**TIC TAC TOE GAME**

**Reg No :** 41731092

**Name :** P. Purna Sai

**Branch :** B.E CSE-AI (A3)

**Subject :** Code optimizing and debugging

**Sub code :** SCSA2402

**AIM :**

To Create TIC TAC TOE Game using python language

**PROJECT OVERVIEW :**

1. Create a 3x3 grid to represent the Tic Tac Toe board using a nested list or array.
2. Write a function to display the current state of the board on the console.
3. Write a function to prompt the first player to enter their move (the row and column where they want to place their symbol).
4. Write a function to validate the move entered by the player. The move is valid if the selected square is empty and within the bounds of the board.
5. Update the board with the player's move if it is valid.
6. Write a function to check if the game has been won by either player. This involves checking if any row, column, or diagonal has three of the same symbols in a row.
7. If the game has been won, display the final state of the board and declare the winner.
8. If the game is not won, prompt the second player to enter their move and repeat steps 4-7.
9. If all the squares are filled and no player has won, declare the game as a tie.
10. . Implement a main function that combines all the above functions to run the game.

**RULES OF THE GAME :**

The rules of Tic Tac Toe are simple. The game is played on a 3x3 grid, and each player takes turns placing their symbol (X or O) on the board. The first player to get three of their symbols in a row wins the game. The row can be horizontal, vertical, or diagonal.

If all nine squares on the board are filled and no player has won, the game is declared a tie. The game can be played with any number of rounds, and the winner is the player who wins the most rounds.

**VARIATIONS :**

Tic Tac Toe has many variations that add new twists to the classic game. One variation is called Gomoku, which is played on a larger board and requires players to get five in a row instead of three. Another variation is called Ultimate Tic Tac Toe, which is played on a larger board made up of smaller tic tac toe boards. Players must win three of these smaller boards in a row to win the game.

There are also many themed versions of Tic Tac Toe, such as Disney-themed boards or sports-themed boards. These versions add a fun twist to the classic game and make it more engaging for younger players.

**STRATEGY :**

Tic Tac Toe may seem like a game of chance, but there is actually a lot of strategy involved. The key to winning is to try to create multiple opportunities to win at the same time. For example, if you have two X's in a row, place your next X in a position that will create a third opportunity to win. This forces your opponent to block you, giving you another opportunity to win.

Another important strategy is to pay attention to your opponent's moves. If they are consistently blocking you in a certain spot, try to create a new opportunity elsewhere on the board. Finally, always try to take the center square first, as this gives you the most opportunities to win.

**CODE :**

**import** **tkinter** **as** **tk**

**from** **tkinter** **import** messagebox

**class** **TicTacToe**:

**def** **\_\_init\_\_**(self):

self.current\_player = None

self.players = {"X": "", "O": ""}

self.game\_over = False

self.board = [["", "", ""], ["", "", ""], ["", "", ""]]

self.root = tk.Tk()

self.root.title("Tic Tac Toe")

# Create player name entry fields and assign symbols

player\_labels = []

**for** symbol **in** self.players:

label = tk.Label(self.root, text=f"Enter name for player '{symbol}':",font=("Arial",**10**))

label.grid(row=len(player\_labels), column=**0**, padx=**15**, pady=**10**)

player\_labels.append(label)

entry = tk.Entry(self.root)

entry.grid(row=len(player\_labels)-**1**, column=**1**, padx=**10**, pady=**10**)

self.players[symbol] = entry

self.current\_player = "X" # Start with X as the first player

# Create buttons for the game board

self.buttons = []

**for** i **in** range(**3**):

row = []

**for** j **in** range(**3**):

button = tk.Button(self.root, text="", font=("Arial", **50**), width=**4**, height=**1**,

bg="black",fg="white",command=**lambda** i=i, j=j: self.button\_click(i, j))

button.grid(row=i+**3**, column=j, padx=**10**, pady=**10**)

row.append(button)

self.buttons.append(row)

# Create a reset button

reset\_button = tk.Button(self.root, text="Reset", font=("Arial", **30**),fg="red",bg="grey",

command=self.reset)

reset\_button.grid(row=**6**, column=**1**, pady=**30**)

self.root.mainloop()

**def** **button\_click**(self, row, col):

**if** self.game\_over:

**return**

**if** self.board[row][col] == "":

self.buttons[row][col].config(text=self.current\_player)

self.board[row][col] = self.current\_player

**if** self.check\_win():

messagebox.showinfo("Tic Tac Toe", f"{self.players[self.current\_player].get()} wins!")

self.game\_over = True

**elif** self.check\_draw():

messagebox.showinfo("Tic Tac Toe", "It's a draw!")

self.game\_over = True

**else**:

self.current\_player = "O" **if** self.current\_player == "X" **else** "X"

**def** **check\_win**(self):

# Check rows

**for** i **in** range(**3**):

**if** self.board[i][**0**] == self.board[i][**1**] == self.board[i][**2**] != "":

**return** True

# Check columns

**for** j **in** range(**3**):

**if** self.board[**0**][j] == self.board[**1**][j] == self.board[**2**][j] != "":

**return** True

# Check diagonals

**if** self.board[**0**][**0**] == self.board[**1**][**1**] == self.board[**2**][**2**] != "":

**return** True

**if** self.board[**0**][**2**] == self.board[**1**][**1**] == self.board[**2**][**0**] != "":

**return** True

**return** False

**def** **check\_draw**(self):

**for** i **in** range(**3**):

**for** j **in** range(**3**):

**if** self.board[i][j] == "":

**return** False

**return** True

**def** **reset**(self):

**for** i **in** range(**3**):

**for** j **in** range(**3**):

self.buttons[i][j].config(text="")

self.board[i][j] = ""

self.current\_player = "X"

self.game\_over = False

game = TicTacToe()

