Ex.No.1

Date:20/1/2021

TOY PROBLEM USING AI

AIM: To Study and Implement Tic Tac Toe Using AI

ALGORITHM:

The whole process can be categorized as follows:

1. First, we use the Function board which is used to print the Board that it was passed.

2. We then create a new function named inputPlayerLetter which used to take the input from the User. It returns the entered letter First and the Computer generated letter next.

3. The next Function that we create is whoGoesFirst(): which determines , whether the player or the Computer will move first.

4. playAgain() Function is used to determine whether the player wants to play another round.

5. Makemove() Function is used to print the Letter on the board based on the move taken by the user.

6. Given a board and a player's letter, this function returns True if that player has won we use the def isWinner Function.

7. def getPlayerMove is used to generate the players move.

8. def chooseRandomMoveFromList is used to generate the AI move which is followed after the Players input.

9. Next , Given a board and the computer's letter, determine where to move and return that move using minimax() Function.

10. def findBestMove will be used by The Computer to find the best Move to play based on the inputs given by the user.

11. def isBoardFull(board): Return True if every space on the board has been taken. Otherwise return False.
12. Print (Welcome to Tic Tac Toe!)
13. Print (Reference of numbering on the board)
14. drawBoard('0 1 2 3 4 5 6 7 8 9'.split())
15. Repeat the Steps When and If user presses Yes to play again. The process is Repeated.

CODE:

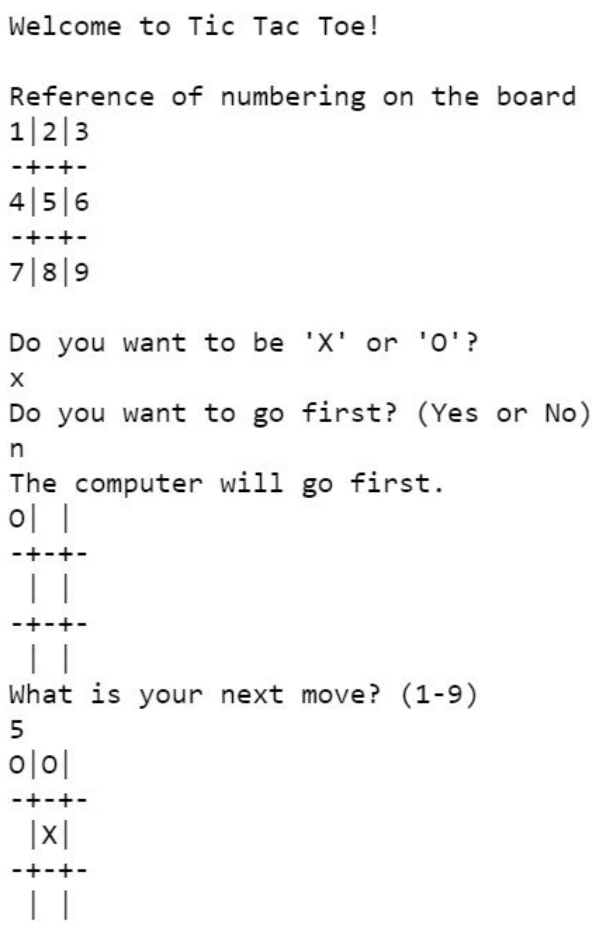
```
import random
 def
drawBoard(board):


        print(board[1] + '|' + board[2] + '|' + board[3])
print('-+-+-')
        print(board[4] + '|' + board[5] + '|' + board[6])
print('-+-+-')
        print(board[7] + '|' + board[8] + '|' + board[9])
def inputPlayerLetter():
.         letter=''
        while not(letter=='X' or letter=='O'):
                print("Do you want to be 'X' or 'O'?")
                letter = input().upper()


        if letter == 'X':
                return ['X','O']
        else:
                return ['O','X']
def whoGoesFirst():
 print('Do you want to go first? (Yes or No)')  if
input().lower().startswith('y'):
                    return
'player'            else:
          return 'computer'
          '''
          # Randomly choose the player who goes
first.    if random.randint(0,1) == 0:
return 'computer'         else:               return
'player'
          '''
def playAgain():
          print('Do you want to play again? (Yes or No)')
          return input().lower().startswith('y')



def makeMove(board, letter, move):
          board[move] = letter



def isWinner(board,letter):
          return ((board[1]==letter and board[2]==letter and board[3]==letter) or
                              (board[4]==letter and board[5]==letter and board[6]==letter) or
                              (board[7]==letter and board[8]==letter and board[9]==letter) or
(board[1]==letter and board[4]==letter and board[7]==letter) or
                              (board[2]==letter and board[5]==letter and board[8]==letter) or
                              (board[3]==letter and board[6]==letter and board[9]==letter) or
                              (board[1]==letter and board[5]==letter and board[9]==letter) or
                              (board[3]==letter and board[5]==letter and board[7]==letter))

def getBoardCopy(board):
        dupBoard = []

        for i in board:
                dupBoard.append(i)

        return dupBoard

def isSpaceFree(board, move):
        return board[move] == ' '

def getPlayerMove(board):
        # Let the player type in their move.    move = ''          while move not in '1 2
3 4 5 6 7 8 9'.split() or not isSpaceFree(board,int(move)):
                print('What is your next move? (1-9)')
                move = input()
        return int(move) def
chooseRandomMoveFromList(board, movesList):
        possibleMoves = []      for i
in movesList:              if
isSpaceFree(board, i):
possibleMoves.append(i)

        if len(possibleMoves) != 0:
                return random.choice(possibleMoves)
else:
                return None

def minimax(board, depth, isMax, alpha, beta, computerLetter):
```

```
.          if computerLetter ==
'X':                playerLetter
= 'O'    else:
                    playerLetter = 'X'
          if isWinner(board,
computerLetter):
                    return 10          if
isWinner(board, playerLetter):
                    return -10
          if
isBoardFull(board):
                    return 0

          if isMax:
                    best = -1000

                    for i in range(1,10):
                              if isSpaceFree(board, i):
                                        board[i] = computerLetter
   best = max(best, minimax(board, depth+1, not isMax, alpha, beta, computerLetter) - depth)
                                        alpha = max(alpha, best)
                                        board[i] = ' '

                                        if alpha >= beta:
                                                  break
                    return best
          else:
                    best = 1000
                    for i in range(1,10):
            if isSpaceFree(board, i):
                    board[i] = playerLetter
   best = min(best, minimax(board, depth+1, not isMax, alpha, beta, computerLetter) + depth)
                                    beta = min(beta, best)
                                    board[i] = ' '
                                    if alpha >=
beta:
                                            break
            return best

def findBestMove(board,
computerLetter):        if
computerLetter == 'X':
playerLetter = 'O'      else:
                    playerLetter = 'X'

            bestVal = -1000
bestMove = -1 for i in
range(1,10):    if
isSpaceFree(board, i):
                            board[i] = computerLetter

                            moveVal = minimax(board, 0, False, -1000, 1000, computerLetter)

                            board[i] = ' '

                            if moveVal > bestVal:
                                    bestMove = i
                            bestVal = moveVal
```

