TABLE OF CONTENTS

1 INTRODUCTION:

Tic-tac-toe, also known as noughts and crosses, is a classic game played on a 3x3 grid by two players who take turns marking the spaces. The goal is to be the first to get three of your marks in a row, either horizontally, vertically, or diagonally. It's considered a zero-sum game of perfect information, which means it is deterministic, and both players always have complete knowledge of the game state. Tic-tac-toe is often used as a teaching tool in computer graphics and in artificial intelligence because of its simplicity and the optimal move can be determined using the minimax algorithm, a method for searching through the game tree to find the best possible move. The game is considered adversarial, meaning the players have opposing goals and must use strategies like minimax with alpha-beta pruning to find the best moves. Additionally, this project utilizes OpenGL and GLUT to create a graphical interface for the game, allowing users to interact with the game visually. The graphical implementation demonstrates key concepts in computer graphics, such as rendering, user interaction, and game state management, providing an engaging and educational experience for players.

2 OBJECTIVES:

The Tic Tac Toe game is designed to provide entertainment and mental stimulation for players of all ages. It aims to engage players in strategic thinking, helping them improve their decision-making skills and stay alert. The game is enjoyable and simple, making it a fun pastime. It also has educational value, as it demonstrates concepts related to artificial intelligence, particularly the minimax algorithm. The game environment is non-violent, ensuring it is suitable for everyone, including children. The interface is user-friendly, allowing players to easily understand and control the game. By playing Tic Tac Toe, players can practice making optimal decisions and improve their planning and foresight skills. This game is not just entertaining but also helps in developing important cognitive and strategic abilities.

3 SCOPE OF THIS GAME:

This report outlines all the necessary requirements for the Tic Tac Toe project. The main goal of this research is to provide a comprehensive overview of both the structured and unstructured elements involved in developing the game "Tic Tac Toe". Designed for the Windows platform, it challenges players to engage in strategic gameplay while offering two distinct modes: player vs. player and player vs. computer.

In the player vs. player mode, users can compete against each other, testing their strategic prowess and tactical decision-making. On the other hand, in the player vs. computer mode, players face off against an AI opponent, providing a challenging experience for solo players.

The game's episodic structure enhances the storytelling pace, making it more engaging, while also offering branching pathways that encourage creativity and boldness. This report demonstrates the seamless action flow between player inputs, scripting, and display outputs.

4 PROBLEM STATEMENT

Tic tac toe game has 9 cells for a 3x3 grid. The two players or 1 player with their respective marks as 'X' and 'O' are required to place their marks in their turns one by one. Once the cell is occupied by a mark it cannot be used again. The game is won if the agent is able to make a row or column or a diagonal occupied completely with their respective marks. The game terminates once the winning situation is gained or the cells are fully occupied. The problem specification for this game is given below:

Problem: Given a 3x3 grid, the agents have to find the optimal cell to fill with respective marks.

Goals: To find the optimal cell to fill with respective marks and in order to win the game, the cell must be filled such that one of the following criteria is satisfied:

1. A row is completely filled by a mark 'X' or 'O'.

2. A diagonal is completely filled by a mark 'X' or 'O'.

3. A column is completely filled by a mark 'X' or 'O'. If these criteria are not satisfied by both the players, the game is terminated with a tie situation.

Constraints:

1. Once the cell is occupied by a mark, it cannot be reused.

2. Player place the mark alternatively. So, consecutive moves from any player are not allowed.

5 ALGORITHM USED:

MiniMax Algorithm

The Minimax algorithm is a recursive method designed to select the optimal move for a player, assuming the opponent is also playing optimally. Its primary goal is to minimize the maximum potential loss. This algorithm is rooted in adversarial search techniques, where players are opponents with conflicting objectives. The MAX player determines the best initial move and then considers every possible response by the MIN player, continuing this alternation until a termination condition is met. MAX prefers moves leading to states of maximum value, while MIN aims for states with minimum value, ensuring both players play optimally. This algorithm evaluates all possible moves and their outcomes, aiming to minimize the possible loss for the worst-case scenario. In this implementation, the algorithm searches through the game tree to determine the best possible move for the computer player in the Tic Tac Toe game.

