console.Write(points);

Console.WriteLine("\n");

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("\nPrevious Turn\n" + previousTurn);

Console.ForegroundColor = ConsoleColor.DarkRed;

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write(" check");

}

if (Checkmate(turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

Console.Write("mate");

}

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(points);

Console.WriteLine("\n");

}

}

else

{

Console.WriteLine("\n\n\n");

}

if (IsEven(turn) == false) // White's Turn

{

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine(" Turn " + (showTurn));

for (boardY = 8; boardY > 0; boardY--) // For each row

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(boardY); // Write row number (y value)

Console.Write(" "); // Indent

for (boardX = 1; boardX < 9; boardX++) // For each column

{

if (piece == "king")

{

if (KingMove(boardX, boardY, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // King movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteKing.X; // Piece's current X

int pieceY = WhiteKing.Y; // Piece's current Y

WhiteKing.X = boardX; // New piece position

WhiteKing.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKing.X = pieceX; // Put piece back

WhiteKing.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteKing.X = pieceX; // Put piece back

WhiteKing.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "queen")

{

if (QueenMove(boardX, boardY, turn, WhitePromo, BlackPromo, // Queen movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteQueen.X; // Piece's current X

int pieceY = WhiteQueen.Y; // Piece's current Y

WhiteQueen.X = boardX; // New piece position

WhiteQueen.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteQueen.X = pieceX; // Put piece back

WhiteQueen.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteQueen.X = pieceX; // Put piece back

WhiteQueen.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "bishop1")

{

if (BishopMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Bishop1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteBishop1.X; // Piece's current X

int pieceY = WhiteBishop1.Y; // Piece's current Y

WhiteBishop1.X = boardX; // New piece position

WhiteBishop1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteBishop1.X = pieceX; // Put piece back

WhiteBishop1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteBishop1.X = pieceX; // Put piece back

WhiteBishop1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "bishop2")

{

if (BishopMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Bishop2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteBishop2.X; // Piece's current X

int pieceY = WhiteBishop2.Y; // Piece's current Y

WhiteBishop2.X = boardX; // New piece position

WhiteBishop2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteBishop2.X = pieceX; // Put piece back

WhiteBishop2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteBishop2.X = pieceX; // Put piece back

WhiteBishop2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "knight1")

{

if (KnightMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Knight1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteKnight1.X; // Piece's current X

int pieceY = WhiteKnight1.Y; // Piece's current Y

WhiteKnight1.X = boardX; // New piece position

WhiteKnight1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKnight1.X = pieceX; // Put piece back

WhiteKnight1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteKnight1.X = pieceX; // Put piece back

WhiteKnight1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "knight2")

{

if (KnightMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Knight2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteKnight2.X; // Piece's current X

int pieceY = WhiteKnight2.Y; // Piece's current Y

WhiteKnight2.X = boardX; // New piece position

WhiteKnight2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKnight2.X = pieceX; // Put piece back

WhiteKnight2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteKnight2.X = pieceX; // Put piece back

WhiteKnight2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "rook1")

{

if (RookMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Rook1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteRook1.X; // Piece's current X

int pieceY = WhiteRook1.Y; // Piece's current Y

WhiteRook1.X = boardX; // New piece position

WhiteRook1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteRook1.X = pieceX; // Put piece back

WhiteRook1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteRook1.X = pieceX; // Put piece back

WhiteRook1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "rook2")

{

if (RookMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Rook2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhiteRook2.X; // Piece's current X

int pieceY = WhiteRook2.Y; // Piece's current Y

WhiteRook2.X = boardX; // New piece position

WhiteRook2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteRook2.X = pieceX; // Put piece back

WhiteRook2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhiteRook2.X = pieceX; // Put piece back

WhiteRook2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn1")

{

if (PawnMove1(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn1.X; // Piece's current X

int pieceY = WhitePawn1.Y; // Piece's current Y

WhitePawn1.X = boardX; // New piece position

WhitePawn1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn1.X = pieceX; // Put piece back

WhitePawn1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn1.X = pieceX; // Put piece back

WhitePawn1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn2")

{

if (PawnMove2(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn2.X; // Piece's current X

int pieceY = WhitePawn2.Y; // Piece's current Y

WhitePawn2.X = boardX; // New piece position

WhitePawn2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn2.X = pieceX; // Put piece back

WhitePawn2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn2.X = pieceX; // Put piece back

WhitePawn2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn3")

{

if (PawnMove3(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn3 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn3.X; // Piece's current X

int pieceY = WhitePawn3.Y; // Piece's current Y

WhitePawn3.X = boardX; // New piece position

WhitePawn3.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn3.X = pieceX; // Put piece back

WhitePawn3.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn3.X = pieceX; // Put piece back

WhitePawn3.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn4")

{

if (PawnMove4(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn4 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn4.X; // Piece's current X

int pieceY = WhitePawn4.Y; // Piece's current Y

WhitePawn4.X = boardX; // New piece position

WhitePawn4.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn4.X = pieceX; // Put piece back

WhitePawn4.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn4.X = pieceX; // Put piece back

WhitePawn4.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn5")

{

if (PawnMove5(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn5 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn5.X; // Piece's current X

int pieceY = WhitePawn5.Y; // Piece's current Y

WhitePawn5.X = boardX; // New piece position

WhitePawn5.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn5.X = pieceX; // Put piece back

WhitePawn5.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn5.X = pieceX; // Put piece back

WhitePawn5.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn6")

{

if (PawnMove6(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn6 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn6.X; // Piece's current X

int pieceY = WhitePawn6.Y; // Piece's current Y

WhitePawn6.X = boardX; // New piece position

WhitePawn6.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn6.X = pieceX; // Put piece back

WhitePawn6.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn6.X = pieceX; // Put piece back

WhitePawn6.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn7")

{

if (PawnMove7(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn7 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn7.X; // Piece's current X

int pieceY = WhitePawn7.Y; // Piece's current Y

WhitePawn7.X = boardX; // New piece position

WhitePawn7.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn7.X = pieceX; // Put piece back

WhitePawn7.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn7.X = pieceX; // Put piece back

WhitePawn7.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn8")

{

if (PawnMove8(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn8 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePawn8.X; // Piece's current X

int pieceY = WhitePawn8.Y; // Piece's current Y

WhitePawn8.X = boardX; // New piece position

WhitePawn8.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn8.X = pieceX; // Put piece back

WhitePawn8.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePawn8.X = pieceX; // Put piece back

WhitePawn8.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (piece == WhitePromo[i].Tag)

{

if (NewPieceMove(boardX, boardY, turn, WhitePromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 1)

{

int pieceX = WhitePromo[i].X; // Piece's current X

int pieceY = WhitePromo[i].Y; // Piece's current Y

WhitePromo[i].X = boardX; // New piece position

WhitePromo[i].Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePromo[i].X = pieceX; // Put piece back

WhitePromo[i].Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

WhitePromo[i].X = pieceX; // Put piece back

WhitePromo[i].Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

}

ConsoleColor SpaceColor; // Used to color empty spaces after Pieces are rendered.

SpaceColor = Console.BackgroundColor;

if (RenderPieces(boardX, boardY, type, SpaceColor, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8)

== true) // Is there a piece present?

{

// Render that piece

}

else

{

if (IsEven(boardY) == true) // Check if the row is even or odd

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■"); // Empty black space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■"); // Empty white space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

}

else

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■"); // Empty white space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

else

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■"); // Empty black space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

}

}

Console.BackgroundColor = BackgroundColor;

}

Console.WriteLine();

}

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\n A B C D E F G H\n"); // Write column numbers (x values)

}

else // Black's Turn

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine(" Turn " + showTurn);

for (boardY = 1; boardY < 9; boardY++) // For each row

{

Console.BackgroundColor = BackgroundColor;

Console.ForegroundColor = ConsoleColor.Black;

Console.Write(boardY); // Write row number (y value)

Console.Write(" "); // Indent

for (boardX = 8; boardX > 0; boardX--) // For each column

{

if (piece == "king")

{

if (KingMove(boardX, boardY, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // King movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackKing.X; // Piece's current X

int pieceY = BlackKing.Y; // Piece's current Y

BlackKing.X = boardX; // New piece position

BlackKing.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKing.X = pieceX; // Put piece back

BlackKing.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackKing.X = pieceX; // Put piece back

BlackKing.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "queen")

{

if (QueenMove(boardX, boardY, turn, WhitePromo, BlackPromo, // Queen movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackQueen.X; // Piece's current X

int pieceY = BlackQueen.Y; // Piece's current Y

BlackQueen.X = boardX; // New piece position

BlackQueen.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackQueen.X = pieceX; // Put piece back

BlackQueen.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackQueen.X = pieceX; // Put piece back

BlackQueen.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "bishop1")

{

if (BishopMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Bishop1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackBishop1.X; // Piece's current X

int pieceY = BlackBishop1.Y; // Piece's current Y

BlackBishop1.X = boardX; // New piece position

BlackBishop1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackBishop1.X = pieceX; // Put piece back

BlackBishop1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackBishop1.X = pieceX; // Put piece back

BlackBishop1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "bishop2")

{

if (BishopMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Bishop2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackBishop2.X; // Piece's current X

int pieceY = BlackBishop2.Y; // Piece's current Y

BlackBishop2.X = boardX; // New piece position

BlackBishop2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackBishop2.X = pieceX; // Put piece back

BlackBishop2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackBishop2.X = pieceX; // Put piece back

BlackBishop2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "knight1")

{

if (KnightMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Knight1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackKnight1.X; // Piece's current X

int pieceY = BlackKnight1.Y; // Piece's current Y

BlackKnight1.X = boardX; // New piece position

BlackKnight1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKnight1.X = pieceX; // Put piece back

BlackKnight1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackKnight1.X = pieceX; // Put piece back

BlackKnight1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "knight2")

{

if (KnightMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Knight2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackKnight2.X; // Piece's current X

int pieceY = BlackKnight2.Y; // Piece's current Y

BlackKnight2.X = boardX; // New piece position

BlackKnight2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKnight2.X = pieceX; // Put piece back

BlackKnight2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackKnight2.X = pieceX; // Put piece back

BlackKnight2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "rook1")

{

if (RookMove1(boardX, boardY, turn, WhitePromo, BlackPromo, // Rook1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackRook1.X; // Piece's current X

int pieceY = BlackRook1.Y; // Piece's current Y

BlackRook1.X = boardX; // New piece position

BlackRook1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackRook1.X = pieceX; // Put piece back

BlackRook1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackRook1.X = pieceX; // Put piece back

BlackRook1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "rook2")

{

if (RookMove2(boardX, boardY, turn, WhitePromo, BlackPromo, // Rook2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackRook2.X; // Piece's current X

int pieceY = BlackRook2.Y; // Piece's current Y

BlackRook2.X = boardX; // New piece position

BlackRook2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackRook2.X = pieceX; // Put piece back

BlackRook2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackRook2.X = pieceX; // Put piece back

BlackRook2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn1")

{

if (PawnMove1(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn1.X; // Piece's current X

int pieceY = BlackPawn1.Y; // Piece's current Y

BlackPawn1.X = boardX; // New piece position

BlackPawn1.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn1.X = pieceX; // Put piece back

BlackPawn1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn1.X = pieceX; // Put piece back

BlackPawn1.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn2")

{

if (PawnMove2(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn2.X; // Piece's current X

int pieceY = BlackPawn2.Y; // Piece's current Y

BlackPawn2.X = boardX; // New piece position

BlackPawn2.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn2.X = pieceX; // Put piece back

BlackPawn2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn2.X = pieceX; // Put piece back

BlackPawn2.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn3")

{

if (PawnMove3(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn3 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn3.X; // Piece's current X

int pieceY = BlackPawn3.Y; // Piece's current Y

BlackPawn3.X = boardX; // New piece position

BlackPawn3.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn3.X = pieceX; // Put piece back

BlackPawn3.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn3.X = pieceX; // Put piece back

BlackPawn3.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn4")

{

if (PawnMove4(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn4 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn4.X; // Piece's current X

int pieceY = BlackPawn4.Y; // Piece's current Y

BlackPawn4.X = boardX; // New piece position

BlackPawn4.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn4.X = pieceX; // Put piece back

BlackPawn4.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn4.X = pieceX; // Put piece back

BlackPawn4.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn5")

{

if (PawnMove5(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn5 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn5.X; // Piece's current X

int pieceY = BlackPawn5.Y; // Piece's current Y

BlackPawn5.X = boardX; // New piece position

BlackPawn5.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn5.X = pieceX; // Put piece back

BlackPawn5.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn5.X = pieceX; // Put piece back

BlackPawn5.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn6")

{

if (PawnMove6(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn6 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn6.X; // Piece's current X

int pieceY = BlackPawn6.Y; // Piece's current Y

BlackPawn6.X = boardX; // New piece position

BlackPawn6.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn6.X = pieceX; // Put piece back

BlackPawn6.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn6.X = pieceX; // Put piece back

BlackPawn6.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn7")

{

if (PawnMove7(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn7 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn7.X; // Piece's current X

int pieceY = BlackPawn7.Y; // Piece's current Y

BlackPawn7.X = boardX; // New piece position

BlackPawn7.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn7.X = pieceX; // Put piece back

BlackPawn7.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn7.X = pieceX; // Put piece back

BlackPawn7.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

if (piece == "pawn8")

{

if (PawnMove8(boardX, boardY, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn8 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPawn8.X; // Piece's current X

int pieceY = BlackPawn8.Y; // Piece's current Y

BlackPawn8.X = boardX; // New piece position

BlackPawn8.Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn8.X = pieceX; // Put piece back

BlackPawn8.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPawn8.X = pieceX; // Put piece back

BlackPawn8.Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (piece == BlackPromo[i].Tag)

{

if (NewPieceMove(boardX, boardY, turn, BlackPromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsOccupied(turn, boardX, boardY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) != 2)

{

int pieceX = BlackPromo[i].X; // Piece's current X

int pieceY = BlackPromo[i].Y; // Piece's current Y

BlackPromo[i].X = boardX; // New piece position

BlackPromo[i].Y = boardY; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPromo[i].X = pieceX; // Put piece back

BlackPromo[i].Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkBlue;

}

else

{

BlackPromo[i].X = pieceX; // Put piece back

BlackPromo[i].Y = pieceY; // Put piece back

Console.BackgroundColor = ConsoleColor.DarkRed;

}

}

}

}

}

ConsoleColor SpaceColor; // Used to color empty spaces after Pieces are rendered.

SpaceColor = Console.BackgroundColor;

if (RenderPieces(boardX, boardY, type, SpaceColor, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8)

== true) // Is there a piece present?

{

// Render that piece

}

else

{

if (IsEven(boardY) == true) // Check if the row is even or odd

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■"); // Empty black space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■"); // Empty white space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

}

else

{

if (IsEven(boardX) == true) // Check if the column is even or odd

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write("■"); // Empty white space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

else

{

Console.ForegroundColor = ConsoleColor.Black;

Console.Write("■"); // Empty black space

Console.BackgroundColor = SpaceColor;

Console.Write(" ");

}

}

}

Console.BackgroundColor = BackgroundColor;

}

Console.WriteLine();

}

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\n H G F E D C B A\n"); // Write column numbers (x values)

}

}

// Selections

static string SelectPiece(string input, string[] pieceSelection, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Piece Selection

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

while (true) // Repeat forever

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\nEnter your Piece or Pawn's location."); // Prompt

Console.WriteLine("Type 'help' for the key, and 'exit' to end the game.\n");

Console.ForegroundColor = ConsoleColor.White;

input = Console.ReadLine(); // Get input

input = input.ToLower(); // Make input lowercase

input = input.Replace(" ", string.Empty); // Remove any spaces from input

Console.ForegroundColor = ConsoleColor.Black;

if (input.Length > 1)

{

input = SelectSpace(turn, input, WhitePromo, BlackPromo, // Convert Space to Piece

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

input = InputInterpretation(input, pieceSelection);

if (IsCaptured(turn, input, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

input = "";

}

for (int i = 0; i < pieceSelection.Length; i++) // Check all possible piece selections

{

if (input == pieceSelection[i]) // Does the selection match?

{

return input; // Chosen piece

}

if (input.Length > 3) // Is the input in algebraic notation?

{

if (input.Remove(input.Length - 2, 2) == pieceSelection[i])

{

return input; // Chosen piece and movement

}

}

}

if (IsEven(turn) == false) // White's turn

{

for (int i = 0; i < WhitePromo.Length; i++) // Check all possible new piece selections

{

if (input == WhitePromo[i].Tag) // Does the selction match?

{

return input; // Chosen piece

}

if (input.Length > 3) // Is the input in algebraic notation?

{

if (input.Remove(input.Length - 2, 2) == WhitePromo[i].Tag)

{

return input; // Chosen piece

}

}

}

}

else // Black's turn

{

for (int i = 0; i < BlackPromo.Length; i++) // Check all possible new piece selections

{

if (input == BlackPromo[i].Tag) // Does the selction match?

{

return input; // Chosen piece

}

if (input.Length > 3) // Is the input in algebraic notation?

{

if (input.Remove(input.Length - 2, 2) == BlackPromo[i].Tag)

{

return input; // Chosen piece

}

}

}

}

if (input == "help" || input == "key") // Help prompt

{

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("\nPlease enter your Piece or Pawn's current location.\nYou can do this by typing the letter coordinate followed by the number coordinate.");

Console.WriteLine("\nYou can also use algebraic notation.\nExample: p1a4 = first pawn from left (on game start, your side) to space a4.\n");

Console.WriteLine("The pieces are as follows:");

Console.WriteLine("♟ P = Pawn Moves up one space but captures diagonally forward, promotes upon reaching the top.");

Console.WriteLine("♜ R = Rook Moves up and down or left and right.");

Console.WriteLine("♞ N = Knight Moves in an L pattern and jumps over pieces.");

Console.WriteLine("♝ B = Bishop Moves diagonally.");

Console.WriteLine("♛ Q = Queen Moves any direction.");

Console.WriteLine("♚ K = King Moves any direction one space, can castle with a rook if both haven't moved.");

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\nWhen a King is in check (about to be captured), he must be protected or moved.");

Console.WriteLine("The game ends when a King is in checkmate (cannot be protected that turn).\n");

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("When you are checkmated, enter 'resign'.\n");

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("\nThat is not a valid location or valid piece. Please type 'help' for the key.\n");

}

}

}

static string SelectLetter(string input, char[] letterSelection) // Letter Selection

{

while (true) // Repeat forever

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\nEnter your Piece or Pawn's destination."); // Prompt

Console.WriteLine("Type 'help' for the key, or 'restart' to go back.\n");

Console.ForegroundColor = ConsoleColor.White;

input = Console.ReadLine(); // Get input

input = input.Replace(" ", string.Empty); // Remove any spaces from input

input = input.ToLower(); // Make input lowercase

Console.ForegroundColor = ConsoleColor.Black;

if (input.Length == 0)

{

input = " ";

}

for (int i = 0; i < letterSelection.Length; i++) // Check all possible letter selections

{

if (input[0] == letterSelection[i]) // Does the selection match?

