**14. Write the python program to implement Apha & Beta pruning algorithm for gaming**

**Program**: import math

# Initialize the board

board = [[" " for \_ in range(3)] for \_ in range(3)]

def print\_board():

print("\n")

for row in board:

print(" | ".join(row))

print("-" \* 9)

def is\_winner(player):

# Check rows, columns, diagonals

for i in range(3):

if all(board[i][j] == player for j in range(3)) or \

all(board[j][i] == player for j in range(3)):

return True

if all(board[i][i] == player for i in range(3)) or \

all(board[i][2 - i] == player for i in range(3)):

return True

return False

def is\_draw():

return all(cell != " " for row in board for cell in row)

def alphabeta(depth, alpha, beta, is\_maximizing):

if is\_winner("O"):

return 1

if is\_winner("X"):

return -1

if is\_draw():

return 0

if is\_maximizing:

max\_eval = -math.inf

for i in range(3):

for j in range(3):

if board[i][j] == " ":

board[i][j] = "O"

eval = alphabeta(depth + 1, alpha, beta, False)

board[i][j] = " "

max\_eval = max(max\_eval, eval)

alpha = max(alpha, eval)

if beta <= alpha:

break

return max\_eval

else:

min\_eval = math.inf

for i in range(3):

for j in range(3):

if board[i][j] == " ":

board[i][j] = "X"

eval = alphabeta(depth + 1, alpha, beta, True)

board[i][j] = " "

min\_eval = min(min\_eval, eval)

beta = min(beta, eval)

if beta <= alpha:

break

return min\_eval

def best\_move():

best\_score = -math.inf

move = (-1, -1)

for i in range(3):

for j in range(3):

if board[i][j] == " ":

board[i][j] = "O"

score = alphabeta(0, -math.inf, math.inf, False)

board[i][j] = " "

if score > best\_score:

best\_score = score

move = (i, j)

return move

def play\_game():

print("Welcome to Tic Tac Toe with Alpha-Beta Pruning!")

print("You are X, AI is O.")

print\_board()

while True:

# Human move

while True:

try:

row = int(input("Enter row (0-2): "))

col = int(input("Enter col (0-2): "))

if board[row][col] == " ":

board[row][col] = "X"

break

else:

print("Cell already taken!")

except:

print("Invalid input. Try again.")

print\_board()

if is\_winner("X"):

print("🎉 You win!")

break

if is\_draw():

print("It's a draw!")

break

# AI move

print("AI is making a move...")

i, j = best\_move()

board[i][j] = "O"

print\_board()

if is\_winner("O"):

print("🤖 AI wins!")

break

if is\_draw():

print("It's a draw!")

break

# Start the game

play\_game()

**output:**