6 CODE

1. VOID INITGAME()

```
{
  playerturn = 1;
  winPosition.x1 = winPosition.y1 = -1;
  for(int i = 0; i <= 2; i++)
    for(int j = 0; j <= 2; j++)
      matrix[i][j] = 0;
}
```

2. ISMOVESLEFT()

```
{
  for (int i = 0; i<3; i++)
    for (int j = 0; j<3; j++)
      if (board[i][j] == 0)
        return true;
  return false;
}
```

3. MINIMAX()

```
{
  int score = evaluate();
  if (score == 10)
    return score - depth;
  if (score == -10)
    return score + depth;
  if (isMovesLeft(matrix) == false)
    return 0;

  if(isMax) {
    int best = -1000;
    for (int i = 0; i<3; i++)
    {
```

```
            for (int j = 0; j<3; j++)
            {
              if (matrix[i][j] == 0)
              {
                matrix[i][j] = 1;
                best = fmax(best,minimax(depth+1, !isMax));
                matrix[i][j] = 0;
              }
            }
          }
          return best;
        }
        else
        {
          int best = 1000;
          for (int i = 0; i<3; i++)
          {
            for (int j = 0; j<3; j++)
            {
              if (matrix[i][j] == 0)
              {
                matrix[i][j] = 2;
                best = fmin(best, minimax(depth+1, !isMax));
                matrix[i][j] = 0;
              }
            }
          }
          return best;
        }
      }
```

4. GETBESTMOVE()

```
   {
     int bestVal = 1000;
```

```
    Move bestMove;
    for(int i=0; i<3; i++) {
      for(int j=0; j<3; j++) {
        if(matrix[i][j] == 0) {
          matrix[i][j] = 2;
          int moveVal = minimax(0, true);
          matrix[i][j] = 0;

          if (moveVal < bestVal)
          {
            bestMove.row = i;
            bestMove.col = j;
            bestVal = moveVal;
          }
        }
      }
    }
    return bestMove;
}
```