{

if (input.Length == 1) // Is it just the letter or a letter and number?

{

return input; // Chosen letter

}

if (input.Length == 2)

{

switch (input[1]) // Check that the number ranges 1-8.

{

case '1':

return input;

case '2':

return input;

case '3':

return input;

case '4':

return input;

case '5':

return input;

case '6':

return input;

case '7':

return input;

case '8':

return input;

}

}

}

}

if (input == "back" || input == "restart" || input == "exit") // Exit prompt

{

input = "π";

return input;

}

if (input == "resign" || input == "forfeit") // Resignation

{

input = "π";

return input;

}

if (input == "draw") // Draw

{

input = "π";

return input;

}

if (input == "help" || input == "key") // Help prompt

{

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("\nPlease enter your Piece or Pawn's destination.\nYou can do this by typing the letter coordinate followed by the number coordinate.\n");

Console.WriteLine("The pieces are as follows:");

Console.WriteLine("♟ P = Pawn Moves up one space but captures diagonally forward, promotes upon reaching the top.");

Console.WriteLine("♜ R = Rook Moves up and down or left and right.");

Console.WriteLine("♞ N = Knight Moves in an L pattern and jumps over pieces.");

Console.WriteLine("♝ B = Bishop Moves diagonally.");

Console.WriteLine("♛ Q = Queen Moves any direction.");

Console.WriteLine("♚ K = King Moves any direction one space, can castle with a rook if both haven't moved.");

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\nWhen a King is in check (about to be captured), he must be protected or moved.");

Console.WriteLine("The game ends when a King is in checkmate (cannot be protected that turn).\n");

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("When you are checkmated, enter 'resign'.\n");

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("\nThat is not a valid destination. Please type 'help' for the key or 'restart' to go back.\n");

}

}

}

static int SelectNumber(string input, int[] numberSelection) // Number Selection

{

int output;

while (true) // Repeat forever

{

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("\nEnter your Piece or Pawn's destination number.");// Prompt

Console.WriteLine("Type 'help' for the key, or 'restart' to go back.\n");

Console.ForegroundColor = ConsoleColor.White;

input = Console.ReadLine(); // Get input

input = input.Replace(" ", string.Empty); // Remove any spaces from input

Int32.TryParse(input, out output); // Convert input to integer

Console.ForegroundColor = ConsoleColor.Black;

for (int i = 0; i < numberSelection.Length; i++) // Check all possible number selections

{

if (output == numberSelection[i]) // Does the selection match?

{

return output; // Chosen number

}

}

if (input == "back" || input == "restart" || input == "exit") // Exit prompt

{

output = 9;

return output;

}

if (input == "resign" || input == "forfeit") // Resignation

{

output = 9;

return output;

}

if (input == "draw") // Draw

{

output = 9;

return output;

}

if (input == "help" || input == "key") // Help prompt

{

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("\nPlease enter your Piece or Pawn's destination number coordinate.\nThis is the Y value of the board. You have already entered the X value. Type 'restart' to go back.\n");

Console.WriteLine("The pieces are as follows:");

Console.WriteLine("♟ P = Pawn Moves up one space but captures diagonally forward, promotes upon reaching the top.");

Console.WriteLine("♜ R = Rook Moves up and down or left and right.");

Console.WriteLine("♞ N = Knight Moves in an L pattern and jumps over pieces.");

Console.WriteLine("♝ B = Bishop Moves diagonally.");

Console.WriteLine("♛ Q = Queen Moves any direction.");

Console.WriteLine("♚ K = King Moves any direction one space, can castle with a rook if both haven't moved.");

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("\nWhen a King is in check (about to be captured), he must be protected or moved.");

Console.WriteLine("The game ends when a King is in checkmate (cannot be protected that turn).\n");

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("When you are checkmated, enter 'resign'.\n");

}

else // Bad input

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine("\nThat is not a valid number coordinate. Please type 'help' for the key or 'restart' to go back.\n");

}

}

}

static string SelectSpace(int turn, string spaceSelected, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Space Selected to Piece

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

string input = spaceSelected;

int X = LetterToNumber(spaceSelected[0]);

int Y = spaceSelected[1] - 48;

if (IsEven(turn) == false) // White's turn

{

if ((WhiteKing.X == X) && (WhiteKing.Y == Y))

{

input = input.Remove(0, 2);

return "king" + input;

}

if ((WhiteQueen.X == X) && (WhiteQueen.Y == Y))

{

input = input.Remove(0, 2);

return "queen" + input;

}

if ((WhiteBishop1.X == X) && (WhiteBishop1.Y == Y))

{

input = input.Remove(0, 2);

return "bishop1" + input;

}

if ((WhiteBishop2.X == X) && (WhiteBishop2.Y == Y))

{

input = input.Remove(0, 2);

return "bishop2" + input;

}

if ((WhiteKnight1.X == X) && (WhiteKnight1.Y == Y))

{

input = input.Remove(0, 2);

return "knight1" + input;

}

if ((WhiteKnight2.X == X) && (WhiteKnight2.Y == Y))

{

input = input.Remove(0, 2);

return "knight2" + input;

}

if ((WhiteRook1.X == X) && (WhiteRook1.Y == Y))

{

input = input.Remove(0, 2);

return "rook1" + input;

}

if ((WhiteRook2.X == X) && (WhiteRook2.Y == Y))

{

input = input.Remove(0, 2);

return "rook2" + input;

}

if ((WhitePawn1.X == X) && (WhitePawn1.Y == Y))

{

input = input.Remove(0, 2);

return "pawn1" + input;

}

if ((WhitePawn2.X == X) && (WhitePawn2.Y == Y))

{

input = input.Remove(0, 2);

return "pawn2" + input;

}

if ((WhitePawn3.X == X) && (WhitePawn3.Y == Y))

{

input = input.Remove(0, 2);

return "pawn3" + input;

}

if ((WhitePawn4.X == X) && (WhitePawn4.Y == Y))

{

input = input.Remove(0, 2);

return "pawn4" + input;

}

if ((WhitePawn5.X == X) && (WhitePawn5.Y == Y))

{

input = input.Remove(0, 2);

return "pawn5" + input;

}

if ((WhitePawn6.X == X) && (WhitePawn6.Y == Y))

{

input = input.Remove(0, 2);

return "pawn6" + input;

}

if ((WhitePawn7.X == X) && (WhitePawn7.Y == Y))

{

input = input.Remove(0, 2);

return "pawn7" + input;

}

if ((WhitePawn8.X == X) && (WhitePawn8.Y == Y))

{

input = input.Remove(0, 2);

return "pawn8" + input;

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (WhitePromo[i].X == X && WhitePromo[i].Y == Y)

{

input = input.Remove(0, 2);

return WhitePromo[i].Tag + input;

}

}

}

else // Black's turn

{

if ((BlackKing.X == X) && (BlackKing.Y == Y))

{

input = input.Remove(0, 2);

return "king" + input;

}

if ((BlackQueen.X == X) && (BlackQueen.Y == Y))

{

input = input.Remove(0, 2);

return "queen" + input;

}

if ((BlackBishop1.X == X) && (BlackBishop1.Y == Y))

{

input = input.Remove(0, 2);

return "bishop1" + input;

}

if ((BlackBishop2.X == X) && (BlackBishop2.Y == Y))

{

input = input.Remove(0, 2);

return "bishop2" + input;

}

if ((BlackKnight1.X == X) && (BlackKnight1.Y == Y))

{

input = input.Remove(0, 2);

return "knight1" + input;

}

if ((BlackKnight2.X == X) && (BlackKnight2.Y == Y))

{

input = input.Remove(0, 2);

return "knight2" + input;

}

if ((BlackRook1.X == X) && (BlackRook1.Y == Y))

{

input = input.Remove(0, 2);

return "rook1" + input;

}

if ((BlackRook2.X == X) && (BlackRook2.Y == Y))

{

input = input.Remove(0, 2);

return "rook2" + input;

}

if ((BlackPawn1.X == X) && (BlackPawn1.Y == Y))

{

input = input.Remove(0, 2);

return "pawn1" + input;

}

if ((BlackPawn2.X == X) && (BlackPawn2.Y == Y))

{

input = input.Remove(0, 2);

return "pawn2" + input;

}

if ((BlackPawn3.X == X) && (BlackPawn3.Y == Y))

{

input = input.Remove(0, 2);

return "pawn3" + input;

}

if ((BlackPawn4.X == X) && (BlackPawn4.Y == Y))

{

input = input.Remove(0, 2);

return "pawn4" + input;

}

if ((BlackPawn5.X == X) && (BlackPawn5.Y == Y))

{

input = input.Remove(0, 2);

return "pawn5" + input;

}

if ((BlackPawn6.X == X) && (BlackPawn6.Y == Y))

{

input = input.Remove(0, 2);

return "pawn6" + input;

}

if ((BlackPawn7.X == X) && (BlackPawn7.Y == Y))

{

input = input.Remove(0, 2);

return "pawn7" + input;

}

if ((BlackPawn8.X == X) && (BlackPawn8.Y == Y))

{

input = input.Remove(0, 2);

return "pawn8" + input;

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (BlackPromo[i].X == X && BlackPromo[i].Y == Y)

{

input = input.Remove(0, 2);

return BlackPromo[i].Tag + input;

}

}

}

return input;

}

static string InputInterpretation(string input, string[] pieceSelection) // Interpret Various Inputs

{

if (input == "oo" || input == "o-o" || input == "00" || input == "0-0" || input == "castlek" || input == "castleshort")

{

input = "shortcastle";

return input;

}

if (input == "ooo" || input == "o-o-o" || input == "000" || input == "0-0-0" && input == "castleq" || input == "castlelong")

{

input = "longcastle";

return input;

}

if (input == "ep")

{

input = "enpassant";

return input;

}

if (input == "promote")

{

input = "promotion";

return input;

}

for (int i = 0; i < pieceSelection.Length; i++)

{

if (input == pieceSelection[i])

{

return input;

}

}

if (input == "")

{

input = " ";

return input;

}

if (input.Contains("king1") == true)

{

input = input.Remove(4, 1);

return input;

}

if (input.Contains("k1") == true && input.Contains("rook") == false)

{

input = input.Remove(0, 2);

return "king" + input;

}

if (input.Contains("queen1") == true)

{

input = input.Remove(5, 1);

return input;

}

if (input.Contains("q1") == true)

{

input = input.Remove(0, 2);

return "queen" + input;

}

if (input.Contains("nite") == true)

{

input = input.Remove(0, 4);

input = "knight" + input;

return input;

}

if (input.Contains("night") == true && input.Contains("knight") == false)

{

input = "k" + input;

return input;

}

if (input[0] == 'k' && input.Contains("king") == false && input.Contains("knight") == false)

{

input = input.Remove(0, 1);

input = "king" + input;

return input;

}

if (input[0] == 'q' && input.Contains("queen") == false)

{

input = input.Remove(0, 1);

input = "queen" + input;

return input;

}

if (input[0] == 'b' && input.Contains("bishop") == false)

{

input = input.Remove(0, 1);

input = "bishop" + input;

return input;

}

if (input[0] == 'n' && input.Contains("nite") == false && input.Contains("night") == false)

{

input = input.Remove(0, 1);

input = "knight" + input;

return input;

}

if (input[0] == 'r' && input.Contains("rook") == false)

{

input = input.Remove(0, 1);

input = "rook" + input;

return input;

}

if (input[0] == 'p' && input.Contains("pawn") == false)

{

input = input.Remove(0, 1);

input = "pawn" + input;

return input;

}

return input;

}

static int IsOccupied(int turn, int X, int Y, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Occupied Space Check

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

// TURN INTEGER IS USED FOR COLOR PRIORITY

if (IsEven(turn) == true)

{

if (X == WhiteKing.X) // White King

{

if (Y == WhiteKing.Y)

{

return 1;

}

}

if (X == WhiteQueen.X) // White Queen

{

if (Y == WhiteQueen.Y)

{

return 1;

}

}

if (X == WhiteBishop1.X) // White Bishop1

{

if (Y == WhiteBishop1.Y)

{

return 1;

}

}

if (X == WhiteBishop2.X) // White Bishop2

{

if (Y == WhiteBishop2.Y)

{

return 1;

}

}

if (X == WhiteKnight1.X) // White Knight1

{

if (Y == WhiteKnight1.Y)

{

return 1;

}

}

if (X == WhiteKnight2.X) // White Knight2

{

if (Y == WhiteKnight2.Y)

{

return 1;

}

}

if (X == WhiteRook1.X) // White Rook1

{

if (Y == WhiteRook1.Y)

{

return 1;

}

}

if (X == WhiteRook2.X) // White Rook2

{

if (Y == WhiteRook2.Y)

{

return 1;

}

}

if (X == WhitePawn1.X) // White Pawn1

{

if (Y == WhitePawn1.Y)

{

return 1;

}

}

if (X == WhitePawn2.X) // White Pawn2

{

if (Y == WhitePawn2.Y)

{

return 1;

}

}

if (X == WhitePawn3.X) // White Pawn3

{

if (Y == WhitePawn3.Y)

{

return 1;

}

}

if (X == WhitePawn4.X) // White Pawn4

{

if (Y == WhitePawn4.Y)

{

return 1;

}

}

if (X == WhitePawn5.X) // White Pawn5

{

if (Y == WhitePawn5.Y)

{

return 1;

}

}

if (X == WhitePawn6.X) // White Pawn6

{

if (Y == WhitePawn6.Y)

{

return 1;

}

}

if (X == WhitePawn7.X) // White Pawn7

{

if (Y == WhitePawn7.Y)

{

return 1;

}

}

if (X == WhitePawn8.X) // White Pawn8

{

if (Y == WhitePawn8.Y)

{

return 1;

}

}

if (X == BlackKing.X) // Black King

{

if (Y == BlackKing.Y)

{

return 2;

}

}

if (X == BlackQueen.X) // Black Queen

{

if (Y == BlackQueen.Y)

{

return 2;

}

}

if (X == BlackBishop1.X) // Black Bishop1

{

if (Y == BlackBishop1.Y)

{

return 2;

}

}

if (X == BlackBishop2.X) // Black Bishop2

{

if (Y == BlackBishop2.Y)

{

return 2;

}

}

if (X == BlackKnight1.X) // Black Knight1

{

if (Y == BlackKnight1.Y)

{

return 2;

}

}

if (X == BlackKnight2.X) // Black Knight2

{

if (Y == BlackKnight2.Y)

{

return 2;

}

}

if (X == BlackRook1.X) // Black Rook1

{

if (Y == BlackRook1.Y)

{

return 2;

}

}

if (X == BlackRook2.X) // Black Rook2

{

if (Y == BlackRook2.Y)

{

return 2;

}

}

if (X == BlackPawn1.X) // Black Pawn1

{

if (Y == BlackPawn1.Y)

{

return 2;

}

}

if (X == BlackPawn2.X) // Black Pawn2

{

if (Y == BlackPawn2.Y)

{

return 2;

}

}

if (X == BlackPawn3.X) // Black Pawn3

{

if (Y == BlackPawn3.Y)

{

return 2;

}

}

if (X == BlackPawn4.X) // Black Pawn4

{

if (Y == BlackPawn4.Y)

{

return 2;

}

}

if (X == BlackPawn5.X) // Black Pawn5

{

if (Y == BlackPawn5.Y)

{

return 2;

}

}

if (X == BlackPawn6.X) // Black Pawn6

{

if (Y == BlackPawn6.Y)

{

return 2;

}

}

if (X == BlackPawn7.X) // Black Pawn7

{

if (Y == BlackPawn7.Y)

{

return 2;

}

}

if (X == BlackPawn8.X) // Black Pawn8

{

if (Y == BlackPawn8.Y)

{

return 2;

}

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (WhitePromo[i].Type != "none")

{

if (X == WhitePromo[i].X)

{

if (Y == WhitePromo[i].Y)

{

return 1;

}

}

}

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (BlackPromo[i].Type != "none")

{

if (X == BlackPromo[i].X)

{

if (Y == BlackPromo[i].Y)

{

return 2;

}

}

}

}

}

else

{

if (X == BlackKing.X) // Black King

{

if (Y == BlackKing.Y)

{

return 2;

}

}

if (X == BlackQueen.X) // Black Queen

{

if (Y == BlackQueen.Y)

{

return 2;

}

}

if (X == BlackBishop1.X) // Black Bishop1

{

if (Y == BlackBishop1.Y)

{

return 2;

}

}

if (X == BlackBishop2.X) // Black Bishop2

{

if (Y == BlackBishop2.Y)

{

return 2;

}

}

if (X == BlackKnight1.X) // Black Knight1

{

if (Y == BlackKnight1.Y)

{

return 2;

}

}

if (X == BlackKnight2.X) // Black Knight2

{

if (Y == BlackKnight2.Y)

{

return 2;

}

}

if (X == BlackRook1.X) // Black Rook1

{

if (Y == BlackRook1.Y)

{

return 2;

}

}

if (X == BlackRook2.X) // Black Rook2

{

if (Y == BlackRook2.Y)

{

return 2;

}

}

if (X == BlackPawn1.X) // Black Pawn1

{

if (Y == BlackPawn1.Y)

{

return 2;

