ARTIFICIAL INTELLIGENCE

ROLL NO. : 18BCE191
DATE: 15/11/2021

Aim

Implement tic-tac-toe using Minimax Algorithm

Code

```
#include<bits/stdc++.h>
#define matrix vector<vector<char>>

using namespace std;

char player = 'X', oppo = 'O', none = '_';
pair<int,int> ended = {-1,-1};

bool check(matrix mat, char ch) {
 bool won = false;
 won |= mat[0][0] == ch && mat[1][1] == ch && mat[2][2] == ch;
 won |= mat[0][2] == ch && mat[1][1] == ch && mat[2][0] == ch;
 for(int i=0;i<3;i++) {
   won |= mat[0][i] == ch && mat[1][i] == ch && mat[2][i] == ch;
   won |= mat[i][0] == ch && mat[i][1] == ch && mat[i][2] == ch;
 }
 return won;
}

int winner(matrix mat) {
 if(check(mat, player)) return 10;
 if(check(mat, oppo)) return -10;
 return 0;
}

vector<pair<int,int>> remainingMoves(matrix board) {
```

```
 vector<pair<int,int>> vp;
 for(int i=0;i<board.size();i++) {
   for(int j=0;j<board[i].size();j++) {
     if(board[i][j] == none) vp.emplace_back(i,j);
   }
 }
 return vp;
}


bool movesLeft(matrix board) {
 return remainingMoves(board).size() > 0;
}

bool draw(matrix board) {
 return winner(board) == 0 && !movesLeft(board);
}


int minmax(matrix board, int depth, bool isMaximizing) {
 int score = winner(board);

 // someone has won
 if(score != 0) return score;

 // if draw
 if(draw(board)) return 0;

 int bestScore = isMaximizing ? INT_MIN : INT_MAX;
 for(auto move: remainingMoves(board)) {
   board[move.first][move.second] = isMaximizing ? player : oppo;
   score = minmax(board, depth+1, !isMaximizing);
   if(isMaximizing) score -= depth;
   else score += depth;
   board[move.first][move.second] = none;
   bestScore = isMaximizing ? max(score, bestScore) : min(score,
bestScore);
 }
 return bestScore;
}
```

```
pair<int,int> bestMove(matrix board, bool isMaximizing = true) {
 int score = winner(board);

 // someone has won or game drawn
 if(score != 0 || draw(board)) return ended;

 pair<int, int> toMove = ended;
 int bestScore = isMaximizing? INT_MIN : INT_MAX;
 for(auto move: remainingMoves(board)) {
   board[move.first][move.second] = isMaximizing ? player : oppo;
   score = minmax(board, 0, !isMaximizing);
   board[move.first][move.second] = none;
   if(isMaximizing && score > bestScore) {
     bestScore = score;
     toMove = move;
   }
   if(!isMaximizing && score < bestScore) {
     bestScore = score;
     toMove = move;
   }
 }
 return toMove;
}

matrix applyMove(matrix board, pair<int,int> move, bool ai) {
 board[move.first][move.second] = ai ? oppo : player;
 return board;
}

void printMove(matrix board, bool AiTurn) {
 cout<<"\n\n";
 for(int i=0;i<board.size();i++) {
   for(int j=0;j<board[i].size();j++) {
     cout<<board[i][j] << " ";
   }
   cout<<endl;
 }
 cout<< (AiTurn ? oppo :player)<<"'s Turn\n\n";
}
```

```cpp
int main() {
 matrix board(3, vector<char> (3));
 cout<<"enter the input : "<<endl;
 for(int i=0;i<3;i++) {
   for(int j=0;j<3;j++) {
     cin>>board[i][j];
   }
 }
 char turn;
 cout<<"Enter the player with 1st turn ? ";
 cin>>turn;
 bool AIsTurn = turn == oppo ? true :false;

 pair<int,int> move;
 while((move = bestMove(board, !AIsTurn))!=ended) {
   board = applyMove(board, move, AIsTurn);
   printMove(board, AIsTurn);
   AIsTurn = !AIsTurn;
 }
 return 0;
}
```

Output

```
(base) rajat@rajat-VivoBook-S14-X430UA:/Rajat1/Books/Artificial Intelligence/Practicals$ g++ MinMax\ algo.cpp -o MinMax\
 algo
(base) rajat@rajat-VivoBook-S14-X430UA:/Rajat1/Books/Artificial Intelligence/Practicals$ ./MinMax\ algo
Enter the initial input :
_ _ _
_ _ X
_ X X
Enter the player with 1st turn ? O


O _ _
_ _ X
_ X X
O's Turn



O _ X
_ _ X
_ X X
X's Turn

(base) rajat@rajat-VivoBook-S14-X430UA:/Rajat1/Books/Artificial Intelligence/Practicals$
```

```
(base) rajat@rajat-VivoBook-S14-X430UA:/Rajat1/Books/Artificial Intelligence/Practicals$ ./MinMax\ algo
Enter the initial input :

_ _ _
_ _ 0
_ _ _
Enter the player with 1st turn ? X


_ _ X
_ _ 0
_ _ _
X's Turn



0 _ X
_ _ 0
_ _ _
0's Turn



0 _ X
X _ 0
_ _ _
X's Turn
```

```
0 0 X
X _ 0
_ _ _
0's Turn



0 0 X
X X 0
_ _ _
X's Turn



0 0 X
X X 0
0 _ _
0's Turn



0 0 X
X X 0
0 X _
X's Turn
```

```
0 0 X
X X 0
0 X 0
0's Turn

(base) rajat@rajat-VivoBook-S14-X430UA:/Rajat1/Books/Artificial Intelligence/Practicals$
```