Program 1: Implement Tic –Tac –Toe Game..

PRANAV JAGADEESH 1BM180S071
Program 1 Implement Tic-Tac-Toe Game
This AI program intelligently responds to players moves. In this game two players X and O take turns by playing N 291 O. If a player gets three of their marks on the leased in a new, column or one of the
two diagnols, they win When the board fills up with neither player winning the game and in a draw

PRANAV TAGADEESH 18M18CSO71 3B Classmate TIC TAC TOE PROGRAM B2 BATCH

```
import random
def drawboard (board):
                 + board [77+ 1/ + board [8]+1/+board [9])
         1 '+ board [4] + '/' + board [5] + '/ '+ board [6]
          " '+ board[1)+ '/ + board[2]+ "/ + board[3]
 def Input Player Letter ():
   while not (letter = = (x' or letter == (o'):
     print ( 'Do you want to be x or 0?'
     ereturn Computer
   else: suturn Hayer
 def play Again ():
 print ('Do you mant to play again? (yes or no)')

return input(). lower(). startsmith ('y')

def make Mone (board, letter, mone)

board (mone) = letter

def is Winner (bo, le):
   neturn ((60(7) == le and bo (8) == le and bo (9) == le
```



```
or (bo[4] == le \text{ and } bo[5] == le \text{ and } bo[6] == le)
or (bo[1] == le \text{ and } bo[2] == le \text{ and } bo[3] == le)
 or (b_0 (7) = -le \text{ and } bO (4) = -le \text{ and } bo (1) = -le)
or (b_0 (8) = -le \text{ and } bO (5) = -le \text{ and } bo (2) = -le)
or (b_0 (9) = -le \text{ and } bO (6) = -le \text{ and } bo (3) = -le)
  or (b) [ 7] == le and bo [5] == le and bo [3] == le)
 or (6 [9] == le and 60 (5) == le and 60 (1) == le)
def get Board Copy (board) is
     dupe Board = []
     for i in board:
         dupe Board append (i)
    return dupe Board
  del is space Free (board, more)=
       neturn board [move == 1)
  def get Player Move ( board ):
     while more not in 123456787 1/2t() or not
     is Space Free (board, int (move)):
        print ('What is your next more ? (1-9)
        netuhn int (mone)
    def thouse Alandom More From List (board, monelist
        possible Mones = ()
        for i'm moueshirt:

if u Space Free (board; i):

passible Morres append(i)

if len (possible Morres) 1 = 0:

return random choice (possible Morres)
             che o return None
        def get Computer Mone (board, computer Letter):

if computer Letter = "X": player Letter = "0"

plue: player Letter = "X"
```



```
for i in sange (1,10):

copy = get Board (opy (board)

if is Space Free (copy, i):

maker More (copy, computer Letter, i)

if & Winner (copy) computer Letter):

trekum i
    for i in trange (1,10):
         cefny = get Board (opy (board)

if 'ispace Free (copy, i):

make Mone (copy, player Letter, i)

if is Winner (copy, player Letter):

neturn i
   move = choose Rein dom Move Fram List (board, [1, 3, 7, 9])
                    choackandom Monetromliet (board, [7,4,6,87)
     def is Board Full (board):
         for i in range (1,10):

if is Space Thee (board, i):

getrom false

return There
             print ('Welcome to Tic Tae Tae 1')
               player Letter, computer letter = inputplayer letter()
twon = who Goer First ()
priat ('The' + twon + 'will go first')
game Iplaying = True
```



 aulita and Tellania
 while game shlaying
 if twin = = player =
 draw Board (The Board)
 mous = get Player Move (the Board)
 inake Mane (the Board, player Letter, mone) if is Winner (the Board, player Cetter);
 of istimus the Board player cerves
print ('Hooray! You have months game!)
gane I playing = Falee)
else à
if is Board Full (the Board).
draw Brand (the Board)
print (The game is a tic!)
break
else:
Surn = 1 computer
else moul = get Computer Moue Mel Board,
comprifer Celler)
make Morre (the Board, compriter Letter, morre)
if is Winner (the Board, computer Ceffer);
draw Board (the Board)
bush (The combuter has beaten you Youtobe.)
game Is playing = False
else:
if Board Full (the Board)
deau Board (the Board)
part (The game is a fie!)
break
elie?
tuen = 'player'
if not play Again ();
if not play Again (); preak

Output:	
Waling to Tic Tay Tol	
Do you mant to be X or 0?	
The computer will go first.	
What is your ment mone?(1-9)	
1 10	
TXT	
X 1010	
0 1 1 2 -	
The game is a tie	\
The game is a tie Do you want to play again? (yes or no))
Do you want to be x or 0!	
\mathcal{V}	
The computer will go first	
XXXX	
0 0	
X 0	
The computer has beaten you! You of Do you want to Jelay again? (ye	lose.
Do you want to play again? (yo	u ou no)
Do you mant to be x or o?	
The player will go first.	
1 o Just.	