}

}

if (X == BlackPawn2.X) // Black Pawn2

{

if (Y == BlackPawn2.Y)

{

return 2;

}

}

if (X == BlackPawn3.X) // Black Pawn3

{

if (Y == BlackPawn3.Y)

{

return 2;

}

}

if (X == BlackPawn4.X) // Black Pawn4

{

if (Y == BlackPawn4.Y)

{

return 2;

}

}

if (X == BlackPawn5.X) // Black Pawn5

{

if (Y == BlackPawn5.Y)

{

return 2;

}

}

if (X == BlackPawn6.X) // Black Pawn6

{

if (Y == BlackPawn6.Y)

{

return 2;

}

}

if (X == BlackPawn7.X) // Black Pawn7

{

if (Y == BlackPawn7.Y)

{

return 2;

}

}

if (X == BlackPawn8.X) // Black Pawn8

{

if (Y == BlackPawn8.Y)

{

return 2;

}

}

if (X == WhiteKing.X) // White King

{

if (Y == WhiteKing.Y)

{

return 1;

}

}

if (X == WhiteQueen.X) // White Queen

{

if (Y == WhiteQueen.Y)

{

return 1;

}

}

if (X == WhiteBishop1.X) // White Bishop1

{

if (Y == WhiteBishop1.Y)

{

return 1;

}

}

if (X == WhiteBishop2.X) // White Bishop2

{

if (Y == WhiteBishop2.Y)

{

return 1;

}

}

if (X == WhiteKnight1.X) // White Knight1

{

if (Y == WhiteKnight1.Y)

{

return 1;

}

}

if (X == WhiteKnight2.X) // White Knight2

{

if (Y == WhiteKnight2.Y)

{

return 1;

}

}

if (X == WhiteRook1.X) // White Rook1

{

if (Y == WhiteRook1.Y)

{

return 1;

}

}

if (X == WhiteRook2.X) // White Rook2

{

if (Y == WhiteRook2.Y)

{

return 1;

}

}

if (X == WhitePawn1.X) // White Pawn1

{

if (Y == WhitePawn1.Y)

{

return 1;

}

}

if (X == WhitePawn2.X) // White Pawn2

{

if (Y == WhitePawn2.Y)

{

return 1;

}

}

if (X == WhitePawn3.X) // White Pawn3

{

if (Y == WhitePawn3.Y)

{

return 1;

}

}

if (X == WhitePawn4.X) // White Pawn4

{

if (Y == WhitePawn4.Y)

{

return 1;

}

}

if (X == WhitePawn5.X) // White Pawn5

{

if (Y == WhitePawn5.Y)

{

return 1;

}

}

if (X == WhitePawn6.X) // White Pawn6

{

if (Y == WhitePawn6.Y)

{

return 1;

}

}

if (X == WhitePawn7.X) // White Pawn7

{

if (Y == WhitePawn7.Y)

{

return 1;

}

}

if (X == WhitePawn8.X) // White Pawn8

{

if (Y == WhitePawn8.Y)

{

return 1;

}

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (BlackPromo[i].Type != "none")

{

if (X == BlackPromo[i].X)

{

if (Y == BlackPromo[i].Y)

{

return 2;

}

}

}

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (WhitePromo[i].Type != "none")

{

if (X == WhitePromo[i].X)

{

if (Y == WhitePromo[i].Y)

{

return 1;

}

}

}

}

}

return 0;

}

static string CapturePiece(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Capturing a Piece

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (destinationX == BlackQueen.X) // Black Queen is captured

{

if (destinationY == BlackQueen.Y)

{

BlackQueen.X = 0;

BlackQueen.Y = 0;

BlackQueen.Captured = true;

return " captures queen";

}

}

if (destinationX == BlackBishop1.X) // Black Bishop1 is captured

{

if (destinationY == BlackBishop1.Y)

{

BlackBishop1.X = 0;

BlackBishop1.Y = 0;

BlackBishop1.Captured = true;

return " captures bishop1";

}

}

if (destinationX == BlackBishop2.X) // Black Bishop2 is captured

{

if (destinationY == BlackBishop2.Y)

{

BlackBishop2.X = 0;

BlackBishop2.Y = 0;

BlackBishop2.Captured = true;

return " captures bishop2";

}

}

if (destinationX == BlackKnight1.X) // Black Knight1 is captured

{

if (destinationY == BlackKnight1.Y)

{

BlackKnight1.X = 0;

BlackKnight1.Y = 0;

BlackKnight1.Captured = true;

return " captures knight1";

}

}

if (destinationX == BlackKnight2.X) // Black Knight2 is captured

{

if (destinationY == BlackKnight2.Y)

{

BlackKnight2.X = 0;

BlackKnight2.Y = 0;

BlackKnight2.Captured = true;

return " captures knight2";

}

}

if (destinationX == BlackRook1.X) // Black Rook1 is captured

{

if (destinationY == BlackRook1.Y)

{

BlackRook1.X = 0;

BlackRook1.Y = 0;

BlackRook1.Captured = true;

return " captures rook1";

}

}

if (destinationX == BlackRook2.X) // Black Rook2 is captured

{

if (destinationY == BlackRook2.Y)

{

BlackRook2.X = 0;

BlackRook2.Y = 0;

BlackRook2.Captured = true;

return " captures rook2";

}

}

if (destinationX == BlackPawn1.X) // Black Pawn1 is captured

{

if (destinationY == BlackPawn1.Y)

{

BlackPawn1.X = 0;

BlackPawn1.Y = 0;

BlackPawn1.Captured = true;

return " captures pawn1";

}

}

if (destinationX == BlackPawn2.X) // Black Pawn2 is captured

{

if (destinationY == BlackPawn2.Y)

{

BlackPawn2.X = 0;

BlackPawn2.Y = 0;

BlackPawn2.Captured = true;

return " captures pawn2";

}

}

if (destinationX == BlackPawn3.X) // Black Pawn3 is captured

{

if (destinationY == BlackPawn3.Y)

{

BlackPawn3.X = 0;

BlackPawn3.Y = 0;

BlackPawn3.Captured = true;

return " captures pawn3";

}

}

if (destinationX == BlackPawn4.X) // Black Pawn4 is captured

{

if (destinationY == BlackPawn4.Y)

{

BlackPawn4.X = 0;

BlackPawn4.Y = 0;

BlackPawn4.Captured = true;

return " captures pawn4";

}

}

if (destinationX == BlackPawn5.X) // Black Pawn5 is captured

{

if (destinationY == BlackPawn5.Y)

{

BlackPawn5.X = 0;

BlackPawn5.Y = 0;

BlackPawn5.Captured = true;

return " captures pawn5";

}

}

if (destinationX == BlackPawn6.X) // Black Pawn6 is captured

{

if (destinationY == BlackPawn6.Y)

{

BlackPawn6.X = 0;

BlackPawn6.Y = 0;

BlackPawn6.Captured = true;

return " captures pawn6";

}

}

if (destinationX == BlackPawn7.X) // Black Pawn7 is captured

{

if (destinationY == BlackPawn7.Y)

{

BlackPawn7.X = 0;

BlackPawn7.Y = 0;

BlackPawn7.Captured = true;

return " captures pawn7";

}

}

if (destinationX == BlackPawn8.X) // Black Pawn8 is captured

{

if (destinationY == BlackPawn8.Y)

{

BlackPawn8.X = 0;

BlackPawn8.Y = 0;

BlackPawn8.Captured = true;

return " captures pawn8";

}

}

for (int i = 0; i < BlackPromo.Length; i++) // Black Promoted Piece is captured

{

if (destinationX == BlackPromo[i].X)

{

if (destinationY == BlackPromo[i].Y)

{

BlackPromo[i].X = 0;

BlackPromo[i].Y = 0;

BlackPromo[i].Captured = true;

return " captures " + BlackPromo[i].Tag;

}

}

}

}

else // Black's Turn

{

if (destinationX == WhiteQueen.X) // White Queen is captured

{

if (destinationY == WhiteQueen.Y)

{

WhiteQueen.X = 0;

WhiteQueen.Y = 0;

WhiteQueen.Captured = true;

return " captures queen";

}

}

if (destinationX == WhiteBishop1.X) // White Bishop1 is captured

{

if (destinationY == WhiteBishop1.Y)

{

WhiteBishop1.X = 0;

WhiteBishop1.Y = 0;

WhiteBishop1.Captured = true;

return " captures bishop1";

}

}

if (destinationX == WhiteBishop2.X) // White Bishop2 is captured

{

if (destinationY == WhiteBishop2.Y)

{

WhiteBishop2.X = 0;

WhiteBishop2.Y = 0;

WhiteBishop2.Captured = true;

return " captures bishop2";

}

}

if (destinationX == WhiteKnight1.X) // White Knight1 is captured

{

if (destinationY == WhiteKnight1.Y)

{

WhiteKnight1.X = 0;

WhiteKnight1.Y = 0;

WhiteKnight1.Captured = true;

return " captures knight1";

}

}

if (destinationX == WhiteKnight2.X) // White Knight2 is captured

{

if (destinationY == WhiteKnight2.Y)

{

WhiteKnight2.X = 0;

WhiteKnight2.Y = 0;

WhiteKnight2.Captured = true;

return " captures knight2";

}

}

if (destinationX == WhiteRook1.X) // White Rook1 is captured

{

if (destinationY == WhiteRook1.Y)

{

WhiteRook1.X = 0;

WhiteRook1.Y = 0;

WhiteRook1.Captured = true;

return " captures rook1";

}

}

if (destinationX == WhiteRook2.X) // White Rook2 is captured

{

if (destinationY == WhiteRook2.Y)

{

WhiteRook2.X = 0;

WhiteRook2.Y = 0;

WhiteRook2.Captured = true;

return " captures rook2";

}

}

if (destinationX == WhitePawn1.X) // White Pawn1 is captured

{

if (destinationY == WhitePawn1.Y)

{

WhitePawn1.X = 0;

WhitePawn1.Y = 0;

WhitePawn1.Captured = true;

return " captures pawn1";

}

}

if (destinationX == WhitePawn2.X) // White Pawn2 is captured

{

if (destinationY == WhitePawn2.Y)

{

WhitePawn2.X = 0;

WhitePawn2.Y = 0;

WhitePawn2.Captured = true;

return " captures pawn2";

}

}

if (destinationX == WhitePawn3.X) // White Pawn3 is captured

{

if (destinationY == WhitePawn3.Y)

{

WhitePawn3.X = 0;

WhitePawn3.Y = 0;

WhitePawn3.Captured = true;

return " captures pawn3";

}

}

if (destinationX == WhitePawn4.X) // White Pawn4 is captured

{

if (destinationY == WhitePawn4.Y)

{

WhitePawn4.X = 0;

WhitePawn4.Y = 0;

WhitePawn4.Captured = true;

return " captures pawn4";

}

}

if (destinationX == WhitePawn5.X) // White Pawn5 is captured

{

if (destinationY == WhitePawn5.Y)

{

WhitePawn5.X = 0;

WhitePawn5.Y = 0;

WhitePawn5.Captured = true;

return " captures pawn5";

}

}

if (destinationX == WhitePawn6.X) // White Pawn6 is captured

{

if (destinationY == WhitePawn6.Y)

{

WhitePawn6.X = 0;

WhitePawn6.Y = 0;

WhitePawn6.Captured = true;

return " captures pawn6";

}

}

if (destinationX == WhitePawn7.X) // White Pawn7 is captured

{

if (destinationY == WhitePawn7.Y)

{

WhitePawn7.X = 0;

WhitePawn7.Y = 0;

WhitePawn7.Captured = true;

return " captures pawn7";

}

}

if (destinationX == WhitePawn8.X) // White Pawn8 is captured

{

if (destinationY == WhitePawn8.Y)

{

WhitePawn8.X = 0;

WhitePawn8.Y = 0;

WhitePawn8.Captured = true;

return " captures pawn8";

}

}

for (int i = 0; i < WhitePromo.Length; i++) // White Promoted Piece is captured

{

if (destinationX == WhitePromo[i].X)

{

if (destinationY == WhitePromo[i].Y)

{

WhitePromo[i].X = 0;

WhitePromo[i].Y = 0;

WhitePromo[i].Captured = true;

return " captures " + WhitePromo[i].Tag;

}

}

}

}

return "";

}

static bool IsCaptured(int turn, string piece, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Piece Captured Check

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's turn

{

if (piece == "queen")

{

if (WhiteQueen.Captured == true)

{

return true;

}

}

if (piece == "bishop1")

{

if (WhiteBishop1.Captured == true)

{

return true;

}

}

if (piece == "bishop2")

{

if (WhiteBishop2.Captured == true)

{

return true;

}

}

if (piece == "knight1")

{

if (WhiteKnight1.Captured == true)

{

return true;

}

}

if (piece == "knight2")

{

if (WhiteKnight2.Captured == true)

{

return true;

}

}

if (piece == "rook1")

{

if (WhiteRook1.Captured == true)

{

return true;

}

}

if (piece == "rook2")

{

if (WhiteRook2.Captured == true)

{

return true;

}

}

if (piece == "pawn1")

{

if (WhitePawn1.Captured == true)

{

return true;

}

}

if (piece == "pawn2")

{

if (WhitePawn2.Captured == true)

{

return true;

}

}

if (piece == "pawn3")

{

if (WhitePawn3.Captured == true)

{

return true;

}

}

if (piece == "pawn4")

{

if (WhitePawn4.Captured == true)

{

return true;

}

}

if (piece == "pawn5")

{

if (WhitePawn5.Captured == true)

{

return true;

}

}

if (piece == "pawn6")

{

if (WhitePawn6.Captured == true)

{

return true;

}

}

if (piece == "pawn7")

{

if (WhitePawn7.Captured == true)

{

return true;

}

}

if (piece == "pawn8")

{

if (WhitePawn8.Captured == true)

{

return true;

}

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (piece == WhitePromo[i].Tag)

{

if (WhitePromo[i].Captured == true)

{

return true;

}

}

}

}

else // Black's turn

{

if (piece == "queen")

{

if (BlackQueen.Captured == true)

{

return true;

}

}

if (piece == "bishop1")

{

if (BlackBishop1.Captured == true)

{

return true;

}

}

if (piece == "bishop2")

{

if (BlackBishop2.Captured == true)

{

return true;

}

}

if (piece == "knight1")

{

if (BlackKnight1.Captured == true)

{

return true;

}

}

if (piece == "knight2")

{

if (BlackKnight2.Captured == true)

{

return true;

}

}

if (piece == "rook1")

{

if (BlackRook1.Captured == true)

{

return true;

}

}

if (piece == "rook2")

{

if (BlackRook2.Captured == true)

{

return true;

}

}

if (piece == "pawn1")

{

if (BlackPawn1.Captured == true)

{

return true;

}

}

if (piece == "pawn2")

{

if (BlackPawn2.Captured == true)

{

return true;

}

}

if (piece == "pawn3")

{

if (BlackPawn3.Captured == true)

{

return true;

}

}

if (piece == "pawn4")

{

if (BlackPawn4.Captured == true)

{

return true;

}

}

if (piece == "pawn5")

{

if (BlackPawn5.Captured == true)

{

return true;

}

}

if (piece == "pawn6")

{

if (BlackPawn6.Captured == true)

{

return true;

}

}

if (piece == "pawn7")

{

if (BlackPawn7.Captured == true)

{

return true;

}

}

if (piece == "pawn8")

{

if (BlackPawn8.Captured == true)

{

return true;

}

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (piece == BlackPromo[i].Tag)

{

if (BlackPromo[i].Captured == true)

{

return true;

}

}

}

}

return false;

}

// Movement Methods

static bool KingMove(int destinationX, int destinationY, int turn, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // King Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if ((destinationX == WhiteKing.X + 1) || (destinationX == WhiteKing.X - 1))

{

if (destinationY == WhiteKing.Y)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Right or Left one

}

}

if ((destinationY == WhiteKing.Y + 1) || (destinationY == WhiteKing.Y - 1))

{

if (destinationX == WhiteKing.X)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Up or Down one

}

}

if (destinationX != WhiteKing.X)

{

if (destinationY != WhiteKing.Y)

{

if (destinationX == WhiteKing.X + 1)

{

if (destinationY == WhiteKing.Y + 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Diagonal Right Up one

}

}

if (destinationX == WhiteKing.X + 1)

{

if (destinationY == WhiteKing.Y - 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Diagonal Right Down one

}

}

if (destinationX == WhiteKing.X - 1)

{

if (destinationY == WhiteKing.Y + 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Diagonal Left Up one

}

}

if (destinationX == WhiteKing.X - 1)

{

if (destinationY == WhiteKing.Y - 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

return true; // Diagonal Left Down one

}

}

}

}

}

else // Black's Turn

{

if ((destinationX == BlackKing.X + 1) || (destinationX == BlackKing.X - 1))

{

if (destinationY == BlackKing.Y)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Right or Left one

}

}

if ((destinationY == BlackKing.Y + 1) || (destinationY == BlackKing.Y - 1))

{

if (destinationX == BlackKing.X)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Up or Down one

}

}

if (destinationX != BlackKing.X)

{

if (destinationY != BlackKing.Y)

{

if (destinationX == BlackKing.X + 1)

{

if (destinationY == BlackKing.Y + 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Diagonal Right Up one

}

}

if (destinationX == BlackKing.X + 1)

{

if (destinationY == BlackKing.Y - 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Diagonal Right Down one

}

}

if (destinationX == BlackKing.X - 1)

{

if (destinationY == BlackKing.Y + 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Diagonal Left Up one

}

}

if (destinationX == BlackKing.X - 1)

{

if (destinationY == BlackKing.Y - 1)

{

if (IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

return true; // Diagonal Left Down one

}

}

}

}

}

return false;

}

static bool QueenMove(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Queen Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteQueen.X, WhiteQueen.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteQueen.Captured == false)

{

if (destinationX > WhiteQueen.X && destinationY == WhiteQueen.Y)

{

for (int currentX = WhiteQueen.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < WhiteQueen.X && destinationY == WhiteQueen.Y)

{

for (int currentX = WhiteQueen.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > WhiteQueen.Y && destinationX == WhiteQueen.X)

