**Tic Tac Toe**

**Introduction:**

This is a detailed report of the Implementation of the Tic Tac Toe game. It is assumed that the player will play optimally. While referring cases consider symmetric moves as well.

**Case1) Player plays First(X), Computer second(O).**

Player plays odd moves, Computer even.

**Move 2**-

**About Hard-coded Moves-**The decisions have been taken after a thorough study of the cases to avoid a fork by the player.

The possible ways the player may have played are-

1. Centre b) Corner c)Edge

a)**For Centre**-Playing an edge can lead to a sure-shot loss. The Player can create a fork as shown.

?OX ?23

?XX ?15

O?? 4??

So play **any one corner**. **A Random corner is selected using function-getRandomCorner()**

1. **For Corner**- Playing an edge leads to similar fork as shown above. Playing a corner can lead to fork as well.

O?X 2?1 XOX 341

?XX ?53 ??? ???

??O ??4 O?X 2?5

So **play Centre**.

1. **For Edge-** Play any one among 2 corners adjacent to the 1st move. Other moves will lead to fork.

OXX 413 413 OXX

?X? ?5? ?52 ?XO

??O ??2 ??? ???

**Move 4,Move 6,Move 8-**

* Use minimax algorithm to find best possible move. A depth level value of recursion is subtracted/added to find shortest win move.
* Also in minimize part of minimax algorithm, since the user decides the input, if worst-case scenario is draw then the best case move (winning moves if possible) is selected.

It is demonstrated as follows-

If game moves are as follows-

??X ??1

XOO 524

??X ??3

Place 1,7 draw the game for sure. Also if user plays correctly then only Place 2,8 give a draw.

**Thus worst case is draw and best case is win at Place 2,8 if user plays incorrect.**

Thus the output is-

?OX ?61

XOO 524

??X ??3

***Author***

***Rajvaibhav Rahane***