ARTIFICIAL INTELLIGENCE

TIC TAC TOE
GAME PLAY ASSIGNMENT - REPORT

- Ananya (CS19B1004) & Yaaminie (CS19B1022)

We have adopted the strategy for the computer to always win or to not lose from https://en.wikipedia.org/wiki/Tic-tac-toe#Strategy:

A player can play a perfect game of tic-tac-toe (to win or at least, result in a draw) if each time it is their turn to play, they choose the first available move from the following list, as used in Newell and Simon's 1972 tic-tac-toe program.

1. **Win**: If the player has two in a row, they can place a third to get three in a row.
2. **Block**: If the opponent has two in a row, the player must play the third themselves to block the opponent.
3. **Fork**: Create an opportunity where the player has two ways to win (two non-blocked lines of 2).
4. **Blocking an opponent's fork**: If there is only one possible fork for the opponent, the player should block it. Otherwise, the player should block all forks in any way that simultaneously allows them to create two in a row. Otherwise, the player should create a two in a row to force the opponent into defending, as long as it doesn't result in them creating a fork. For example, if "X" has two opposite corners and "O" has the center, "O" must not play a corner in order to win. (Playing a corner in this scenario creates a fork for "X" to win.)
5. **Center**: A player marks the center. (If it is the first move of the game, playing on a corner gives the second player more opportunities to make a mistake and may therefore be the better choice; however, it makes no difference between perfect players.)
6. **Opposite corner**: If the opponent is in the corner, the player plays the opposite corner.
7. **Empty corner**: The player plays in a corner square.
8. **Empty side**: The player plays in a middle square on any of the 4 sides.

We have implemented this in the C programming language, by allowing the player to choose who can make the first move, human or the computer. By using functions in C, we have created void functions for implementing AI code (depending on which the computer will make the next move) and for displaying the updated version of the board after every move. Our code is simplified to the most basic level and we have also added comments to ensure fast understanding.