{

for (int currentY = WhiteQueen.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < WhiteQueen.Y && destinationX == WhiteQueen.X)

{

for (int currentY = WhiteQueen.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

int distanceX = Math.Abs(destinationX - WhiteQueen.X);

int distanceY = Math.Abs(destinationY - WhiteQueen.Y);

if (WhiteQueen.X != destinationX && WhiteQueen.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > WhiteQueen.X)

{

if (destinationY > WhiteQueen.Y)

{

for (int currentX = WhiteQueen.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteQueen.Y + Math.Abs(currentX - WhiteQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = WhiteQueen.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteQueen.Y - Math.Abs(currentX - WhiteQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > WhiteQueen.Y)

{

for (int currentX = WhiteQueen.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteQueen.Y + Math.Abs(currentX - WhiteQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = WhiteQueen.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteQueen.Y - Math.Abs(currentX - WhiteQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackQueen.X, BlackQueen.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackQueen.Captured == false)

{

if (destinationX > BlackQueen.X && destinationY == BlackQueen.Y)

{

for (int currentX = BlackQueen.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < BlackQueen.X && destinationY == BlackQueen.Y)

{

for (int currentX = BlackQueen.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > BlackQueen.Y && destinationX == BlackQueen.X)

{

for (int currentY = BlackQueen.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < BlackQueen.Y && destinationX == BlackQueen.X)

{

for (int currentY = BlackQueen.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

int distanceX = Math.Abs(destinationX - BlackQueen.X);

int distanceY = Math.Abs(destinationY - BlackQueen.Y);

if (BlackQueen.X != destinationX && BlackQueen.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > BlackQueen.X)

{

if (destinationY > BlackQueen.Y)

{

for (int currentX = BlackQueen.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackQueen.Y + Math.Abs(currentX - BlackQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = BlackQueen.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackQueen.Y - Math.Abs(currentX - BlackQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > BlackQueen.Y)

{

for (int currentX = BlackQueen.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackQueen.Y + Math.Abs(currentX - BlackQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = BlackQueen.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackQueen.Y - Math.Abs(currentX - BlackQueen.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

return false;

}

static bool BishopMove1(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Bishop Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteBishop1.X, WhiteBishop1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteBishop1.Captured == false)

{

int distanceX = Math.Abs(destinationX - WhiteBishop1.X);

int distanceY = Math.Abs(destinationY - WhiteBishop1.Y);

if (WhiteBishop1.X != destinationX && WhiteBishop1.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > WhiteBishop1.X)

{

if (destinationY > WhiteBishop1.Y)

{

for (int currentX = WhiteBishop1.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteBishop1.Y + Math.Abs(currentX - WhiteBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = WhiteBishop1.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteBishop1.Y - Math.Abs(currentX - WhiteBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > WhiteBishop1.Y)

{

for (int currentX = WhiteBishop1.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteBishop1.Y + Math.Abs(currentX - WhiteBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = WhiteBishop1.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteBishop1.Y - Math.Abs(currentX - WhiteBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackBishop1.X, BlackBishop1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackBishop1.Captured == false)

{

int distanceX = Math.Abs(destinationX - BlackBishop1.X);

int distanceY = Math.Abs(destinationY - BlackBishop1.Y);

if (BlackBishop1.X != destinationX && BlackBishop1.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > BlackBishop1.X)

{

if (destinationY > BlackBishop1.Y)

{

for (int currentX = BlackBishop1.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackBishop1.Y + Math.Abs(currentX - BlackBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = BlackBishop1.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackBishop1.Y - Math.Abs(currentX - BlackBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > BlackBishop1.Y)

{

for (int currentX = BlackBishop1.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackBishop1.Y + Math.Abs(currentX - BlackBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = BlackBishop1.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackBishop1.Y - Math.Abs(currentX - BlackBishop1.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

return false;

}

static bool BishopMove2(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Bishop Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteBishop2.X, WhiteBishop2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteBishop2.Captured == false)

{

int distanceX = Math.Abs(destinationX - WhiteBishop2.X);

int distanceY = Math.Abs(destinationY - WhiteBishop2.Y);

if (WhiteBishop2.X != destinationX && WhiteBishop2.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > WhiteBishop2.X)

{

if (destinationY > WhiteBishop2.Y)

{

for (int currentX = WhiteBishop2.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteBishop2.Y + Math.Abs(currentX - WhiteBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = WhiteBishop2.X + 1; currentX < destinationX; currentX++)

{

int currentY = WhiteBishop2.Y - Math.Abs(currentX - WhiteBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > WhiteBishop2.Y)

{

for (int currentX = WhiteBishop2.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteBishop2.Y + Math.Abs(currentX - WhiteBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = WhiteBishop2.X - 1; currentX > destinationX; currentX--)

{

int currentY = WhiteBishop2.Y - Math.Abs(currentX - WhiteBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackBishop2.X, BlackBishop2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackBishop2.Captured == false)

{

int distanceX = Math.Abs(destinationX - BlackBishop2.X);

int distanceY = Math.Abs(destinationY - BlackBishop2.Y);

if (BlackBishop2.X != destinationX && BlackBishop2.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > BlackBishop2.X)

{

if (destinationY > BlackBishop2.Y)

{

for (int currentX = BlackBishop2.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackBishop2.Y + Math.Abs(currentX - BlackBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = BlackBishop2.X + 1; currentX < destinationX; currentX++)

{

int currentY = BlackBishop2.Y - Math.Abs(currentX - BlackBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > BlackBishop2.Y)

{

for (int currentX = BlackBishop2.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackBishop2.Y + Math.Abs(currentX - BlackBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = BlackBishop2.X - 1; currentX > destinationX; currentX--)

{

int currentY = BlackBishop2.Y - Math.Abs(currentX - BlackBishop2.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

return false;

}

static bool KnightMove1(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Knight Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteKnight1.X, WhiteKnight1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteKnight1.Captured == false)

{

if ((destinationX == WhiteKnight1.X + 2) || (destinationX == WhiteKnight1.X - 2))

{

if ((destinationY == WhiteKnight1.Y + 1) || (destinationY == WhiteKnight1.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == WhiteKnight1.X + 1) || (destinationX == WhiteKnight1.X - 1))

{

if ((destinationY == WhiteKnight1.Y + 2) || (destinationY == WhiteKnight1.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == WhiteKnight1.Y + 2) || (destinationY == WhiteKnight1.Y - 2))

{

if ((destinationX == WhiteKnight1.X + 1) || (destinationX == WhiteKnight1.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == WhiteKnight1.Y + 1) || (destinationY == WhiteKnight1.Y - 1))

{

if ((destinationX == WhiteKnight1.X + 2) || (destinationX == WhiteKnight1.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackKnight1.X, BlackKnight1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackKnight1.Captured == false)

{

if ((destinationX == BlackKnight1.X + 2) || (destinationX == BlackKnight1.X - 2))

{

if ((destinationY == BlackKnight1.Y + 1) || (destinationY == BlackKnight1.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == BlackKnight1.X + 1) || (destinationX == BlackKnight1.X - 1))

{

if ((destinationY == BlackKnight1.Y + 2) || (destinationY == BlackKnight1.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == BlackKnight1.Y + 2) || (destinationY == BlackKnight1.Y - 2))

{

if ((destinationX == BlackKnight1.X + 1) || (destinationX == BlackKnight1.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == BlackKnight1.Y + 1) || (destinationY == BlackKnight1.Y - 1))

{

if ((destinationX == BlackKnight1.X + 2) || (destinationX == BlackKnight1.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

return false;

}

static bool KnightMove2(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Knight Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteKnight2.X, WhiteKnight2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteKnight2.Captured == false)

{

if ((destinationX == WhiteKnight2.X + 2) || (destinationX == WhiteKnight2.X - 2))

{

if ((destinationY == WhiteKnight2.Y + 1) || (destinationY == WhiteKnight2.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == WhiteKnight2.X + 1) || (destinationX == WhiteKnight2.X - 1))

{

if ((destinationY == WhiteKnight2.Y + 2) || (destinationY == WhiteKnight2.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == WhiteKnight2.Y + 2) || (destinationY == WhiteKnight2.Y - 2))

{

if ((destinationX == WhiteKnight2.X + 1) || (destinationX == WhiteKnight2.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == WhiteKnight2.Y + 1) || (destinationY == WhiteKnight2.Y - 1))

{

if ((destinationX == WhiteKnight2.X + 2) || (destinationX == WhiteKnight2.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackKnight2.X, BlackKnight2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackKnight2.Captured == false)

{

if ((destinationX == BlackKnight2.X + 2) || (destinationX == BlackKnight2.X - 2))

{

if ((destinationY == BlackKnight2.Y + 1) || (destinationY == BlackKnight2.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == BlackKnight2.X + 1) || (destinationX == BlackKnight2.X - 1))

{

if ((destinationY == BlackKnight2.Y + 2) || (destinationY == BlackKnight2.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == BlackKnight2.Y + 2) || (destinationY == BlackKnight2.Y - 2))

{

if ((destinationX == BlackKnight2.X + 1) || (destinationX == BlackKnight2.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == BlackKnight2.Y + 1) || (destinationY == BlackKnight2.Y - 1))

{

if ((destinationX == BlackKnight2.X + 2) || (destinationX == BlackKnight2.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

return false;

}

static bool RookMove1(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Rook Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteRook1.X, WhiteRook1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteRook1.Captured == false)

{

if (destinationX > WhiteRook1.X && destinationY == WhiteRook1.Y)

{

for (int currentX = WhiteRook1.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < WhiteRook1.X && destinationY == WhiteRook1.Y)

{

for (int currentX = WhiteRook1.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > WhiteRook1.Y && destinationX == WhiteRook1.X)

{

for (int currentY = WhiteRook1.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < WhiteRook1.Y && destinationX == WhiteRook1.X)

{

for (int currentY = WhiteRook1.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackRook1.X, BlackRook1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackRook1.Captured == false)

{

if (destinationX > BlackRook1.X && destinationY == BlackRook1.Y)

{

for (int currentX = BlackRook1.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < BlackRook1.X && destinationY == BlackRook1.Y)

{

for (int currentX = BlackRook1.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > BlackRook1.Y && destinationX == BlackRook1.X)

{

for (int currentY = BlackRook1.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < BlackRook1.Y && destinationX == BlackRook1.X)

{

for (int currentY = BlackRook1.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

return false;

}

static bool RookMove2(int destinationX, int destinationY, int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Rook Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhiteRook2.X, WhiteRook2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhiteRook2.Captured == false)

{

if (destinationX > WhiteRook2.X && destinationY == WhiteRook2.Y)

{

for (int currentX = WhiteRook2.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < WhiteRook2.X && destinationY == WhiteRook2.Y)

{

for (int currentX = WhiteRook2.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > WhiteRook2.Y && destinationX == WhiteRook2.X)

{

for (int currentY = WhiteRook2.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < WhiteRook2.Y && destinationX == WhiteRook2.X)

{

for (int currentY = WhiteRook2.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackRook2.X, BlackRook2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackRook2.Captured == false)

{

if (destinationX > BlackRook2.X && destinationY == BlackRook2.Y)

{

for (int currentX = BlackRook2.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < BlackRook2.X && destinationY == BlackRook2.Y)

{

for (int currentX = BlackRook2.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > BlackRook2.Y && destinationX == BlackRook2.X)

{

for (int currentY = BlackRook2.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < BlackRook2.Y && destinationX == BlackRook2.X)

{

for (int currentY = BlackRook2.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

return false;

}

static bool PawnMove1(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn1.X, WhitePawn1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn1.Captured == false)

{

if (destinationY == WhitePawn1.Y + 1 && destinationX == WhitePawn1.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn1.Y + 2 && destinationX == WhitePawn1.X && WhitePawn1.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn1.Y + 1 && destinationX == WhitePawn1.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn1.Y + 1 && destinationX == WhitePawn1.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn1.Y + 1 && destinationX == WhitePawn1.X + 1)

{

if (EnPassant(WhitePawn1.X, WhitePawn1.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn1.Y + 1 && destinationX == WhitePawn1.X - 1)

{

if (EnPassant(WhitePawn1.X, WhitePawn1.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn1.X, BlackPawn1.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn1.Captured == false)

{

if (destinationY == BlackPawn1.Y - 1 && destinationX == BlackPawn1.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn1.Y - 2 && destinationX == BlackPawn1.X && BlackPawn1.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn1.Y - 1 && destinationX == BlackPawn1.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn1.Y - 1 && destinationX == BlackPawn1.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn1.Y - 1 && destinationX == BlackPawn1.X + 1)

{

if (EnPassant(BlackPawn1.X, BlackPawn1.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn1.Y - 1 && destinationX == BlackPawn1.X - 1)

{

if (EnPassant(BlackPawn1.X, BlackPawn1.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove2(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn2.X, WhitePawn2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn2.Captured == false)

{

if (destinationY == WhitePawn2.Y + 1 && destinationX == WhitePawn2.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn2.Y + 2 && destinationX == WhitePawn2.X && WhitePawn2.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn2.Y + 1 && destinationX == WhitePawn2.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn2.Y + 1 && destinationX == WhitePawn2.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn2.Y + 1 && destinationX == WhitePawn2.X + 1)

{

if (EnPassant(WhitePawn2.X, WhitePawn2.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn2.Y + 1 && destinationX == WhitePawn2.X - 1)

{

if (EnPassant(WhitePawn2.X, WhitePawn2.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn2.X, BlackPawn2.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn2.Captured == false)

{

if (destinationY == BlackPawn2.Y - 1 && destinationX == BlackPawn2.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn2.Y - 2 && destinationX == BlackPawn2.X && BlackPawn2.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn2.Y - 1 && destinationX == BlackPawn2.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn2.Y - 1 && destinationX == BlackPawn2.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn2.Y - 1 && destinationX == BlackPawn2.X + 1)

{

if (EnPassant(BlackPawn2.X, BlackPawn2.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn2.Y - 1 && destinationX == BlackPawn2.X - 1)

{

if (EnPassant(BlackPawn2.X, BlackPawn2.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove3(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn3.X, WhitePawn3.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn3.Captured == false)

{

if (destinationY == WhitePawn3.Y + 1 && destinationX == WhitePawn3.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn3.Y + 2 && destinationX == WhitePawn3.X && WhitePawn3.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn3.Y + 1 && destinationX == WhitePawn3.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn3.Y + 1 && destinationX == WhitePawn3.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn3.Y + 1 && destinationX == WhitePawn3.X + 1)

{

if (EnPassant(WhitePawn3.X, WhitePawn3.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn3.Y + 1 && destinationX == WhitePawn3.X - 1)

{

if (EnPassant(WhitePawn3.X, WhitePawn3.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn3.X, BlackPawn3.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn3.Captured == false)

{

if (destinationY == BlackPawn3.Y - 1 && destinationX == BlackPawn3.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn3.Y - 2 && destinationX == BlackPawn3.X && BlackPawn3.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn3.Y - 1 && destinationX == BlackPawn3.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn3.Y - 1 && destinationX == BlackPawn3.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn3.Y - 1 && destinationX == BlackPawn3.X + 1)

{

if (EnPassant(BlackPawn3.X, BlackPawn3.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn3.Y - 1 && destinationX == BlackPawn3.X - 1)

{

if (EnPassant(BlackPawn3.X, BlackPawn3.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove4(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn4.X, WhitePawn4.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn4.Captured == false)

{

if (destinationY == WhitePawn4.Y + 1 && destinationX == WhitePawn4.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn4.Y + 2 && destinationX == WhitePawn4.X && WhitePawn4.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn4.Y + 1 && destinationX == WhitePawn4.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn4.Y + 1 && destinationX == WhitePawn4.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn4.Y + 1 && destinationX == WhitePawn4.X + 1)

{

if (EnPassant(WhitePawn4.X, WhitePawn4.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn4.Y + 1 && destinationX == WhitePawn4.X - 1)

{

if (EnPassant(WhitePawn4.X, WhitePawn4.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn4.X, BlackPawn4.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn4.Captured == false)

{

if (destinationY == BlackPawn4.Y - 1 && destinationX == BlackPawn4.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn4.Y - 2 && destinationX == BlackPawn4.X && BlackPawn4.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn4.Y - 1 && destinationX == BlackPawn4.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn4.Y - 1 && destinationX == BlackPawn4.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn4.Y - 1 && destinationX == BlackPawn4.X + 1)

{

if (EnPassant(BlackPawn4.X, BlackPawn4.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn4.Y - 1 && destinationX == BlackPawn4.X - 1)

{

if (EnPassant(BlackPawn4.X, BlackPawn4.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove5(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsOccupied(turn, WhitePawn5.X, WhitePawn5.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (IsEven(turn) == false) // White's Turn

{

if (WhitePawn5.Captured == false)

{

if (destinationY == WhitePawn5.Y + 1 && destinationX == WhitePawn5.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn5.Y + 2 && destinationX == WhitePawn5.X && WhitePawn5.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn5.Y + 1 && destinationX == WhitePawn5.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn5.Y + 1 && destinationX == WhitePawn5.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn5.Y + 1 && destinationX == WhitePawn5.X + 1)

{

if (EnPassant(WhitePawn5.X, WhitePawn5.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn5.Y + 1 && destinationX == WhitePawn5.X - 1)

{

if (EnPassant(WhitePawn5.X, WhitePawn5.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn5.X, BlackPawn5.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn5.Captured == false)

{

if (destinationY == BlackPawn5.Y - 1 && destinationX == BlackPawn5.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn5.Y - 2 && destinationX == BlackPawn5.X && BlackPawn5.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn5.Y - 1 && destinationX == BlackPawn5.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn5.Y - 1 && destinationX == BlackPawn5.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn5.Y - 1 && destinationX == BlackPawn5.X + 1)

{

if (EnPassant(BlackPawn5.X, BlackPawn5.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn5.Y - 1 && destinationX == BlackPawn5.X - 1)

{

if (EnPassant(BlackPawn5.X, BlackPawn5.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove6(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn6.X, WhitePawn6.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn6.Captured == false)

