Exp. No:3

Date:

MINMAX Algorithm

Tic-Tac-Toe

```
import math

def minimax (board, depth, is_max):
 scores = {'x': 10, 'o': -10, 'draw': 0}
 win_conditions = [(0,1,2), (3,4,5), (6,7,8), (0,3,6),
 (1,4,7), (2,5,8), (0,4,8), (2,4,6)]

  def evaluate (b):
   for x, y, z in win-conditions:
      if b[x] = = b[y] = = b[z] != ' ':
         return scores [b(x)]
   if ' ' not in b:
      return scores ['draw']
    return None

  score = evaluate (board)
     if score is not None:
        return score

   if is_max:
      best = - math.inf
       for i in range(9):
      if board[i]== ' ':
         board [i] = 'x'
```

```
            best = max (best, minimax (board, depth + 1,
                                                False))
            board [i] = ' '

        return best

    else:

        best = math.inf
        for i in range (9):
            if board [i] = = ' ':
                board [i] = 'o'

                best = min (best, minimax (board, depth+1,
                board [i] = ' '                    True))

        return best

def find _ best _ move (board):

    best _move = -1
    best _val = - math. inf
    for i in range (9):
        if board [i] = = ' ':
            board [i] = 'x'

            move _ val = minimax (board, 0, False)
            board [i] = ' '

            if move _val > best _ val :
                best _val = mov _val
                best _move = i
    return best _ move
```

```
board = [' '] *9

Print ("Best move for player x:", find_best_
                move (board))
```

Result:

Thus, the program was executed successfully & o/p is verified.