class ChessBoard:

def \_\_init\_\_(self):

self.board = self.setup\_board()

self.current\_player = "white" # Initialize with white player

self.winner = None

def setup\_board(self):

board = [

["r", "n", "b", "q", "k", "b", "n", "r"],

["p", "p", "p", "p", "p", "p", "p", "p"],

[" ", " ", " ", " ", " ", " ", " ", " "],

[" ", " ", " ", " ", " ", " ", " ", " "],

[" ", " ", " ", " ", " ", " ", " ", " "],

[" ", " ", " ", " ", " ", " ", " ", " "],

["P", "P", "P", "P", "P", "P", "P", "P"],

["R", "N", "B", "Q", "K", "B", "N", "R"]

]

return board

def is\_checkmate(self):

# Check if the game is in a checkmate state

pass

def get\_legal\_moves(self, piece, row, col):

legal\_moves = []

# Logic to generate legal moves for the piece at position (row, col)

return legal\_moves

def make\_move(self, move):

start\_row, start\_col, end\_row, end\_col = move

piece = self.board[start\_row][start\_col]

self.board[start\_row][start\_col] = " "

self.board[end\_row][end\_col] = piece

self.current\_player = "white" if self.current\_player == "black" else "black"

def evaluate\_board(self):

evaluation = 0

# Logic to evaluate the current board state

return evaluation

class ChessAI:

def \_\_init\_\_(self, color):

self.color = color

def choose\_move(self, board):

# For now, just return a random move

legal\_moves = self.generate\_legal\_moves(board)

return legal\_moves[0] if legal\_moves else None

def generate\_legal\_moves(self, board):

legal\_moves = []

# Logic to generate legal moves for the AI's color

return legal\_moves

def main():

# Initialize the chessboard

board = ChessBoard()

# Initialize AI players

white\_ai = ChessAI("white")

black\_ai = ChessAI("black")

# Game loop

while not board.is\_checkmate():

# White player's turn

if board.current\_player == "white":

move = white\_ai.choose\_move(board)

# Black player's turn

else:

move = black\_ai.choose\_move(board)

# Make the chosen move

if move:

board.make\_move(move)

# Print the current board state

print\_board(board)

# Print the winner

print("Game over! Winner: ", board.winner)

def print\_board(board):

for row in board.board:

print(" ".join(row))

if \_\_name\_\_ == "\_\_main\_\_":

main()