{

if (destinationY == WhitePawn6.Y + 1 && destinationX == WhitePawn6.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn6.Y + 2 && destinationX == WhitePawn6.X && WhitePawn6.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn6.Y + 1 && destinationX == WhitePawn6.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn6.Y + 1 && destinationX == WhitePawn6.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn6.Y + 1 && destinationX == WhitePawn6.X + 1)

{

if (EnPassant(WhitePawn6.X, WhitePawn6.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn6.Y + 1 && destinationX == WhitePawn6.X - 1)

{

if (EnPassant(WhitePawn6.X, WhitePawn6.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn6.X, BlackPawn6.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn6.Captured == false)

{

if (destinationY == BlackPawn6.Y - 1 && destinationX == BlackPawn6.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn6.Y - 2 && destinationX == BlackPawn6.X && BlackPawn6.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn6.Y - 1 && destinationX == BlackPawn6.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn6.Y - 1 && destinationX == BlackPawn6.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn6.Y - 1 && destinationX == BlackPawn6.X + 1)

{

if (EnPassant(BlackPawn6.X, BlackPawn6.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn6.Y - 1 && destinationX == BlackPawn6.X - 1)

{

if (EnPassant(BlackPawn6.X, BlackPawn6.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove7(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn7.X, WhitePawn7.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn7.Captured == false)

{

if (destinationY == WhitePawn7.Y + 1 && destinationX == WhitePawn7.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn7.Y + 2 && destinationX == WhitePawn7.X && WhitePawn7.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn7.Y + 1 && destinationX == WhitePawn7.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn7.Y + 1 && destinationX == WhitePawn7.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn7.Y + 1 && destinationX == WhitePawn7.X + 1)

{

if (EnPassant(WhitePawn7.X, WhitePawn7.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn7.Y + 1 && destinationX == WhitePawn7.X - 1)

{

if (EnPassant(WhitePawn7.X, WhitePawn7.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn7.X, BlackPawn7.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn7.Captured == false)

{

if (destinationY == BlackPawn7.Y - 1 && destinationX == BlackPawn7.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn7.Y - 2 && destinationX == BlackPawn7.X && BlackPawn7.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn7.Y - 1 && destinationX == BlackPawn7.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn7.Y - 1 && destinationX == BlackPawn7.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn7.Y - 1 && destinationX == BlackPawn7.X + 1)

{

if (EnPassant(BlackPawn7.X, BlackPawn7.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn7.Y - 1 && destinationX == BlackPawn7.X - 1)

{

if (EnPassant(BlackPawn7.X, BlackPawn7.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool PawnMove8(int destinationX, int destinationY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Pawn Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, WhitePawn8.X, WhitePawn8.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

if (WhitePawn8.Captured == false)

{

if (destinationY == WhitePawn8.Y + 1 && destinationX == WhitePawn8.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up one

}

}

if (destinationY == WhitePawn8.Y + 2 && destinationX == WhitePawn8.X && WhitePawn8.Y == 2)

{

if ((IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY - 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Up two

}

}

if (destinationY == WhitePawn8.Y + 1 && destinationX == WhitePawn8.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Right one

}

}

if (destinationY == WhitePawn8.Y + 1 && destinationX == WhitePawn8.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return true; // Diagonal Up Left one

}

}

if (destinationY == WhitePawn8.Y + 1 && destinationX == WhitePawn8.X + 1)

{

if (EnPassant(WhitePawn8.X, WhitePawn8.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Right one

}

}

if (destinationY == WhitePawn8.Y + 1 && destinationX == WhitePawn8.X - 1)

{

if (EnPassant(WhitePawn8.X, WhitePawn8.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Up Left one

}

}

}

}

else // Black's Turn

{

if (IsOccupied(turn, BlackPawn8.X, BlackPawn8.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

if (BlackPawn8.Captured == false)

{

if (destinationY == BlackPawn8.Y - 1 && destinationX == BlackPawn8.X)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down one

}

}

if (destinationY == BlackPawn8.Y - 2 && destinationX == BlackPawn8.X && BlackPawn8.Y == 7)

{

if ((IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, destinationY + 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0))

{

return true; // Down two

}

}

if (destinationY == BlackPawn8.Y - 1 && destinationX == BlackPawn8.X + 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Right one

}

}

if (destinationY == BlackPawn8.Y - 1 && destinationX == BlackPawn8.X - 1)

{

if ((IsOccupied(turn, destinationX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1))

{

return true; // Diagonal Down Left one

}

}

if (destinationY == BlackPawn8.Y - 1 && destinationX == BlackPawn8.X + 1)

{

if (EnPassant(BlackPawn8.X, BlackPawn8.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Right one

}

}

if (destinationY == BlackPawn8.Y - 1 && destinationX == BlackPawn8.X - 1)

{

if (EnPassant(BlackPawn8.X, BlackPawn8.Y, turn, capture, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8))

{

return true; // En Passant Diagonal Down Left one

}

}

}

}

return false;

}

static bool NewPieceMove(int destinationX, int destinationY, int turn, NewPiece NewPiece, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // New Piece Movement

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

// Check the type of piece moving, create a temporary version of that piece, and then check its movements using other pieces' movement methods.

if (IsEven(turn) == false) // White's Turn

{

if (IsOccupied(turn, NewPiece.X, NewPiece.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2)

{

return false;

}

}

else

{

if (IsOccupied(turn, NewPiece.X, NewPiece.Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1)

{

return false;

}

}

if (NewPiece.Type == "none")

{

return false;

}

if (NewPiece.Type == "queen")

{

if (IsEven(turn) == false) // White's Turn

{

if (NewPiece.Captured == false)

{

if (destinationX > NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

int distanceX = Math.Abs(destinationX - NewPiece.X);

int distanceY = Math.Abs(destinationY - NewPiece.Y);

if (NewPiece.X != destinationX && NewPiece.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > NewPiece.X)

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

else // Black's Turn

{

if (NewPiece.Captured == false)

{

if (destinationX > NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

int distanceX = Math.Abs(destinationX - NewPiece.X);

int distanceY = Math.Abs(destinationY - NewPiece.Y);

if (NewPiece.X != destinationX && NewPiece.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > NewPiece.X)

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

}

if (NewPiece.Type == "bishop")

{

if (IsEven(turn) == false) // White's Turn

{

if (NewPiece.Captured == false)

{

int distanceX = Math.Abs(destinationX - NewPiece.X);

int distanceY = Math.Abs(destinationY - NewPiece.Y);

if (NewPiece.X != destinationX && NewPiece.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > NewPiece.X)

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

else // Black's Turn

{

if (NewPiece.Captured == false)

{

int distanceX = Math.Abs(destinationX - NewPiece.X);

int distanceY = Math.Abs(destinationY - NewPiece.Y);

if (NewPiece.X != destinationX && NewPiece.Y != destinationY)

{

if (distanceX == distanceY)

{

if (destinationX > NewPiece.X)

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Right

}

else

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Right

}

}

else

{

if (destinationY > NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y + Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Up Left

}

else

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

int currentY = NewPiece.Y - Math.Abs(currentX - NewPiece.X);

if ((IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Diagonal Down Left

}

}

}

}

}

}

}

if (NewPiece.Type == "knight")

{

if (IsEven(turn) == false) // White's Turn

{

if (NewPiece.Captured == false)

{

if ((destinationX == NewPiece.X + 2) || (destinationX == NewPiece.X - 2))

{

if ((destinationY == NewPiece.Y + 1) || (destinationY == NewPiece.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == NewPiece.X + 1) || (destinationX == NewPiece.X - 1))

{

if ((destinationY == NewPiece.Y + 2) || (destinationY == NewPiece.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == NewPiece.Y + 2) || (destinationY == NewPiece.Y - 2))

{

if ((destinationX == NewPiece.X + 1) || (destinationX == NewPiece.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == NewPiece.Y + 1) || (destinationY == NewPiece.Y - 1))

{

if ((destinationX == NewPiece.X + 2) || (destinationX == NewPiece.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

else // Black's Turn

{

if (NewPiece.Captured == false)

{

if ((destinationX == NewPiece.X + 2) || (destinationX == NewPiece.X - 2))

{

if ((destinationY == NewPiece.Y + 1) || (destinationY == NewPiece.Y - 1))

{

return true; // Right or Left two, Up or Down one

}

}

if ((destinationX == NewPiece.X + 1) || (destinationX == NewPiece.X - 1))

{

if ((destinationY == NewPiece.Y + 2) || (destinationY == NewPiece.Y - 2))

{

return true; // Right or Left one, Up or Down two

}

}

if ((destinationY == NewPiece.Y + 2) || (destinationY == NewPiece.Y - 2))

{

if ((destinationX == NewPiece.X + 1) || (destinationX == NewPiece.X - 1))

{

return true; // Up or Down two, Left or Right one

}

}

if ((destinationY == NewPiece.Y + 1) || (destinationY == NewPiece.Y - 1))

{

if ((destinationX == NewPiece.X + 2) || (destinationX == NewPiece.X - 2))

{

return true; // Up or Down one, Left or Right two

}

}

}

}

}

if (NewPiece.Type == "rook")

{

if (IsEven(turn) == false) // White's Turn

{

if (NewPiece.Captured == false)

{

if (destinationX > NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

else // Black's Turn

{

if (NewPiece.Captured == false)

{

if (destinationX > NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X + 1; currentX < destinationX; currentX++)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Right

}

if (destinationX < NewPiece.X && destinationY == NewPiece.Y)

{

for (int currentX = NewPiece.X - 1; currentX > destinationX; currentX--)

{

if ((IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, currentX, destinationY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Left

}

if (destinationY > NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y + 1; currentY < destinationY; currentY++)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Up

}

if (destinationY < NewPiece.Y && destinationX == NewPiece.X)

{

for (int currentY = NewPiece.Y - 1; currentY > destinationY; currentY--)

{

if ((IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 1) |

(IsOccupied(turn, destinationX, currentY, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 2))

{

return false; // Check if path is clear

}

}

return true; // Down

}

}

}

}

return false;

}

// Special Moves

static bool EnPassant(int currentX, int currentY, int turn, bool capture, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // En Passant

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

switch (enPassantBlack) // Find which Pawn has passed

{

case 1:

if (currentY == BlackPawn1.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn1.X - 1 || currentX == BlackPawn1.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn1.X, BlackPawn1.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 2:

if (currentY == BlackPawn2.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn2.X - 1 || currentX == BlackPawn2.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn2.X, BlackPawn2.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 3:

if (currentY == BlackPawn3.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn3.X - 1 || currentX == BlackPawn3.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn3.X, BlackPawn3.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 4:

if (currentY == BlackPawn4.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn4.X - 1 || currentX == BlackPawn4.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn4.X, BlackPawn4.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 5:

if (currentY == BlackPawn5.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn5.X - 1 || currentX == BlackPawn5.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn5.X, BlackPawn5.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 6:

if (currentY == BlackPawn6.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn6.X - 1 || currentX == BlackPawn6.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn6.X, BlackPawn6.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 7:

if (currentY == BlackPawn7.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn7.X - 1 || currentX == BlackPawn7.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn7.X, BlackPawn7.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 8:

if (currentY == BlackPawn8.Y) // Make sure both Pawns are on the same row

{

if (currentX == BlackPawn8.X - 1 || currentX == BlackPawn8.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(BlackPawn8.X, BlackPawn8.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

}

}

else // Black's Turn

{

switch (enPassantWhite) // Find which Pawn has passed

{

case 1:

if (currentY == WhitePawn1.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn1.X - 1 || currentX == WhitePawn1.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn1.X, WhitePawn1.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 2:

if (currentY == WhitePawn2.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn2.X - 1 || currentX == WhitePawn2.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn2.X, WhitePawn2.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 3:

if (currentY == WhitePawn3.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn3.X - 1 || currentX == WhitePawn3.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn3.X, WhitePawn3.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 4:

if (currentY == WhitePawn4.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn4.X - 1 || currentX == WhitePawn4.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn4.X, WhitePawn4.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 5:

if (currentY == WhitePawn5.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn5.X - 1 || currentX == WhitePawn5.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn5.X, WhitePawn5.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 6:

if (currentY == WhitePawn6.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn6.X - 1 || currentX == WhitePawn6.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn6.X, WhitePawn6.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 7:

if (currentY == WhitePawn7.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn7.X - 1 || currentX == WhitePawn7.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn7.X, WhitePawn7.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

case 8:

if (currentY == WhitePawn8.Y) // Make sure both Pawns are on the same row

{

if (currentX == WhitePawn8.X - 1 || currentX == WhitePawn8.X + 1)

{ // Make sure both Pawns are adjacent

if (capture)

{

CapturePiece(WhitePawn8.X, WhitePawn8.Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8);

}

return true;

}

}

break;

}

}

return false;

}

static string FindEnPassant(int turn, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Locate En Passant

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's Turn

{

switch (enPassantBlack) // Find which Pawn has passed

{

// Return a code consisting of the capturing pawn's number, the destination letter coordinate, and the destination number coordinate

case 1:

if (EnPassantTest(WhitePawn1, BlackPawn1) == true)

{

return "1h6";

}

if (EnPassantTest(WhitePawn2, BlackPawn1) == true)

{

return "2h6";

}

if (EnPassantTest(WhitePawn3, BlackPawn1) == true)

{

return "3h6";

}

if (EnPassantTest(WhitePawn4, BlackPawn1) == true)

{

return "4h6";

}

if (EnPassantTest(WhitePawn5, BlackPawn1) == true)

{

return "5h6";

}

if (EnPassantTest(WhitePawn6, BlackPawn1) == true)

{

return "6h6";

}

if (EnPassantTest(WhitePawn7, BlackPawn1) == true)

{

return "7h6";

}

if (EnPassantTest(WhitePawn8, BlackPawn1) == true)

{

return "8h6";

}

break;

case 2:

if (EnPassantTest(WhitePawn1, BlackPawn2) == true)

{

return "1g6";

}

if (EnPassantTest(WhitePawn2, BlackPawn2) == true)

{

return "2g6";

}

if (EnPassantTest(WhitePawn3, BlackPawn2) == true)

{

return "3g6";

}

if (EnPassantTest(WhitePawn4, BlackPawn2) == true)

{

return "4g6";

}

if (EnPassantTest(WhitePawn5, BlackPawn2) == true)

{

return "5g6";

}

if (EnPassantTest(WhitePawn6, BlackPawn2) == true)

{

return "6g6";

}

if (EnPassantTest(WhitePawn7, BlackPawn2) == true)

{

return "7g6";

}

if (EnPassantTest(WhitePawn8, BlackPawn2) == true)

{

return "8g6";

}

break;

case 3:

if (EnPassantTest(WhitePawn1, BlackPawn3) == true)

{

return "1f6";

}

if (EnPassantTest(WhitePawn2, BlackPawn3) == true)

{

return "2f6";

}

if (EnPassantTest(WhitePawn3, BlackPawn3) == true)

{

return "3f6";

}

if (EnPassantTest(WhitePawn4, BlackPawn3) == true)

{

return "4f6";

}

if (EnPassantTest(WhitePawn5, BlackPawn3) == true)

{

return "5f6";

}

if (EnPassantTest(WhitePawn6, BlackPawn3) == true)

{

return "6f6";

}

if (EnPassantTest(WhitePawn7, BlackPawn3) == true)

{

return "7f6";

}

if (EnPassantTest(WhitePawn8, BlackPawn3) == true)

{

return "8f6";

}

break;

case 4:

if (EnPassantTest(WhitePawn1, BlackPawn4) == true)

{

return "1e6";

}

if (EnPassantTest(WhitePawn2, BlackPawn4) == true)

{

return "2e6";

}

if (EnPassantTest(WhitePawn3, BlackPawn4) == true)

{

return "3e6";

}

if (EnPassantTest(WhitePawn4, BlackPawn4) == true)

{

return "4e6";

}

if (EnPassantTest(WhitePawn5, BlackPawn4) == true)

{

return "5e6";

}

if (EnPassantTest(WhitePawn6, BlackPawn4) == true)

{

return "6e6";

}

if (EnPassantTest(WhitePawn7, BlackPawn4) == true)

{

return "7e6";

}

if (EnPassantTest(WhitePawn8, BlackPawn4) == true)

{

return "8e6";

}

break;

case 5:

if (EnPassantTest(WhitePawn1, BlackPawn5) == true)

{

return "1d6";

}

if (EnPassantTest(WhitePawn2, BlackPawn5) == true)

{

return "2d6";

}

if (EnPassantTest(WhitePawn3, BlackPawn5) == true)

{

return "3d6";

}

if (EnPassantTest(WhitePawn4, BlackPawn5) == true)

{

return "4d6";

}

if (EnPassantTest(WhitePawn5, BlackPawn5) == true)

{

return "5d6";

}

if (EnPassantTest(WhitePawn6, BlackPawn5) == true)

{

return "6d6";

}

if (EnPassantTest(WhitePawn7, BlackPawn5) == true)

{

return "7d6";

}

if (EnPassantTest(WhitePawn8, BlackPawn5) == true)

{

return "8d6";

}

break;

case 6:

if (EnPassantTest(WhitePawn1, BlackPawn6) == true)

{

return "1c6";

}

if (EnPassantTest(WhitePawn2, BlackPawn6) == true)

{

return "2c6";

}

if (EnPassantTest(WhitePawn3, BlackPawn6) == true)

{

return "3c6";

}

if (EnPassantTest(WhitePawn4, BlackPawn6) == true)

{

return "4c6";

}

if (EnPassantTest(WhitePawn5, BlackPawn6) == true)

{

return "5c6";

}

if (EnPassantTest(WhitePawn6, BlackPawn6) == true)

{

return "6c6";

}

if (EnPassantTest(WhitePawn7, BlackPawn6) == true)

{

return "7c6";

}

if (EnPassantTest(WhitePawn8, BlackPawn6) == true)

{

return "8c6";

}

break;

case 7:

if (EnPassantTest(WhitePawn1, BlackPawn7) == true)

{

return "1b6";

}

if (EnPassantTest(WhitePawn2, BlackPawn7) == true)

{

return "2b6";

}

if (EnPassantTest(WhitePawn3, BlackPawn7) == true)

{

return "3b6";

}

if (EnPassantTest(WhitePawn4, BlackPawn7) == true)

{

return "4b6";