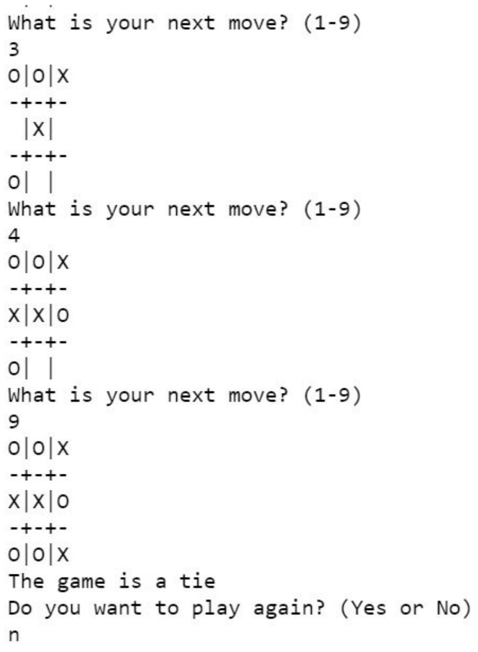
```
        return bestMove

def isBoardFull(board):          for i
in range(1,10):                  if
isSpaceFree(board, i):
                        return False
        return True

print('\nWelcome to Tic Tac Toe!\n')
print('Reference of numbering on the
board') drawBoard('0 1 2 3 4 5 6 7 8
9'.split()) print('')

while True:
        theBoard = [' '] * 10
        playerLetter, computerLetter =
        inputPlayerLetter() turn = whoGoesFirst()
        print('The ' + turn + ' will go first.')
        gameIsPlaying = True

        while gameIsPlaying:
                if turn ==
'player':
                        drawBoard(theBoard)
        move = getPlayerMove(theBoard)
makeMove(theBoard, playerLetter, move)

                        if isWinner(theBoard, playerLetter):
```

```
                                        drawBoard(theBoard)
                                        print('You won the
game')                                            gameIsPlaying
= False                                 else:
          if isBoardFull(theBoard):
                                                  drawBoard(theBoard)
                                                  print('The game is a tie')
                                                  break
                                        else:
                                                  turn = 'computer'
                    else:
                              move = findBestMove(theBoard, computerLetter)
makeMove(theBoard, computerLetter, move)

                              if isWinner(theBoard, computerLetter):
                                        drawBoard(theBoard)
                                        print('You lose the
          game')                                  gameIsPlaying
          = False                       else:
                    if isBoardFull(theBoard):
                                                  drawBoard(theBoard)
                                                  print('The game is a tie')
                                                  break
                                        else:
    turn = 'player'  if not playAgain():
                    break
```

Output:



```
Welcome to Tic Tac Toe!

Reference of numbering on the board
1|2|3
-+-+-
4|5|6
-+-+-
7|8|9

Do you want to be 'X' or 'O'?
x
Do you want to go first? (Yes or No)
n
The computer will go first.
O| |
-+-+-
 | |
-+-+-
 | |
What is your next move? (1-9)
5
O|O|
-+-+-
 |X|
-+-+-
 | |
```



```
What is your next move? (1-9)
3
O|O|X
-+-+-
 |X|
-+-+-
O| |
What is your next move? (1-9)
4
O|O|X
-+-+-
X|X|O
-+-+-
O| |
What is your next move? (1-9)
9
O|O|X
-+-+-
X|X|O
-+-+-
O|O|X
The game is a tie
Do you want to play again? (Yes or No)
n
```

Result: We have successfully studied and implemented TIC TAC TOE Using AI.