Page
o x 0 x X 0 x Hooray You ruon the game ! Do you want to play again? (yes or no) n

```
# Tic Tac Toe
import random
def drawBoard(board):
  # prints the board.
  print(' | |')
  print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])
  print(' | |')
  print('----')
  print(' | |')
  print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])
  print(' | |')
  print('----')
  print(' | |')
  print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])
  print(' | |')
def inputPlayerLetter():
  # player type which letter they want.
  letter = "
  while not (letter == 'X' or letter == 'O'):
     print('Do you want to be X or O?')
     letter = input().upper()
  # the first element is the player's letter, the second is computer's letter.
  if letter == 'X':
     return ['X', 'O']
```

```
else:
      return ['O', 'X']
def whoGoesFirst():
   # Randomly choose the player who goes first.
   if random.randint(0, 1) == 0:
      return 'computer'
   else:
      return 'player'
def playAgain():
   # returns True if player wants to play again, else returns False.
   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(bo, le):
   #True if that player has won.
   # bo instead of board and le instead of letter.
   return ((bo[7] == le \text{ and } bo[8] == le \text{ and } bo[9] == le) or #top
   (bo[4] == le \text{ and } bo[5] == le \text{ and } bo[6] == le) \text{ or } \#middle
   (bo[1] == le \text{ and } bo[2] == le \text{ and } bo[3] == le) \text{ or } \#bottom
   (bo[7] == le \text{ and } bo[4] == le \text{ and } bo[1] == le) \text{ or } \#left \text{ side}
   (bo[8] == le \text{ and } bo[5] == le \text{ and } bo[2] == le) \text{ or } \#middle
   (bo[9] == le and bo[6] == le and bo[3] == le) or #right side
   (bo[7] == le \text{ and } bo[5] == le \text{ and } bo[3] == le) \text{ or } \#diagonal
   (bo[9] == le \text{ and } bo[5] == le \text{ and } bo[1] == le)) #diagonal
```

```
def getBoardCopy(board):
  #duplicate of board list and return duplicate.
  dupeBoard = []
  for i in board:
     dupeBoard.append(i)
  return dupeBoard
def isSpaceFree(board, move):
  # Return true if passed move is free on board.
  return board[move] == ' '
def getPlayerMove(board):
  # Let player type his 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):
  # Returns a valid move.
  # Returns None if no valid move.
  possibleMoves = []
  for i in movesList:
    if isSpaceFree(board, i):
       possibleMoves.append(i)
  if len(possibleMoves) != 0:
```

```
return random.choice(possibleMoves)
  else:
    return None
def getComputerMove(board, computerLetter):
  #determine where to move and return that move.
  if computerLetter == 'X':
    playerLetter = 'O'
  else:
    playerLetter = 'X'
  #check if we can win in next move
  for i in range(1, 10):
    copy = getBoardCopy(board)
    if isSpaceFree(copy, i):
       makeMove(copy, computerLetter, i)
       if isWinner(copy, computerLetter):
         return i
  # Check if the player could winand block them.
  for i in range(1, 10):
    copy = getBoardCopy(board)
    if isSpaceFree(copy, i):
       makeMove(copy, playerLetter, i)
       if isWinner(copy, playerLetter):
         return i
  #take one of the corners, if they are free.
  move = chooseRandomMoveFromList(board, [1, 3, 7, 9])
  if move != None:
    return move
```

```
# Try to take the center, if it is free.
  if isSpaceFree(board, 5):
     return 5
  # Move on one of the sides.
  return chooseRandomMoveFromList(board, [2, 4, 6, 8])
def isBoardFull(board):
  # Return True if every space on the board is filled. Otherwise return False.
  for i in range(1, 10):
     if isSpaceFree(board, i):
       return False
  return True
print('Welcome to Tic Tac Toe!')
while True:
  # Reset the board
  theBoard = [' '] * 10
  playerLetter, computerLetter = inputPlayerLetter()
  turn = whoGoesFirst()
  print('The ' + turn + ' will go first.')
  gameIsPlaying = True
  while gameIsPlaying:
     if turn == 'player':
       # Player's turn.
       drawBoard(theBoard)
```

```
move = getPlayerMove(theBoard)
  makeMove(theBoard, playerLetter, move)
  if isWinner(theBoard, playerLetter):
    drawBoard(theBoard)
    print('Hooray! You have won the game!')
    gameIsPlaying = False
  else:
    if isBoardFull(theBoard):
       drawBoard(theBoard)
       print('The game is a tie!')
       break
    else:
       turn = 'computer'
else:
  # Computer's turn.
  move = getComputerMove(theBoard, computerLetter) \\
  makeMove(theBoard, computerLetter, move)
  if isWinner(theBoard, computerLetter):
    drawBoard(theBoard)
    print('The computer has beaten you! You lose.')
    gameIsPlaying = False
  else:
    if isBoardFull(theBoard):
       drawBoard(theBoard)
       print('The game is a tie!')
       break
    else:
```

```
turn = 'player'
```

if not playAgain():

break