}

if (EnPassantTest(WhitePawn5, BlackPawn7) == true)

{

return "5b6";

}

if (EnPassantTest(WhitePawn6, BlackPawn7) == true)

{

return "6b6";

}

if (EnPassantTest(WhitePawn7, BlackPawn7) == true)

{

return "7b6";

}

if (EnPassantTest(WhitePawn8, BlackPawn7) == true)

{

return "8b6";

}

break;

case 8:

if (EnPassantTest(WhitePawn1, BlackPawn8) == true)

{

return "1a6";

}

if (EnPassantTest(WhitePawn2, BlackPawn8) == true)

{

return "2a6";

}

if (EnPassantTest(WhitePawn3, BlackPawn8) == true)

{

return "3a6";

}

if (EnPassantTest(WhitePawn4, BlackPawn8) == true)

{

return "4a6";

}

if (EnPassantTest(WhitePawn5, BlackPawn8) == true)

{

return "5a6";

}

if (EnPassantTest(WhitePawn6, BlackPawn8) == true)

{

return "6a6";

}

if (EnPassantTest(WhitePawn7, BlackPawn8) == true)

{

return "7a6";

}

if (EnPassantTest(WhitePawn8, BlackPawn8) == true)

{

return "8a6";

}

break;

}

}

else // Black's Turn

{

switch (enPassantWhite) // Find which Pawn has passed

{

// Return a code consisting of the capturing pawn's number, the destination letter coordinate, and the destination number coordinate

case 1:

if (EnPassantTest(BlackPawn1, WhitePawn1) == true)

{

return "1a3";

}

if (EnPassantTest(BlackPawn2, WhitePawn1) == true)

{

return "2a3";

}

if (EnPassantTest(BlackPawn3, WhitePawn1) == true)

{

return "3a3";

}

if (EnPassantTest(BlackPawn4, WhitePawn1) == true)

{

return "4a3";

}

if (EnPassantTest(BlackPawn5, WhitePawn1) == true)

{

return "5a3";

}

if (EnPassantTest(BlackPawn6, WhitePawn1) == true)

{

return "6a3";

}

if (EnPassantTest(BlackPawn7, WhitePawn1) == true)

{

return "7a3";

}

if (EnPassantTest(BlackPawn8, WhitePawn1) == true)

{

return "8a3";

}

break;

case 2:

if (EnPassantTest(BlackPawn1, WhitePawn2) == true)

{

return "1b3";

}

if (EnPassantTest(BlackPawn2, WhitePawn2) == true)

{

return "2b3";

}

if (EnPassantTest(BlackPawn3, WhitePawn2) == true)

{

return "3b3";

}

if (EnPassantTest(BlackPawn4, WhitePawn2) == true)

{

return "4b3";

}

if (EnPassantTest(BlackPawn5, WhitePawn2) == true)

{

return "5b3";

}

if (EnPassantTest(BlackPawn6, WhitePawn2) == true)

{

return "6b3";

}

if (EnPassantTest(BlackPawn7, WhitePawn2) == true)

{

return "7b3";

}

if (EnPassantTest(BlackPawn8, WhitePawn2) == true)

{

return "8b3";

}

break;

case 3:

if (EnPassantTest(BlackPawn1, WhitePawn3) == true)

{

return "1c3";

}

if (EnPassantTest(BlackPawn2, WhitePawn3) == true)

{

return "2c3";

}

if (EnPassantTest(BlackPawn3, WhitePawn3) == true)

{

return "3c3";

}

if (EnPassantTest(BlackPawn4, WhitePawn3) == true)

{

return "4c3";

}

if (EnPassantTest(BlackPawn5, WhitePawn3) == true)

{

return "5c3";

}

if (EnPassantTest(BlackPawn6, WhitePawn3) == true)

{

return "6c3";

}

if (EnPassantTest(BlackPawn7, WhitePawn3) == true)

{

return "7c3";

}

if (EnPassantTest(BlackPawn8, WhitePawn3) == true)

{

return "8c3";

}

break;

case 4:

if (EnPassantTest(BlackPawn1, WhitePawn4) == true)

{

return "1d3";

}

if (EnPassantTest(BlackPawn2, WhitePawn4) == true)

{

return "2d3";

}

if (EnPassantTest(BlackPawn3, WhitePawn4) == true)

{

return "3d3";

}

if (EnPassantTest(BlackPawn4, WhitePawn4) == true)

{

return "4d3";

}

if (EnPassantTest(BlackPawn5, WhitePawn4) == true)

{

return "5d3";

}

if (EnPassantTest(BlackPawn6, WhitePawn4) == true)

{

return "6d3";

}

if (EnPassantTest(BlackPawn7, WhitePawn4) == true)

{

return "7d3";

}

if (EnPassantTest(BlackPawn8, WhitePawn4) == true)

{

return "8d3";

}

break;

case 5:

if (EnPassantTest(BlackPawn1, WhitePawn5) == true)

{

return "1e3";

}

if (EnPassantTest(BlackPawn2, WhitePawn5) == true)

{

return "2e3";

}

if (EnPassantTest(BlackPawn3, WhitePawn5) == true)

{

return "3e3";

}

if (EnPassantTest(BlackPawn4, WhitePawn5) == true)

{

return "4e3";

}

if (EnPassantTest(BlackPawn5, WhitePawn5) == true)

{

return "5e3";

}

if (EnPassantTest(BlackPawn6, WhitePawn5) == true)

{

return "6e3";

}

if (EnPassantTest(BlackPawn7, WhitePawn5) == true)

{

return "7e3";

}

if (EnPassantTest(BlackPawn8, WhitePawn5) == true)

{

return "8e3";

}

break;

case 6:

if (EnPassantTest(BlackPawn1, WhitePawn6) == true)

{

return "1f3";

}

if (EnPassantTest(BlackPawn2, WhitePawn6) == true)

{

return "2f3";

}

if (EnPassantTest(BlackPawn3, WhitePawn6) == true)

{

return "3f3";

}

if (EnPassantTest(BlackPawn4, WhitePawn6) == true)

{

return "4f3";

}

if (EnPassantTest(BlackPawn5, WhitePawn6) == true)

{

return "5f3";

}

if (EnPassantTest(BlackPawn6, WhitePawn6) == true)

{

return "6f3";

}

if (EnPassantTest(BlackPawn7, WhitePawn6) == true)

{

return "7f3";

}

if (EnPassantTest(BlackPawn8, WhitePawn6) == true)

{

return "8f3";

}

break;

case 7:

if (EnPassantTest(BlackPawn1, WhitePawn7) == true)

{

return "1g3";

}

if (EnPassantTest(BlackPawn2, WhitePawn7) == true)

{

return "2g3";

}

if (EnPassantTest(BlackPawn3, WhitePawn7) == true)

{

return "3g3";

}

if (EnPassantTest(BlackPawn4, WhitePawn7) == true)

{

return "4g3";

}

if (EnPassantTest(BlackPawn5, WhitePawn7) == true)

{

return "5g3";

}

if (EnPassantTest(BlackPawn6, WhitePawn7) == true)

{

return "6g3";

}

if (EnPassantTest(BlackPawn7, WhitePawn7) == true)

{

return "7g3";

}

if (EnPassantTest(BlackPawn8, WhitePawn7) == true)

{

return "8g3";

}

break;

case 8:

if (EnPassantTest(BlackPawn1, WhitePawn8) == true)

{

return "1h3";

}

if (EnPassantTest(BlackPawn2, WhitePawn8) == true)

{

return "2h3";

}

if (EnPassantTest(BlackPawn3, WhitePawn8) == true)

{

return "3h3";

}

if (EnPassantTest(BlackPawn4, WhitePawn8) == true)

{

return "4h3";

}

if (EnPassantTest(BlackPawn5, WhitePawn8) == true)

{

return "5h3";

}

if (EnPassantTest(BlackPawn6, WhitePawn8) == true)

{

return "6h3";

}

if (EnPassantTest(BlackPawn7, WhitePawn8) == true)

{

return "7h3";

}

if (EnPassantTest(BlackPawn8, WhitePawn8) == true)

{

return "8h3";

}

break;

default: break;

}

}

return "1a1";

}

static bool EnPassantTest(Pawn Pawn, Pawn PassedPawn) // Test for En Passant

{

if (PassedPawn.Y == Pawn.Y)

{

if (PassedPawn.X == Pawn.X + 1)

{

return true;

}

if (PassedPawn.X == Pawn.X - 1)

{

return true;

}

}

return false;

}

static bool CastlingTest(int turn, int enPassantWhite, int enPassantBlack, int castleValue, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Test for Castling

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's turn

{

if (castleValue == 1) // White kingside

{

if (WhiteKing.X == 5 && WhiteKing.Y == 1 && WhiteKing.HasMoved == false)

{

if (WhiteRook2.X == 8 && WhiteRook2.Y == 1 && WhiteRook2.HasMoved == false)

{

if (IsOccupied(turn, 6, 1, WhitePromo, BlackPromo, // Determine if the way is clear and unchecked

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 7, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 5, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 6, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 7, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

return true;

}

}

}

}

}

}

}

}

if (castleValue == 2) // White queenside

{

if (WhiteKing.X == 5 && WhiteKing.Y == 1 && WhiteKing.HasMoved == false)

{

if (WhiteRook1.X == 1 && WhiteRook1.Y == 1 && WhiteRook1.HasMoved == false)

{

if (IsOccupied(turn, 2, 1, WhitePromo, BlackPromo, // Determine if the way is clear and unchecked

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 3, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 4, 1, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 3, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 4, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 5, 1, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

return true;

}

}

}

}

}

}

}

}

}

}

else // Black's turn

{

if (castleValue == 1) // Black kingside

{

if (BlackKing.X == 5 && BlackKing.Y == 8 && BlackKing.HasMoved == false)

{

if (BlackRook1.X == 8 && BlackRook1.Y == 8 && BlackRook1.HasMoved == false)

{

if (IsOccupied(turn, 6, 8, WhitePromo, BlackPromo, // Determine if the way is clear and unchecked

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 7, 8, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 5, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 6, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 7, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

return true;

}

}

}

}

}

}

}

}

if (castleValue == 2) // Black queenside

{

if (BlackKing.X == 5 && BlackKing.Y == 8 && BlackKing.HasMoved == false)

{

if (BlackRook2.X == 1 && BlackRook2.Y == 8 && BlackRook2.HasMoved == false)

{

if (IsOccupied(turn, 2, 8, WhitePromo, BlackPromo, // Determine if the way is clear and unchecked

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 3, 8, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (IsOccupied(turn, 4, 8, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == 0)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 3, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 4, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

if (InCheck(turn, enPassantWhite, enPassantBlack, 5, 8, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

return true;

}

}

}

}

}

}

}

}

}

}

return false;

}

static bool Promotion(int turn, // Promotion

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's turn

{

if (WhitePawn1.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn1 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn1.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn1.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn1.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn1.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn2.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn2 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn2.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn2.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn2.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn2.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn3.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn3 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn3.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn3.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn3.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn3.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn4.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn4 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn4.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn4.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn4.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn4.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn5.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn5 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn5.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn5.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn5.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn5.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn6.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn6 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn6.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn6.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn6.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn6.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn7.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn7 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn7.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn7.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn7.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn7.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (WhitePawn8.Y == 8) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn8 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

WhitePawn8.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

WhitePawn8.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

WhitePawn8.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

WhitePawn8.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

}

else // Black's turn

{

if (BlackPawn1.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn1 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn1.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn1.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn1.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn1.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn2.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn2 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn2.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn2.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn2.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn2.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn3.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn3 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn3.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn3.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn3.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn3.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn4.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn4 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn4.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn4.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn4.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn4.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn5.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn5 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn5.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn5.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn5.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn5.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn6.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn6 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn6.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn6.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn6.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn6.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn7.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn7 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn7.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn7.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn7.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn7.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

if (BlackPawn8.Y == 1) // Determine which pawn is at the top

{

string promotion;

while (true) // Until a valid promotion is selected

{

Console.ForegroundColor = ConsoleColor.DarkBlue; // Prompt

Console.WriteLine("\nPromote Pawn8 to Queen, Bishop, Knight, or Rook?\n");

Console.ForegroundColor = ConsoleColor.White;

promotion = Console.ReadLine(); // Read input

promotion = promotion.ToLower(); // Set input to lowercase

promotion = promotion.Replace(" ", string.Empty); // Remove any spaces from input

if (promotion == "queen" || promotion == "q")

{

BlackPawn8.Promotion = "queen"; // Queen selected

return true;

}

if (promotion == "bishop" || promotion == "b")

{

BlackPawn8.Promotion = "bishop"; // Bishop selected

return true;

}

if (promotion == "knight" || promotion == "n" || promotion == "k")

{

BlackPawn8.Promotion = "knight"; // Knight selected

return true;

}

if (promotion == "rook" || promotion == "r")

{

BlackPawn8.Promotion = "rook"; // Rook selected

return true;

}

Console.ForegroundColor = ConsoleColor.DarkRed; // Bad input

Console.WriteLine("\nThat is not a valid promotion.\n");

}

}

}

return false;

}

static string CreateNewPiece(NewPiece[] WhitePromo, NewPiece[] BlackPromo,

int queenCountWhite, int bishopCountWhite, int knightCountWhite, int rookCountWhite,

int queenCountBlack, int bishopCountBlack, int knightCountBlack, int rookCountBlack,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

int selection = 0;

// Find which Pawn is currently promoted, then replace that Pawn with the selected New Piece.

if (WhitePawn1.Promotion == "queen") // White Pawn1 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn1.X;

WhitePromo[i].Y = WhitePawn1.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn1.X = 0;

WhitePawn1.Y = 0;

WhitePawn1.Captured = true;

WhitePawn1.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn1.Promotion == "bishop") // White Pawn1 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn1.X;

WhitePromo[i].Y = WhitePawn1.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn1.X = 0;

WhitePawn1.Y = 0;

WhitePawn1.Captured = true;

WhitePawn1.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn1.Promotion == "knight") // White Pawn1 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn1.X;

WhitePromo[i].Y = WhitePawn1.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn1.X = 0;

WhitePawn1.Y = 0;

WhitePawn1.Captured = true;

WhitePawn1.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn1.Promotion == "rook") // White Pawn1 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn1.X;

WhitePromo[i].Y = WhitePawn1.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn1.X = 0;

WhitePawn1.Y = 0;

WhitePawn1.Captured = true;

WhitePawn1.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn1.Promotion == "queen") // Black Pawn1 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn1.X;

BlackPromo[i].Y = BlackPawn1.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn1.X = 0;

BlackPawn1.Y = 0;

BlackPawn1.Captured = true;

BlackPawn1.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn1.Promotion == "bishop") // Black Pawn1 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn1.X;

BlackPromo[i].Y = BlackPawn1.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn1.X = 0;

BlackPawn1.Y = 0;

BlackPawn1.Captured = true;

BlackPawn1.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn1.Promotion == "knight") // Black Pawn1 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn1.X;

BlackPromo[i].Y = BlackPawn1.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn1.X = 0;

BlackPawn1.Y = 0;

BlackPawn1.Captured = true;

BlackPawn1.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn1.Promotion == "rook") // Black Pawn1 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn1.X;

BlackPromo[i].Y = BlackPawn1.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn1.X = 0;

BlackPawn1.Y = 0;

BlackPawn1.Captured = true;

BlackPawn1.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn2.Promotion == "queen") // White Pawn2 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn2.X;

WhitePromo[i].Y = WhitePawn2.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn2.X = 0;

WhitePawn2.Y = 0;

WhitePawn2.Captured = true;

WhitePawn2.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn2.Promotion == "bishop") // White Pawn2 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn2.X;

WhitePromo[i].Y = WhitePawn2.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn2.X = 0;

WhitePawn2.Y = 0;

WhitePawn2.Captured = true;

WhitePawn2.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn2.Promotion == "knight") // White Pawn2 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn2.X;

WhitePromo[i].Y = WhitePawn2.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn2.X = 0;

WhitePawn2.Y = 0;

WhitePawn2.Captured = true;

WhitePawn2.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn2.Promotion == "rook") // White Pawn2 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn2.X;

WhitePromo[i].Y = WhitePawn2.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn2.X = 0;

WhitePawn2.Y = 0;

WhitePawn2.Captured = true;

WhitePawn2.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn2.Promotion == "queen") // Black Pawn2 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn2.X;

BlackPromo[i].Y = BlackPawn2.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn2.X = 0;

BlackPawn2.Y = 0;

BlackPawn2.Captured = true;

BlackPawn2.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn2.Promotion == "bishop") // Black Pawn2 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn2.X;

BlackPromo[i].Y = BlackPawn2.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn2.X = 0;

BlackPawn2.Y = 0;

BlackPawn2.Captured = true;

BlackPawn2.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn2.Promotion == "knight") // Black Pawn2 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn2.X;

BlackPromo[i].Y = BlackPawn2.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn2.X = 0;

BlackPawn2.Y = 0;

BlackPawn2.Captured = true;

BlackPawn2.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn2.Promotion == "rook") // Black Pawn2 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn2.X;

BlackPromo[i].Y = BlackPawn2.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn2.X = 0;

BlackPawn2.Y = 0;

BlackPawn2.Captured = true;

BlackPawn2.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn3.Promotion == "queen") // White Pawn3 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn3.X;

WhitePromo[i].Y = WhitePawn3.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn3.X = 0;

WhitePawn3.Y = 0;

WhitePawn3.Captured = true;

WhitePawn3.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn3.Promotion == "bishop") // White Pawn3 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn3.X;

WhitePromo[i].Y = WhitePawn3.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn3.X = 0;

WhitePawn3.Y = 0;

WhitePawn3.Captured = true;

WhitePawn3.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn3.Promotion == "knight") // White Pawn3 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn3.X;

WhitePromo[i].Y = WhitePawn3.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn3.X = 0;

WhitePawn3.Y = 0;

WhitePawn3.Captured = true;

WhitePawn3.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn3.Promotion == "rook") // White Pawn3 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn3.X;

WhitePromo[i].Y = WhitePawn3.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn3.X = 0;

WhitePawn3.Y = 0;

WhitePawn3.Captured = true;

