Ex. no :

Date :

Minimax Algorithm

Aim :-

To implement MINIMAX Algorithm problem using python.

Source code

```
from math import inf as infinity
from random import choice
import platform
import time
from os import system
HUMAN = -1
COMP = +1
board = [
    [0, 0, 0],
    [0, 0, 0],
    [0, 0, 0],
]
def evaluate(state):
    if wins(state, COMP):
        score = +1
    elif wins(state, human):
        score = -1
    else
        score = 0
    return score
def wins(state, player):
    win_state = [
        [state[0][0], state[0][1], state[0][2],
        [state[1][0], state[1][1], state[1][2],
        [state[2][0], state[2][1], state[2][2],
        [state[0][0], state[1][0], state[2][0],
        [state[0][1], state[1][1], state[2][1],
```

```
    [state[0][2], state[1][2], state[2][2]],
    [state[0][0], state[1][1], state[2][2]],
    [state[2][0], state[1][1], state[0][2]],
]
if [player, player, player] in win_state:
    return True
else
    return False
def game_over(state):
    return wins(state, HUMAN) or wins(state, COMP)
def empty_cells(state):
    cells = []
    for x, row in enumerate(state):
        for y, cell in enumerate(row):
            if cell == 0:
                cells.append([x,y])
    return cells
def valid_move(x,y):
    if [x,y] in empty_cells(board):
        return True
    else:
        return False
def set_move(x,y, player):
    if valid_move(x,y):
        board[x][y] = player
        return True
    else
        return False
def minimax(state, depth, player):
    if player == COMP:
        best = [-1,-1,-infinity]
    else
        best = [-1,-1,+infinity]
```

```
if depth == 0 or game_over(state):
    score = evaluate(state)
    return [-1, -1, score]
for cell in empty_cells(state):
    x, y = cell[0], cell[1]
    state[x][y] = player
    score = minimax(state, depth - 1, -player)
    state[x][y] = 0
    score[0], score[1] = x, y
    if player == COMP:
        if score[2] > best[2]:
            best = score
    else
        best = score
return best
def clean():
```

```
Initial board:
X | O | X
---------
O | X |
---------
  |   | O
---------

Board after PLAYER_X's best move:
X | O | X
---------
O | X |
---------
X |   | O
---------
```

Result

Thus minmax algorithm is executed and output is verified