AI Tutorial 4

Write a program to implement Single Player Game (Using Heuristic Function).

Code:-

```
#include<bits/stdc++.h>

using namespace std;


#define COMPUTER 1

#define HUMAN 2


#define SIDE 3 // Length of the board


// Computer will move with 'O'

// and human with 'X'

#define COMPUTERMOVE 'O'

#define HUMANMOVE 'X'


// A function to show the current board status

void showBoard(char board[][SIDE])

{

	printf("\n\n");


	printf("\t\t\t %c | %c | %c \n", board[0][0],

								board[0][1], board[0][2]);

	printf("\t\t\t--------------\n");

	printf("\t\t\t %c | %c | %c \n", board[1][0],

								board[1][1], board[1][2]);

	printf("\t\t\t--------------\n");

	printf("\t\t\t %c | %c | %c \n\n", board[2][0],

								board[2][1], board[2][2]);
	return;
}

// A function to show the instructions
void showInstructions()
{
	printf("\t\t\t Tic-Tac-Toe\n\n");
	printf("Choose a cell numbered from 1 to 9 as below"
			" and play\n\n");

	printf("\t\t\t 1 | 2 | 3 \n");
	printf("\t\t\t--------------\n");
	printf("\t\t\t 4 | 5 | 6 \n");
	printf("\t\t\t--------------\n");
	printf("\t\t\t 7 | 8 | 9 \n\n");

	printf("-\t-\t-\t-\t-\t-\t-\t-\t-\t-\n\n");

	return;
}



// A function to initialise the game
void initialise(char board[][SIDE], int moves[])
{
	// Initiate the random number generator so that
	// the same configuration doesn't arises
	srand(time(NULL));

	// Initially the board is empty
```

```
        for (int i=0; i<SIDE; i++)
        {
                for (int j=0; j<SIDE; j++)
                        board[i][j] = ' ';
        }

        // Fill the moves with numbers
        for (int i=0; i<SIDE*SIDE; i++)
                moves[i] = i;

        // randomise the moves
        random_shuffle(moves, moves + SIDE*SIDE);

        return;
}

// A function to declare the winner of the game
void declareWinner(int whoseTurn)
{
        if (whoseTurn == COMPUTER)
                printf("COMPUTER has won\n");
        else
                printf("HUMAN has won\n");
        return;
}

// A function that returns true if any of the row
// is crossed with the same player's move
bool rowCrossed(char board[][SIDE])
{
        for (int i=0; i<SIDE; i++)
        {
                if (board[i][0] == board[i][1] &&
                        board[i][1] == board[i][2] &&
                        board[i][0] != ' ')
                        return (true);
        }
        return(false);
}

// A function that returns true if any of the column
// is crossed with the same player's move
bool columnCrossed(char board[][SIDE])
{
        for (int i=0; i<SIDE; i++)
        {
                if (board[0][i] == board[1][i] &&
                        board[1][i] == board[2][i] &&
                        board[0][i] != ' ')
                        return (true);
        }
        return(false);
}

// A function that returns true if any of the diagonal
// is crossed with the same player's move
bool diagonalCrossed(char board[][SIDE])
{
        if (board[0][0] == board[1][1] &&
                board[1][1] == board[2][2] &&
                board[0][0] != ' ')
                return(true);
```

```
	if (board[0][2] == board[1][1] &&
		board[1][1] == board[2][0] &&
		board[0][2] != ' ')
		return(true);

	return(false);
}

// A function that returns true if the game is over
// else it returns a false
bool gameOver(char board[][SIDE])
{
	return(rowCrossed(board) || columnCrossed(board)
					|| diagonalCrossed(board) );
}

// A function to play Tic-Tac-Toe
void playTicTacToe(int whoseTurn)
{
	// A 3*3 Tic-Tac-Toe board for playing
	char board[SIDE][SIDE];

	int moves[SIDE*SIDE];

	// Initialise the game
	initialise(board, moves);

	// Show the instructions before playing
	showInstructions();
```

```
int moveIndex = 0, x, y;


// Keep playing till the game is over or it is a draw
while (gameOver(board) == false &&
                    moveIndex != SIDE*SIDE)
{
          if (whoseTurn == COMPUTER)
          {
                    x = moves[moveIndex] / SIDE;
                    y = moves[moveIndex] % SIDE;
                    board[x][y] = COMPUTERMOVE;
                    printf("COMPUTER has put a %c in cell %d\n",
                                        COMPUTERMOVE, moves[moveIndex]+1);
                    showBoard(board);
                    moveIndex ++;
                    whoseTurn = HUMAN;
          }

          else if (whoseTurn == HUMAN)
          {
                    x = moves[moveIndex] / SIDE;
                    y = moves[moveIndex] % SIDE;
                    board[x][y] = HUMANMOVE;
                    printf ("HUMAN has put a %c in cell %d\n",
                                        HUMANMOVE, moves[moveIndex]+1);
                    showBoard(board);
                    moveIndex ++;
                    whoseTurn = COMPUTER;
          }
}
```

```
        // If the game has drawn
        if (gameOver(board) == false &&
                        moveIndex == SIDE * SIDE)
                printf("It's a draw\n");
        else
        {
                // Toggling the user to declare the actual
                // winner
                if (whoseTurn == COMPUTER)
                        whoseTurn = HUMAN;
                else if (whoseTurn == HUMAN)
                        whoseTurn = COMPUTER;

                // Declare the winner
                declareWinner(whoseTurn);
        }
        return;
}

// Driver program
int main()
{
        // Let us play the game with COMPUTER starting first
        playTicTacToe(COMPUTER);

        return (0);
}
```

Output:-

Enrollment No :- 170200107066

```
$g++ -o main *.cpp
$main
                         Tic-Tac-Toe

Choose a cell numbered from 1 to 9 as below and play

                         1 | 2 | 3
                        --------------
                         4 | 5 | 6
                        --------------
                         7 | 8 | 9

-       -       -       -       -       -       -       -       -       -

COMPUTER has put a O in cell 8


                           |   |
                        --------------
                           |   |
                        --------------
                           | O |

HUMAN has put a X in cell 3


                           |   | X
                        --------------
                           |   |
                        --------------
                           | O |
```

```
COMPUTER has put a O in cell 6


                           |   | X
                        --------------
                           |   | O
                        --------------
                           | O |

HUMAN has put a X in cell 4


                           |   | X
                        --------------
                         X |   | O
                        --------------
                           | O |

COMPUTER has put a O in cell 9


                           |   | X
                        --------------
                         X |   | O
                        --------------
                           | O | O

HUMAN has put a X in cell 7


                           |   | X
                        --------------
                         X |   | O
                        --------------
                         X | O | O

COMPUTER has put a O in cell 2


                           | O | X
                        --------------
                         X |   | O
                        --------------
                         X | O | O

HUMAN has put a X in cell 1


                         X | O | X
                        --------------
                         X |   | O
                        --------------
                         X | O | O

HUMAN has won
```