WhitePawn3.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn3.Promotion == "queen") // Black Pawn3 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn3.X;

BlackPromo[i].Y = BlackPawn3.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn3.X = 0;

BlackPawn3.Y = 0;

BlackPawn3.Captured = true;

BlackPawn3.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn3.Promotion == "bishop") // Black Pawn3 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn3.X;

BlackPromo[i].Y = BlackPawn3.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn3.X = 0;

BlackPawn3.Y = 0;

BlackPawn3.Captured = true;

BlackPawn3.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn3.Promotion == "knight") // Black Pawn3 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn3.X;

BlackPromo[i].Y = BlackPawn3.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn3.X = 0;

BlackPawn3.Y = 0;

BlackPawn3.Captured = true;

BlackPawn3.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn3.Promotion == "rook") // Black Pawn3 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn3.X;

BlackPromo[i].Y = BlackPawn3.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn3.X = 0;

BlackPawn3.Y = 0;

BlackPawn3.Captured = true;

BlackPawn3.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn4.Promotion == "queen") // White Pawn4 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn4.X;

WhitePromo[i].Y = WhitePawn4.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn4.X = 0;

WhitePawn4.Y = 0;

WhitePawn4.Captured = true;

WhitePawn4.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn4.Promotion == "bishop") // White Pawn4 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn4.X;

WhitePromo[i].Y = WhitePawn4.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn4.X = 0;

WhitePawn4.Y = 0;

WhitePawn4.Captured = true;

WhitePawn4.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn4.Promotion == "knight") // White Pawn4 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn4.X;

WhitePromo[i].Y = WhitePawn4.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn4.X = 0;

WhitePawn4.Y = 0;

WhitePawn4.Captured = true;

WhitePawn4.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn4.Promotion == "rook") // White Pawn4 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn4.X;

WhitePromo[i].Y = WhitePawn4.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn4.X = 0;

WhitePawn4.Y = 0;

WhitePawn4.Captured = true;

WhitePawn4.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn4.Promotion == "queen") // Black Pawn4 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn4.X;

BlackPromo[i].Y = BlackPawn4.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn4.X = 0;

BlackPawn4.Y = 0;

BlackPawn4.Captured = true;

BlackPawn4.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn4.Promotion == "bishop") // Black Pawn4 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn4.X;

BlackPromo[i].Y = BlackPawn4.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn4.X = 0;

BlackPawn4.Y = 0;

BlackPawn4.Captured = true;

BlackPawn4.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn4.Promotion == "knight") // Black Pawn4 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn4.X;

BlackPromo[i].Y = BlackPawn4.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn4.X = 0;

BlackPawn4.Y = 0;

BlackPawn4.Captured = true;

BlackPawn4.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn4.Promotion == "rook") // Black Pawn4 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn4.X;

BlackPromo[i].Y = BlackPawn4.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn4.X = 0;

BlackPawn4.Y = 0;

BlackPawn4.Captured = true;

BlackPawn4.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn5.Promotion == "queen") // White Pawn5 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn5.X;

WhitePromo[i].Y = WhitePawn5.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn5.X = 0;

WhitePawn5.Y = 0;

WhitePawn5.Captured = true;

WhitePawn5.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn5.Promotion == "bishop") // White Pawn5 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn5.X;

WhitePromo[i].Y = WhitePawn5.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn5.X = 0;

WhitePawn5.Y = 0;

WhitePawn5.Captured = true;

WhitePawn5.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn5.Promotion == "knight") // White Pawn5 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn5.X;

WhitePromo[i].Y = WhitePawn5.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn5.X = 0;

WhitePawn5.Y = 0;

WhitePawn5.Captured = true;

WhitePawn5.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn5.Promotion == "rook") // White Pawn5 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn5.X;

WhitePromo[i].Y = WhitePawn5.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn5.X = 0;

WhitePawn5.Y = 0;

WhitePawn5.Captured = true;

WhitePawn5.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn5.Promotion == "queen") // Black Pawn5 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn5.X;

BlackPromo[i].Y = BlackPawn5.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn5.X = 0;

BlackPawn5.Y = 0;

BlackPawn5.Captured = true;

BlackPawn5.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn5.Promotion == "bishop") // Black Pawn5 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn5.X;

BlackPromo[i].Y = BlackPawn5.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn5.X = 0;

BlackPawn5.Y = 0;

BlackPawn5.Captured = true;

BlackPawn5.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn5.Promotion == "knight") // Black Pawn5 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn5.X;

BlackPromo[i].Y = BlackPawn5.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn5.X = 0;

BlackPawn5.Y = 0;

BlackPawn5.Captured = true;

BlackPawn5.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn5.Promotion == "rook") // Black Pawn5 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn5.X;

BlackPromo[i].Y = BlackPawn5.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn5.X = 0;

BlackPawn5.Y = 0;

BlackPawn5.Captured = true;

BlackPawn5.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn6.Promotion == "queen") // White Pawn6 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn6.X;

WhitePromo[i].Y = WhitePawn6.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn6.X = 0;

WhitePawn6.Y = 0;

WhitePawn6.Captured = true;

WhitePawn6.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn6.Promotion == "bishop") // White Pawn6 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn6.X;

WhitePromo[i].Y = WhitePawn6.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn6.X = 0;

WhitePawn6.Y = 0;

WhitePawn6.Captured = true;

WhitePawn6.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn6.Promotion == "knight") // White Pawn6 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn6.X;

WhitePromo[i].Y = WhitePawn6.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn6.X = 0;

WhitePawn6.Y = 0;

WhitePawn6.Captured = true;

WhitePawn6.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn6.Promotion == "rook") // White Pawn6 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn6.X;

WhitePromo[i].Y = WhitePawn6.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn6.X = 0;

WhitePawn6.Y = 0;

WhitePawn6.Captured = true;

WhitePawn6.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn6.Promotion == "queen") // Black Pawn6 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn6.X;

BlackPromo[i].Y = BlackPawn6.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn6.X = 0;

BlackPawn6.Y = 0;

BlackPawn6.Captured = true;

BlackPawn6.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn6.Promotion == "bishop") // Black Pawn6 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn6.X;

BlackPromo[i].Y = BlackPawn6.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn6.X = 0;

BlackPawn6.Y = 0;

BlackPawn6.Captured = true;

BlackPawn6.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn6.Promotion == "knight") // Black Pawn6 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn6.X;

BlackPromo[i].Y = BlackPawn6.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn6.X = 0;

BlackPawn6.Y = 0;

BlackPawn6.Captured = true;

BlackPawn6.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn6.Promotion == "rook") // Black Pawn6 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn6.X;

BlackPromo[i].Y = BlackPawn6.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn6.X = 0;

BlackPawn6.Y = 0;

BlackPawn6.Captured = true;

BlackPawn6.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn7.Promotion == "queen") // White Pawn7 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn7.X;

WhitePromo[i].Y = WhitePawn7.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn7.X = 0;

WhitePawn7.Y = 0;

WhitePawn7.Captured = true;

WhitePawn7.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn7.Promotion == "bishop") // White Pawn7 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn7.X;

WhitePromo[i].Y = WhitePawn7.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn7.X = 0;

WhitePawn7.Y = 0;

WhitePawn7.Captured = true;

WhitePawn7.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn7.Promotion == "knight") // White Pawn7 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn7.X;

WhitePromo[i].Y = WhitePawn7.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn7.X = 0;

WhitePawn7.Y = 0;

WhitePawn7.Captured = true;

WhitePawn7.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn7.Promotion == "rook") // White Pawn7 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn7.X;

WhitePromo[i].Y = WhitePawn7.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn7.X = 0;

WhitePawn7.Y = 0;

WhitePawn7.Captured = true;

WhitePawn7.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn7.Promotion == "queen") // Black Pawn7 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn7.X;

BlackPromo[i].Y = BlackPawn7.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn7.X = 0;

BlackPawn7.Y = 0;

BlackPawn7.Captured = true;

BlackPawn7.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn7.Promotion == "bishop") // Black Pawn7 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn7.X;

BlackPromo[i].Y = BlackPawn7.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn7.X = 0;

BlackPawn7.Y = 0;

BlackPawn7.Captured = true;

BlackPawn7.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn7.Promotion == "knight") // Black Pawn7 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn7.X;

BlackPromo[i].Y = BlackPawn7.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn7.X = 0;

BlackPawn7.Y = 0;

BlackPawn7.Captured = true;

BlackPawn7.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn7.Promotion == "rook") // Black Pawn7 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn7.X;

BlackPromo[i].Y = BlackPawn7.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn7.X = 0;

BlackPawn7.Y = 0;

BlackPawn7.Captured = true;

BlackPawn7.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (WhitePawn8.Promotion == "queen") // White Pawn8 to Queen

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn8.X;

WhitePromo[i].Y = WhitePawn8.Y;

WhitePromo[i].Tag = "queen" + queenCountWhite.ToString();

WhitePromo[i].Type = "queen";

selection = i;

break;

}

}

WhitePawn8.X = 0;

WhitePawn8.Y = 0;

WhitePawn8.Captured = true;

WhitePawn8.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn8.Promotion == "bishop") // White Pawn8 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn8.X;

WhitePromo[i].Y = WhitePawn8.Y;

WhitePromo[i].Tag = "bishop" + bishopCountWhite.ToString();

WhitePromo[i].Type = "bishop";

selection = i;

break;

}

}

WhitePawn8.X = 0;

WhitePawn8.Y = 0;

WhitePawn8.Captured = true;

WhitePawn8.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn8.Promotion == "knight") // White Pawn8 to Knight

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn8.X;

WhitePromo[i].Y = WhitePawn8.Y;

WhitePromo[i].Tag = "knight" + knightCountWhite.ToString();

WhitePromo[i].Type = "knight";

selection = i;

break;

}

}

WhitePawn8.X = 0;

WhitePawn8.Y = 0;

WhitePawn8.Captured = true;

WhitePawn8.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (WhitePawn8.Promotion == "rook") // White Pawn8 to Rook

{

for (int i = 0; i < 8; i++)

{

if (WhitePromo[i].Type == "none")

{

WhitePromo[i].X = WhitePawn8.X;

WhitePromo[i].Y = WhitePawn8.Y;

WhitePromo[i].Tag = "rook" + rookCountWhite.ToString();

WhitePromo[i].Type = "rook";

selection = i;

break;

}

}

WhitePawn8.X = 0;

WhitePawn8.Y = 0;

WhitePawn8.Captured = true;

WhitePawn8.Promotion = "pawn";

return WhitePromo[selection].Tag;

}

if (BlackPawn8.Promotion == "queen") // Black Pawn8 to Queen

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn8.X;

BlackPromo[i].Y = BlackPawn8.Y;

BlackPromo[i].Tag = "queen" + queenCountBlack.ToString();

BlackPromo[i].Type = "queen";

selection = i;

break;

}

}

BlackPawn8.X = 0;

BlackPawn8.Y = 0;

BlackPawn8.Captured = true;

BlackPawn8.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn8.Promotion == "bishop") // Black Pawn8 to Bishop

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn8.X;

BlackPromo[i].Y = BlackPawn8.Y;

BlackPromo[i].Tag = "bishop" + bishopCountBlack.ToString();

BlackPromo[i].Type = "bishop";

selection = i;

break;

}

}

BlackPawn8.X = 0;

BlackPawn8.Y = 0;

BlackPawn8.Captured = true;

BlackPawn8.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn8.Promotion == "knight") // Black Pawn8 to Knight

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn8.X;

BlackPromo[i].Y = BlackPawn8.Y;

BlackPromo[i].Tag = "knight" + knightCountBlack.ToString();

BlackPromo[i].Type = "knight";

selection = i;

break;

}

}

BlackPawn8.X = 0;

BlackPawn8.Y = 0;

BlackPawn8.Captured = true;

BlackPawn8.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

if (BlackPawn8.Promotion == "rook") // Black Pawn8 to Rook

{

for (int i = 0; i < 8; i++)

{

if (BlackPromo[i].Type == "none")

{

BlackPromo[i].X = BlackPawn8.X;

BlackPromo[i].Y = BlackPawn8.Y;

BlackPromo[i].Tag = "rook" + rookCountBlack.ToString();

BlackPromo[i].Type = "rook";

selection = i;

break;

}

}

BlackPawn8.X = 0;

BlackPawn8.Y = 0;

BlackPawn8.Captured = true;

BlackPawn8.Promotion = "pawn";

return BlackPromo[selection].Tag;

}

return "";

}

static string FindPromotion(int turn, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Locate Promotion

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

if (IsEven(turn) == false) // White's turn

{

for (int X = 1; X <= 8; X++) // Check all possible spaces for promotion

{

if (PawnMove1(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "1" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove2(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "2" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove3(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "3" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove4(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "4" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove5(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "5" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove6(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "6" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove7(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "7" + NumberToLetter(X) + "8";

return code;

}

if (PawnMove8(X, 8, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "8" + NumberToLetter(X) + "8";

return code;

}

}

}

else // Black's turn

{

for (int X = 8; X >= 1; X--) // Check all possible spaces for promotion

{

if (PawnMove1(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "1" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove2(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "2" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove3(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "3" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove4(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "4" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove5(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "5" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove6(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "6" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove7(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "7" + NumberToLetter(X) + "1";

return code;

}

if (PawnMove8(X, 1, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

string code = "8" + NumberToLetter(X) + "1";

return code;

}

}

}

return "1h8";

}

static int LetterToNumber(char X) // Convert Letter to Number

{

switch (X)

{

case 'a':

return 1;

case 'b':

return 2;

case 'c':

return 3;

case 'd':

return 4;

case 'e':

return 5;

case 'f':

return 6;

case 'g':

return 7;

case 'h':

return 8;

}

return 0;

}

static string NumberToLetter(int X) // Convert Number to Letter

{

switch (X)

{

case 1:

return "a";

case 2:

return "b";

case 3:

return "c";

case 4:

return "d";

case 5:

return "e";

case 6:

return "f";

case 7:

return "g";

case 8:

return "h";

}

return "";

}

static bool InCheck(int turn, int enPassantWhite, int enPassantBlack, int X, int Y, bool message, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Is that space in Check next turn?

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

turn = turn + 1; // Check for next turn's movements

// Check each piece's possible movements for collisions with the selected coordinates, and that your destination isn't one of your own pieces.

if (KingMove(X, Y, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White King is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black King is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (QueenMove(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Queen is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Queen is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (BishopMove1(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Bishop is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Bishop is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (BishopMove2(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Bishop is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Bishop is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (KnightMove1(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Knight is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Knight is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (KnightMove2(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Knight is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Knight is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (RookMove1(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Rook is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Rook is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (RookMove2(X, Y, turn, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Rook is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Rook is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove1(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove2(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove3(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove4(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove5(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove6(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove7(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (PawnMove8(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

if (IsEven(turn) == false)

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black Pawn is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

}

return true;

}

if (IsEven(turn) == false)

{

for (int i = 0; i < WhitePromo.Length; i++)

{

if (NewPieceMove(X, Y, turn, WhitePromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

string piece = WhitePromo[i].Tag;

string piece1 = piece.ToUpper();

string piece2 = piece.ToLower();

piece = piece1[0] + piece2;

piece = piece.Remove(1, 1);

piece = piece.Remove(piece.Length - 1, 1);

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("White " + piece + " is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

return true;

}

}

}

else

{

for (int i = 0; i < BlackPromo.Length; i++)

{

if (NewPieceMove(X, Y, turn, BlackPromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (message == true)

{

string piece = BlackPromo[i].Tag;

string piece1 = piece.ToUpper();

string piece2 = piece.ToLower();

piece = piece1[0] + piece2;

piece = piece.Remove(1, 1);

piece = piece.Remove(piece.Length - 1, 1);

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("Black " + piece + " is attacking.\n");

Console.ForegroundColor = ConsoleColor.Black;

}

return true;

}

}

}

return false;

}

// CHECKMATE TEST CODE SEQUENCES:

// ENTER THE SEQUENCE IN THE CENTER TO TRY TO GET THE RESULT ON THE LEFT, RECORD RESULT ON RIGHT.

// WHITE FAR MATE: p5e4 p2g5 p1a3 p3f5 qh5 SUCCESS

// WHITE FAR CHECK: p5e4 p3f5 qh5 SUCCESS

// WHITE CLOSE MATE: p5e4 p1h6 f1c4 p1h5 qh5 p2g5 qf7 SUCCESS

// WHITE CLOSE CHECK: p5e4 p1h6 qh5 p2g5 qf7 SUCCESS

// BLACK FAR MATE: p7g4 p4e5 p6f4 qh4 SUCCESS

// BLACK FAR CHECK: p1a3 p4e5 p6f4 qh4 SUCCESS

// BLACK CLOSE MATE: p1a3 p4e5 p1a4 b1c5 p1a5 qh4 p1a6 qf2 SUCCESS

// BLACK CLOSE CHECK: p1a3 p4e5 p1a4 qh4 p1a5 qf2 SUCCESS

static bool Checkmate(int turn, int enPassantWhite, int enPassantBlack, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Is there any possible moves that remove Check?

