## * Min-Max Algorithm for Tic-Tac-Toe

```
board = [[' ', ' ', ' '], [' ', ' ', ' '], [' ', ' ', ' ']]

function print_board (board):
    for row in board:
        print row

function check_winner (board)
    for row in board:
        if row[0] == row[1] == row[2]
        & row[0] != ' ';
            return row[0]

    for col in range(3):
        if board[0][col] == board[1][col]
        == board[2][col] and
            board[0][col] != ' ';
        return board[0][col].

    if board[0][0] == board[1][1] == board
    [2][2] and board[0][0] != ' ';
    return board[0][0]

    if board[0][2] == board[1][1] == b[2][0]
        return b[0][2]

    return None.
```

```
def is_full(board):
    for row in board:
        if ' ' in row:
            return False
    return True

def minimax(board, depth, is_maxi):
    win = check_winner(board)
    if win == 'X':
        return 10 - depth
    elif win == 'O':
        return depth - 10
    elif is_full(board):
        return 0

if is_maxi:
    best_score = float('-inf')
    for i in range(3):
        for j in range(3):
            if board[i][j] == ' ':
                board[i][j] = 'X'
                score = minimax(board,
                    depth+1, False)
                best_score = max(best_score,
                    score),

    return best_score
else:
    best_score = float('inf')
    for i in range(3):
        for j in range(3):

if board[i][j] == 'X':
    board[i][j] = 'O'
```

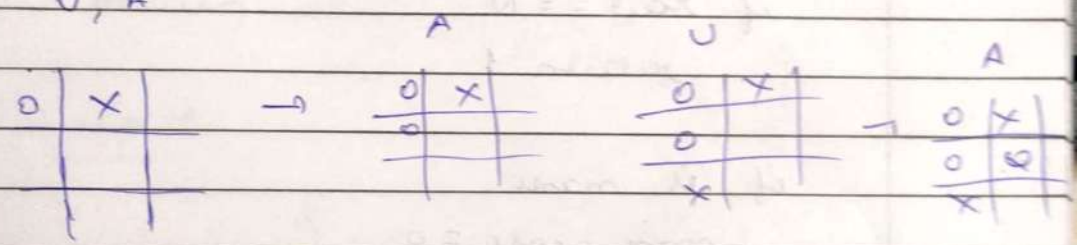
Date __/__/__
Page _____

score = min(b, d+1, +inf)
return best_score

find best move()
for each empty cell
if score > best_score
best_score = score
best_move = (i, j)

Output:

User - O AI - X




Now AI can minimize user's score.