EX.N0: 11

ALPHA-BETAPRUNING

DATE: 22/05/24

AIM:

To implement the Alpha-Beta Pruning using python Program.

ALGORITHM:

Step 1: Start with the current game state, player, depth, alpha (initially negative infinity), and

beta (initially positive infinity).

Step 2: If the game is over, return the utility value.

Step 3: For the maximizing player, explore moves and update alpha while pruning when beta is

less than or equal to alpha.

Step 4: For the minimizing player, explore moves and update beta while pruning when alpha is

greater than or equal to beta.

Step 5: Make an initial call with the current player, depth 0, -∞ for alpha, and +∞ for beta to find

the best move.

Step 6: Recursively explore and prune branches based on the alpha and beta values to optimize

the search.

Step 7: Return the utility value of the best move for the current player at the initial call.

Program:

MAX, MIN = 1000, -1000

def minimax(depth, nodeIndex, maximizingPlayer, values, alpha, beta):

if depth == 3:

return values[nodeIndex]

if maximizingPlayer:

best = MIN

for i in range(2):

val = minimax(depth + 1, nodeIndex \* 2 + i, False, values, alpha, beta)

best = max(best, val)

alpha = max(alpha, best)

if beta <= alpha:

break

return best

else:

best = MAX

for i in range(2):

val = minimax(depth + 1, nodeIndex \* 2 + i, True, values, alpha, beta)

best = min(best, val)

beta = min(beta, best)

if beta <= alpha:

break

return best

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

values = [3, 5, 6, 9, 1, 2, 0, -1]

print("The optimal value is:", minimax(0, 0, True, values, MIN, MAX))

SAMPLE I/O AND O/P:

The optimal value is : 5

OUTPUT:

The optimal value is : 5

RESULT:

Thus the experiment to solve Alpha beta pruning algorithm by using python has been executed and verified Successfully.