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

int checkedX = 0;

int checkedY = 0;

if (IsEven(turn) == false) // Stalemate detection (King must be in check to be Checkmate)

{

checkedX = WhiteKing.X;

checkedY = WhiteKing.Y;

}

else

{

checkedX = BlackKing.X;

checkedY = BlackKing.Y;

}

// Check each piece's possible movements. If that movement removes Check, return false. If there are no movements that remove check, return true.

if (InCheck(turn, enPassantWhite, enPassantBlack, checkedX, checkedY, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

for (int Y = 1; Y < 9; Y++)

{

for (int X = 1; X < 9; X++)

{

/\*

Console.Write(X + ", " + Y);

Console.WriteLine(" " + IsOccupied(turn, X, Y, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8));

\*/

if (KingMove(X, Y, turn, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // King movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteKing.X; // Piece's current X

int pieceY = WhiteKing.Y; // Piece's current Y

WhiteKing.X = X; // New piece position

WhiteKing.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKing.X = pieceX; // Put piece back

WhiteKing.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteKing.X = pieceX; // Put piece back

WhiteKing.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackKing.X; // Piece's current X

int pieceY = BlackKing.Y; // Piece's current Y

BlackKing.X = X; // New piece position

BlackKing.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKing.X = pieceX; // Put piece back

BlackKing.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackKing.X = pieceX; // Put piece back

BlackKing.Y = pieceY; // Put piece back

}

}

}

if (QueenMove(X, Y, turn, WhitePromo, BlackPromo, // Queen movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteQueen.X; // Piece's current X

int pieceY = WhiteQueen.Y; // Piece's current Y

WhiteQueen.X = X; // New piece position

WhiteQueen.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteQueen.X = pieceX; // Put piece back

WhiteQueen.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteQueen.X = pieceX; // Put piece back

WhiteQueen.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackQueen.X; // Piece's current X

int pieceY = BlackQueen.Y; // Piece's current Y

BlackQueen.X = X; // New piece position

BlackQueen.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackQueen.X = pieceX; // Put piece back

BlackQueen.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackQueen.X = pieceX; // Put piece back

BlackQueen.Y = pieceY; // Put piece back

}

}

}

if (BishopMove1(X, Y, turn, WhitePromo, BlackPromo, // Bishop1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteBishop1.X; // Piece's current X

int pieceY = WhiteBishop1.Y; // Piece's current Y

WhiteBishop1.X = X; // New piece position

WhiteBishop1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteBishop1.X = pieceX; // Put piece back

WhiteBishop1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteBishop1.X = pieceX; // Put piece back

WhiteBishop1.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackBishop1.X; // Piece's current X

int pieceY = BlackBishop1.Y; // Piece's current Y

BlackBishop1.X = X; // New piece position

BlackBishop1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackBishop1.X = pieceX; // Put piece back

BlackBishop1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackBishop1.X = pieceX; // Put piece back

BlackBishop1.Y = pieceY; // Put piece back

}

}

}

if (BishopMove2(X, Y, turn, WhitePromo, BlackPromo, // Bishop2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteBishop2.X; // Piece's current X

int pieceY = WhiteBishop2.Y; // Piece's current Y

WhiteBishop2.X = X; // New piece position

WhiteBishop2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteBishop2.X = pieceX; // Put piece back

WhiteBishop2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteBishop2.X = pieceX; // Put piece back

WhiteBishop2.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackBishop2.X; // Piece's current X

int pieceY = BlackBishop2.Y; // Piece's current Y

BlackBishop2.X = X; // New piece position

BlackBishop2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackBishop2.X = pieceX; // Put piece back

BlackBishop2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackBishop2.X = pieceX; // Put piece back

BlackBishop2.Y = pieceY; // Put piece back

}

}

}

if (KnightMove1(X, Y, turn, WhitePromo, BlackPromo, // Knight1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteKnight1.X; // Piece's current X

int pieceY = WhiteKnight1.Y; // Piece's current Y

WhiteKnight1.X = X; // New piece position

WhiteKnight1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKnight1.X = pieceX; // Put piece back

WhiteKnight1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteKnight1.X = pieceX; // Put piece back

WhiteKnight1.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackKnight1.X; // Piece's current X

int pieceY = BlackKnight1.Y; // Piece's current Y

BlackKnight1.X = X; // New piece position

BlackKnight1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKnight1.X = pieceX; // Put piece back

BlackKnight1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackKnight1.X = pieceX; // Put piece back

BlackKnight1.Y = pieceY; // Put piece back

}

}

}

if (KnightMove2(X, Y, turn, WhitePromo, BlackPromo, // Knight2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteKnight2.X; // Piece's current X

int pieceY = WhiteKnight2.Y; // Piece's current Y

WhiteKnight2.X = X; // New piece position

WhiteKnight2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteKnight2.X = pieceX; // Put piece back

WhiteKnight2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteKnight2.X = pieceX; // Put piece back

WhiteKnight2.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackKnight2.X; // Piece's current X

int pieceY = BlackKnight2.Y; // Piece's current Y

BlackKnight2.X = X; // New piece position

BlackKnight2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackKnight2.X = pieceX; // Put piece back

BlackKnight2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackKnight2.X = pieceX; // Put piece back

BlackKnight2.Y = pieceY; // Put piece back

}

}

}

if (RookMove1(X, Y, turn, WhitePromo, BlackPromo, // Rook1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteRook1.X; // Piece's current X

int pieceY = WhiteRook1.Y; // Piece's current Y

WhiteRook1.X = X; // New piece position

WhiteRook1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteRook1.X = pieceX; // Put piece back

WhiteRook1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteRook1.X = pieceX; // Put piece back

WhiteRook1.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackRook1.X; // Piece's current X

int pieceY = BlackRook1.Y; // Piece's current Y

BlackRook1.X = X; // New piece position

BlackRook1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackRook1.X = pieceX; // Put piece back

BlackRook1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackRook1.X = pieceX; // Put piece back

BlackRook1.Y = pieceY; // Put piece back

}

}

}

if (RookMove2(X, Y, turn, WhitePromo, BlackPromo, // Rook2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhiteRook2.X; // Piece's current X

int pieceY = WhiteRook2.Y; // Piece's current Y

WhiteRook2.X = X; // New piece position

WhiteRook2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhiteRook2.X = pieceX; // Put piece back

WhiteRook2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhiteRook2.X = pieceX; // Put piece back

WhiteRook2.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackRook2.X; // Piece's current X

int pieceY = BlackRook2.Y; // Piece's current Y

BlackRook2.X = X; // New piece position

BlackRook2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackRook2.X = pieceX; // Put piece back

BlackRook2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackRook2.X = pieceX; // Put piece back

BlackRook2.Y = pieceY; // Put piece back

}

}

}

if (PawnMove1(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn1 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn1.X; // Piece's current X

int pieceY = WhitePawn1.Y; // Piece's current Y

WhitePawn1.X = X; // New piece position

WhitePawn1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn1.X = pieceX; // Put piece back

WhitePawn1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn1.X = pieceX; // Put piece back

WhitePawn1.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn1.X; // Piece's current X

int pieceY = BlackPawn1.Y; // Piece's current Y

BlackPawn1.X = X; // New piece position

BlackPawn1.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn1.X = pieceX; // Put piece back

BlackPawn1.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn1.X = pieceX; // Put piece back

BlackPawn1.Y = pieceY; // Put piece back

}

}

}

if (PawnMove2(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn2 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn2.X; // Piece's current X

int pieceY = WhitePawn2.Y; // Piece's current Y

WhitePawn2.X = X; // New piece position

WhitePawn2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn2.X = pieceX; // Put piece back

WhitePawn2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn2.X = pieceX; // Put piece back

WhitePawn2.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn2.X; // Piece's current X

int pieceY = BlackPawn2.Y; // Piece's current Y

BlackPawn2.X = X; // New piece position

BlackPawn2.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn2.X = pieceX; // Put piece back

BlackPawn2.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn2.X = pieceX; // Put piece back

BlackPawn2.Y = pieceY; // Put piece back

}

}

}

if (PawnMove3(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn3 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn3.X; // Piece's current X

int pieceY = WhitePawn3.Y; // Piece's current Y

WhitePawn3.X = X; // New piece position

WhitePawn3.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn3.X = pieceX; // Put piece back

WhitePawn3.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn3.X = pieceX; // Put piece back

WhitePawn3.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn3.X; // Piece's current X

int pieceY = BlackPawn3.Y; // Piece's current Y

BlackPawn3.X = X; // New piece position

BlackPawn3.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn3.X = pieceX; // Put piece back

BlackPawn3.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn3.X = pieceX; // Put piece back

BlackPawn3.Y = pieceY; // Put piece back

}

}

}

if (PawnMove4(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn4 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn4.X; // Piece's current X

int pieceY = WhitePawn4.Y; // Piece's current Y

WhitePawn4.X = X; // New piece position

WhitePawn4.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn4.X = pieceX; // Put piece back

WhitePawn4.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn4.X = pieceX; // Put piece back

WhitePawn4.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn4.X; // Piece's current X

int pieceY = BlackPawn4.Y; // Piece's current Y

BlackPawn4.X = X; // New piece position

BlackPawn4.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn4.X = pieceX; // Put piece back

BlackPawn4.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn4.X = pieceX; // Put piece back

BlackPawn4.Y = pieceY; // Put piece back

}

}

}

if (PawnMove5(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn5 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn5.X; // Piece's current X

int pieceY = WhitePawn5.Y; // Piece's current Y

WhitePawn5.X = X; // New piece position

WhitePawn5.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn5.X = pieceX; // Put piece back

WhitePawn5.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn5.X = pieceX; // Put piece back

WhitePawn5.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn5.X; // Piece's current X

int pieceY = BlackPawn5.Y; // Piece's current Y

BlackPawn5.X = X; // New piece position

BlackPawn5.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn5.X = pieceX; // Put piece back

BlackPawn5.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn5.X = pieceX; // Put piece back

BlackPawn5.Y = pieceY; // Put piece back

}

}

}

if (PawnMove6(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn6 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn6.X; // Piece's current X

int pieceY = WhitePawn6.Y; // Piece's current Y

WhitePawn6.X = X; // New piece position

WhitePawn6.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn6.X = pieceX; // Put piece back

WhitePawn6.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn6.X = pieceX; // Put piece back

WhitePawn6.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn6.X; // Piece's current X

int pieceY = BlackPawn6.Y; // Piece's current Y

BlackPawn6.X = X; // New piece position

BlackPawn6.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn6.X = pieceX; // Put piece back

BlackPawn6.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn6.X = pieceX; // Put piece back

BlackPawn6.Y = pieceY; // Put piece back

}

}

}

if (PawnMove7(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn7 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn7.X; // Piece's current X

int pieceY = WhitePawn7.Y; // Piece's current Y

WhitePawn7.X = X; // New piece position

WhitePawn7.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn7.X = pieceX; // Put piece back

WhitePawn7.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn7.X = pieceX; // Put piece back

WhitePawn7.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn7.X; // Piece's current X

int pieceY = BlackPawn7.Y; // Piece's current Y

BlackPawn7.X = X; // New piece position

BlackPawn7.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn7.X = pieceX; // Put piece back

BlackPawn7.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn7.X = pieceX; // Put piece back

BlackPawn7.Y = pieceY; // Put piece back

}

}

}

if (PawnMove8(X, Y, turn, false, enPassantWhite, enPassantBlack, WhitePromo, BlackPromo, // Pawn8 movement

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

if (IsEven(turn) == false) // White's turn

{

int pieceX = WhitePawn8.X; // Piece's current X

int pieceY = WhitePawn8.Y; // Piece's current Y

WhitePawn8.X = X; // New piece position

WhitePawn8.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePawn8.X = pieceX; // Put piece back

WhitePawn8.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePawn8.X = pieceX; // Put piece back

WhitePawn8.Y = pieceY; // Put piece back

}

}

else // Black's turn

{

int pieceX = BlackPawn8.X; // Piece's current X

int pieceY = BlackPawn8.Y; // Piece's current Y

BlackPawn8.X = X; // New piece position

BlackPawn8.Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPawn8.X = pieceX; // Put piece back

BlackPawn8.Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPawn8.X = pieceX; // Put piece back

BlackPawn8.Y = pieceY; // Put piece back

}

}

}

// New piece movement

if (IsEven(turn) == false) // White's turn

{

for (int i = 0; i < WhitePromo.Length; i++)

{

if (NewPieceMove(X, Y, turn, WhitePromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

int pieceX = WhitePromo[i].X; // Piece's current X

int pieceY = WhitePromo[i].Y; // Piece's current Y

WhitePromo[i].X = X; // New piece position

WhitePromo[i].Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, WhiteKing.X, WhiteKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

WhitePromo[i].X = pieceX; // Put piece back

WhitePromo[i].Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

WhitePromo[i].X = pieceX; // Put piece back

WhitePromo[i].Y = pieceY; // Put piece back

}

}

}

}

else // Black's turn

{

for (int i = 0; i < BlackPromo.Length; i++)

{

if (NewPieceMove(X, Y, turn, BlackPromo[i], WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == true)

{

int pieceX = BlackPromo[i].X; // Piece's current X

int pieceY = BlackPromo[i].Y; // Piece's current Y

BlackPromo[i].X = X; // New piece position

BlackPromo[i].Y = Y; // New piece position

if (InCheck(turn, enPassantWhite, enPassantBlack, BlackKing.X, BlackKing.Y, false, WhitePromo, BlackPromo,

WhiteKing, WhiteQueen, WhiteBishop1, WhiteBishop2, WhiteKnight1, WhiteKnight2, WhiteRook1, WhiteRook2,

WhitePawn1, WhitePawn2, WhitePawn3, WhitePawn4, WhitePawn5, WhitePawn6, WhitePawn7, WhitePawn8,

BlackKing, BlackQueen, BlackBishop1, BlackBishop2, BlackKnight1, BlackKnight2, BlackRook1, BlackRook2,

BlackPawn1, BlackPawn2, BlackPawn3, BlackPawn4, BlackPawn5, BlackPawn6, BlackPawn7, BlackPawn8) == false)

{

BlackPromo[i].X = pieceX; // Put piece back

BlackPromo[i].Y = pieceY; // Put piece back

return false; // Not checkmate

}

else

{

BlackPromo[i].X = pieceX; // Put piece back

BlackPromo[i].Y = pieceY; // Put piece back

}

}

}

}

}

}

return true; // Checkmate

}

return false;

}

static int Points(int turn, NewPiece[] WhitePromo, NewPiece[] BlackPromo, // Points (Based on Captured Pieces)

King WhiteKing, Queen WhiteQueen, Bishop WhiteBishop1, Bishop WhiteBishop2, Knight WhiteKnight1, Knight WhiteKnight2, Rook WhiteRook1, Rook WhiteRook2,

Pawn WhitePawn1, Pawn WhitePawn2, Pawn WhitePawn3, Pawn WhitePawn4, Pawn WhitePawn5, Pawn WhitePawn6, Pawn WhitePawn7, Pawn WhitePawn8,

King BlackKing, Queen BlackQueen, Bishop BlackBishop1, Bishop BlackBishop2, Knight BlackKnight1, Knight BlackKnight2, Rook BlackRook1, Rook BlackRook2,

Pawn BlackPawn1, Pawn BlackPawn2, Pawn BlackPawn3, Pawn BlackPawn4, Pawn BlackPawn5, Pawn BlackPawn6, Pawn BlackPawn7, Pawn BlackPawn8)

{

int points = 0;

if (IsEven(turn) == false) // White's turn

{

if (BlackQueen.Captured == true)

{

points = points + 9;

}

if (BlackBishop1.Captured == true)

{

points = points + 3;

}

if (BlackBishop2.Captured == true)

{

points = points + 3;

}

if (BlackKnight1.Captured == true)

{

points = points + 3;

}

if (BlackKnight2.Captured == true)

{

points = points + 3;

}

if (BlackRook1.Captured == true)

{

points = points + 5;

}

if (BlackRook2.Captured == true)

{

points = points + 5;

}

if (BlackPawn1.Captured == true)

{

points = points + 1;

}

if (BlackPawn2.Captured == true)

{

points = points + 1;

}

if (BlackPawn3.Captured == true)

{

points = points + 1;

}

if (BlackPawn4.Captured == true)

{

points = points + 1;

}

if (BlackPawn5.Captured == true)

{

points = points + 1;

}

if (BlackPawn6.Captured == true)

{

points = points + 1;

}

if (BlackPawn7.Captured == true)

{

points = points + 1;

}

if (BlackPawn8.Captured == true)

{

points = points + 1;

}

for (int i = 0; i < BlackPromo.Length; i++)

{

if (BlackPromo[i].Captured == true)

{

switch (BlackPromo[i].Type)

{

case "queen":

points = points + 9;

break;

case "bishop":

points = points + 3;

break;

case "knight":

points = points + 3;

break;

case "rook":

points = points + 5;

break;

}

}

}

}

else // Black's turn

{

if (WhiteQueen.Captured == true)

{

points = points + 9;

}

if (WhiteBishop1.Captured == true)

{

points = points + 3;

}

if (WhiteBishop2.Captured == true)

{

points = points + 3;

}

if (WhiteKnight1.Captured == true)

{

points = points + 3;

}

if (WhiteKnight2.Captured == true)

{

points = points + 3;

}

if (WhiteRook1.Captured == true)

{

points = points + 5;

}

if (WhiteRook2.Captured == true)

{

points = points + 5;

}

if (WhitePawn1.Captured == true)

{

points = points + 1;

}

if (WhitePawn2.Captured == true)

{

points = points + 1;

}

if (WhitePawn3.Captured == true)

{

points = points + 1;

}

if (WhitePawn4.Captured == true)

{

points = points + 1;

}

if (WhitePawn5.Captured == true)

{

points = points + 1;

}

if (WhitePawn6.Captured == true)

{

points = points + 1;

}

if (WhitePawn7.Captured == true)

{

points = points + 1;

}

if (WhitePawn8.Captured == true)

{

points = points + 1;

}

for (int i = 0; i < WhitePromo.Length; i++)

{

if (WhitePromo[i].Captured == true)

{

switch (WhitePromo[i].Type)

{

case "queen":

points = points + 9;

break;

case "bishop":

points = points + 3;

break;

case "knight":

points = points + 3;

break;

case "rook":

points = points + 5;

break;

}

}

}

}

return points; // Total Points

}

// Piece Classes

public class King // King Class

{

public int X = 5;

public int Y;

public bool HasMoved = false;

}

public class Queen // Queen Class

{

public int X = 4;

public int Y;

public bool Captured = false;

}

public class Bishop // Bishop Class

{

public int X;

public int Y;

public bool Captured = false;

}

public class Knight // Knight Class

{

public int X;

public int Y;

public bool Captured = false;

}

public class Rook // Rook Class

{

public int X;

public int Y;

public bool HasMoved = false;

public bool Captured = false;

}

public class Pawn // Pawn Class

{

public int X;

public int Y;

public bool Captured = false;

public string Promotion = "pawn";

}

public class NewPiece // New Piece Class

{

public int X = 0;

public int Y = 0;

public bool Captured = false;

public string Tag = "";

public string Type = "none";

}

}

}