7 OUTPUT SNAPSHOT:

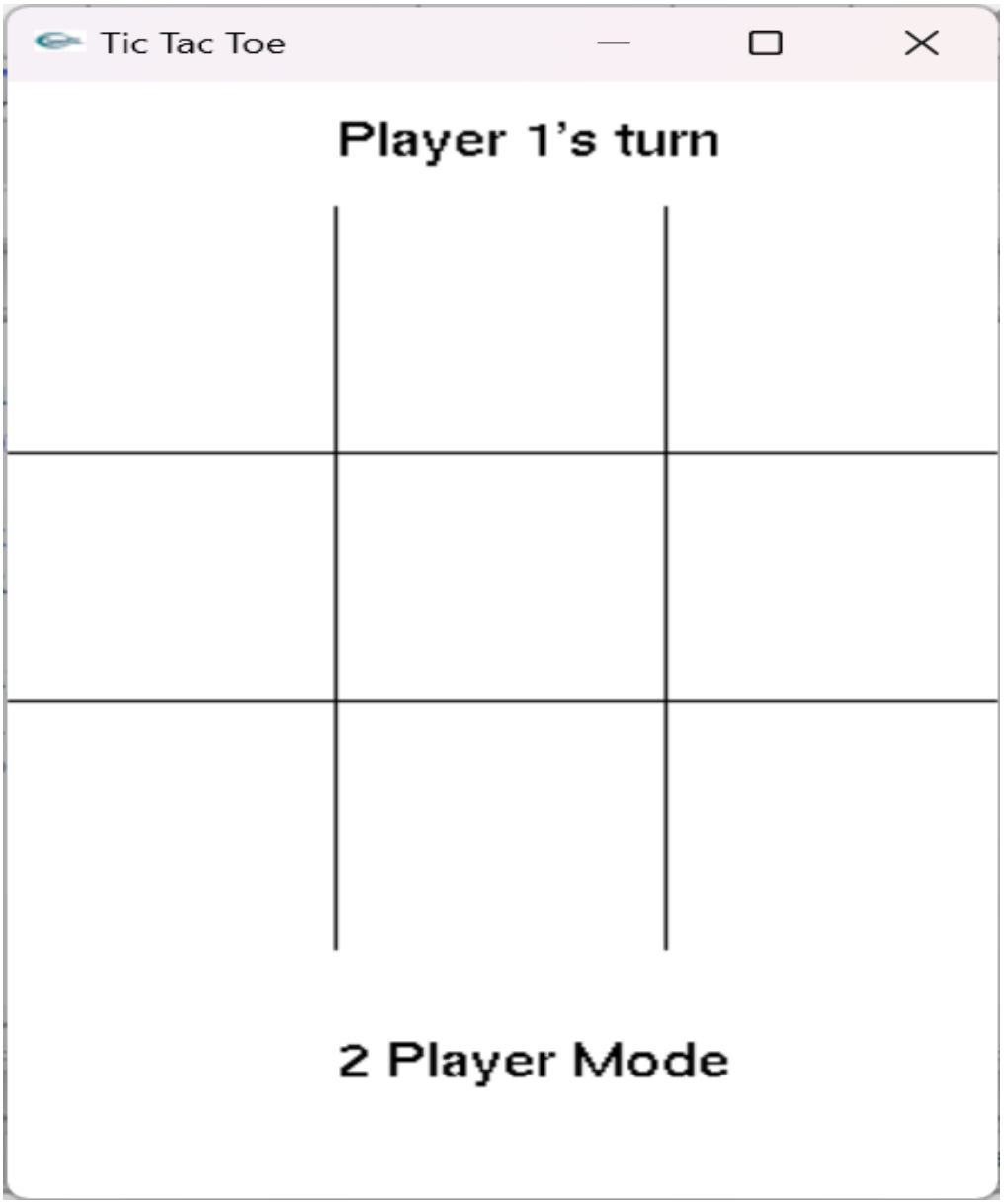

Figure 1: Tic-Tac-Toe Grid

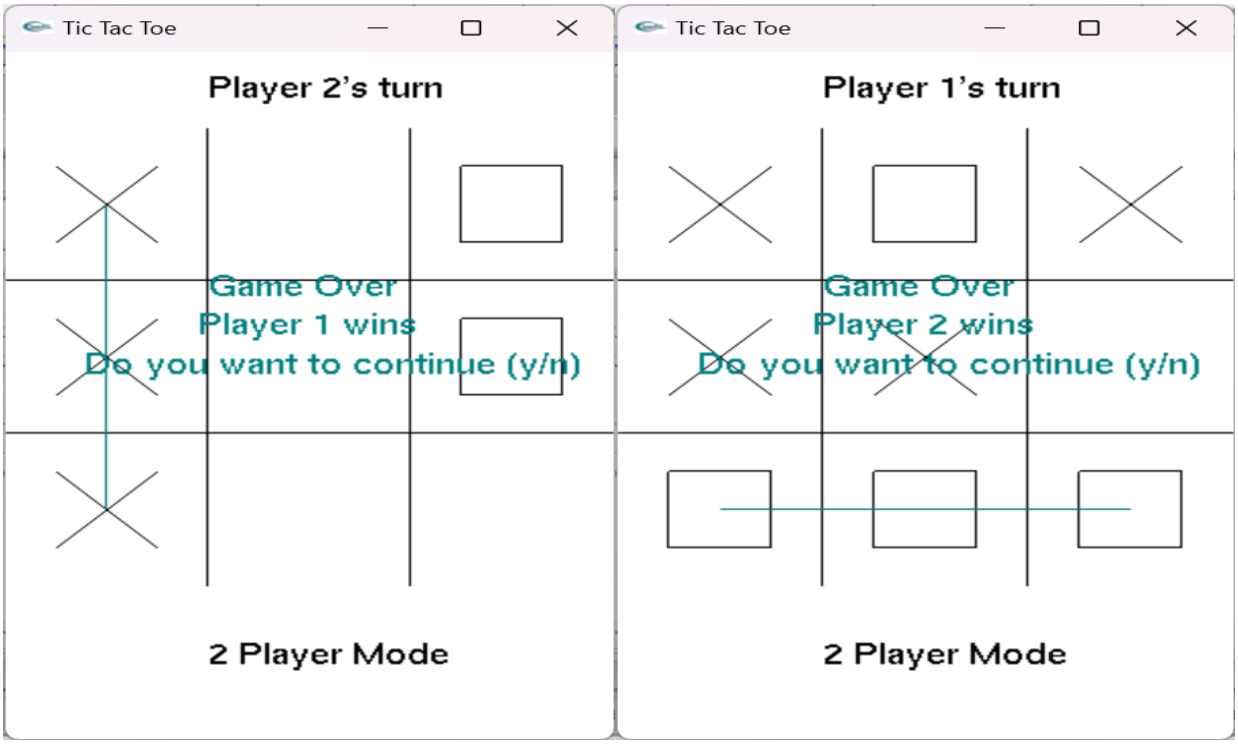

Figure 2: Player vs Player game

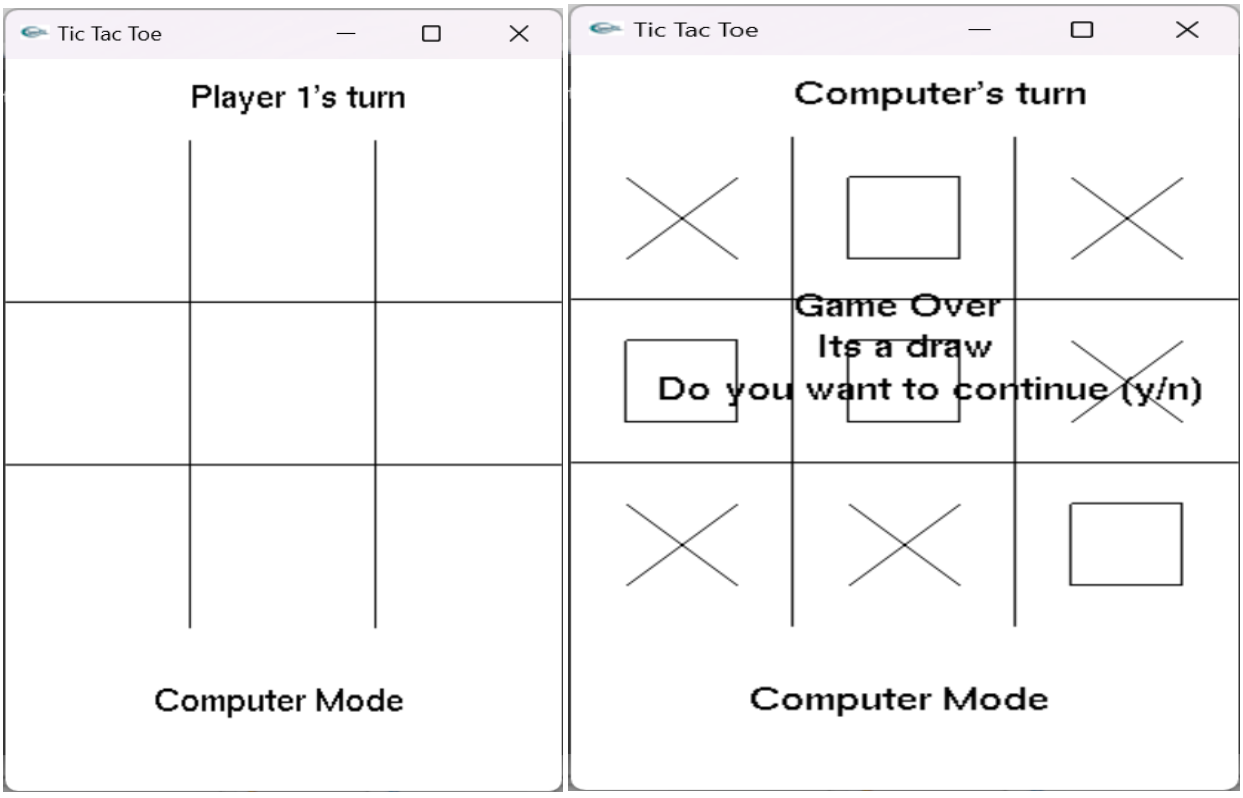

Figure 3: Player vs Computer game

8 CONCLUSION:

The Tic Tac Toe game is most familiar among all the age groups. Intelligence can be a property of any purpose-driven decision maker. This basic idea has been suggested many times. An algorithm of playing Tic Tac Toe has been presented and tested that works in efficient way. Overall the system works without any bugs.

9 REFERENCE:

[1] Patel, P., Patel, A., & Patel, N. (2020). Understanding the Minimax Algorithm. Journal of AI and Research, 15(3), 200-215.

[2] T. Y. Al-Taharwa, S. A. Al-Rahamneh, and F. A. Al-Taharwa. "Tic Tac Toe." ResearchGate, 2017

[3] https://github.com/Tic-Tac-Toe