Exp. No. 5

Date: Minimax Algorithm

Aim:

To implement minimax algorithm.

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 eval(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]]

if [player, player, player] in win_state:

    return true:
else
    return false

def game_over (state):
    return wins (state, human) or wins (state, ai)

def ec (state):
    cells = []

for x, row in enumerate (state):
for y, cell in enumerate (x row):

    if cell == 0:
        cells . append([x, y])
    return cells

        def valid_move (x, y):
            return cells

def valid_move (x, y):
    if [x, y] in empty ec (board):
        return true.
    else.
        return false.
```

```
def render (state, c_choice, h_choice):
    chars = 3

    -1 : h_choice,
    +1 : c_choice,
    0 : ' '}

    str_line = "  -  -  -  "
    print ('  ' + str_line)

    for row in state:
        for cell in row:
            symbol = chars[cell]
            print ('|' + symbol + ' ', end=" ")
        print ("|\n" + str_line)

def ai_turn (c_choice, h_choice):
    depth = len (ee(board))
    if depth == 0 or game_over (board):
        return
clear ().
```