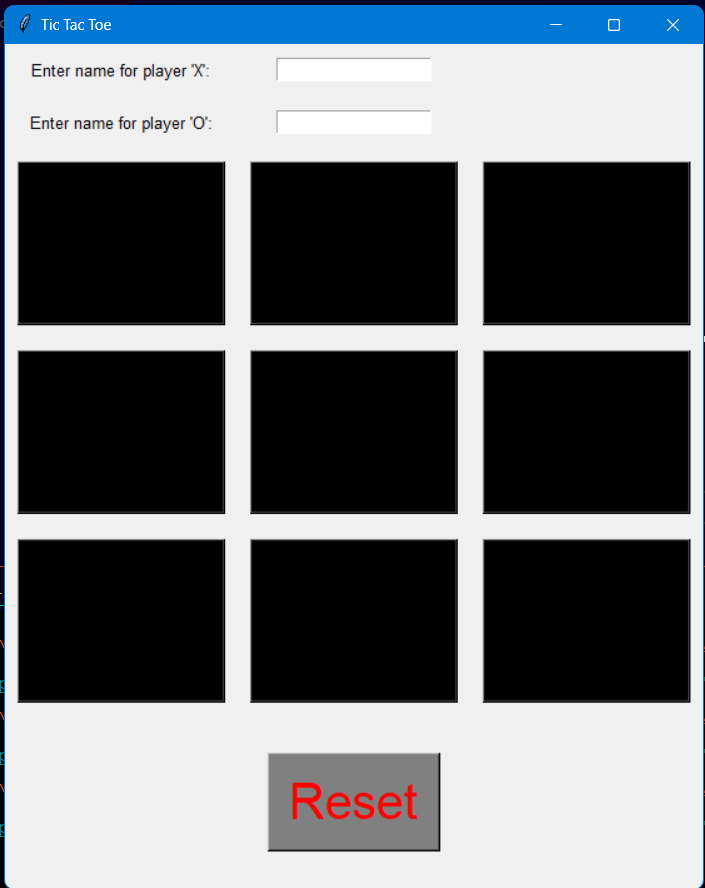
**EXPLANATION OF CODE :**

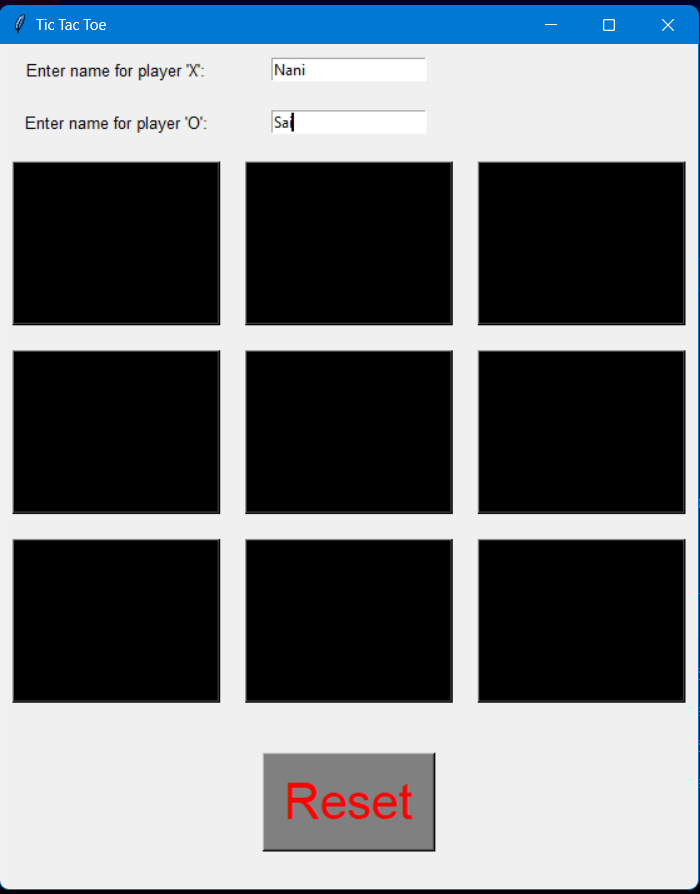
1. **tkinter**: This is a Python library used to create GUI applications. It is used to create the main window and widgets such as labels, buttons, and entries.
2. **messagebox**: This is a module in the **tkinter** library used to display message boxes such as alerts, warnings, and confirmations.
3. **self.\_\_init\_\_()**: This is a constructor method that initializes the **TicTacToe** object. It is called when an instance of the class is created.
4. **tk.Tk()**: This creates the main window object for the application.
5. **tk.Label()**: This creates a label widget that displays text on the GUI.
6. **tk.Entry()**: This creates an entry widget that allows the user to enter text.
7. **tk.Button()**: This creates a button widget that can be clicked to trigger an action.
8. **self.button\_click()**: This is a method that is called when a button is clicked. It updates the game board and checks for a win or a draw.
9. **self.check\_win()**: This is a method that checks if a player has won the game.

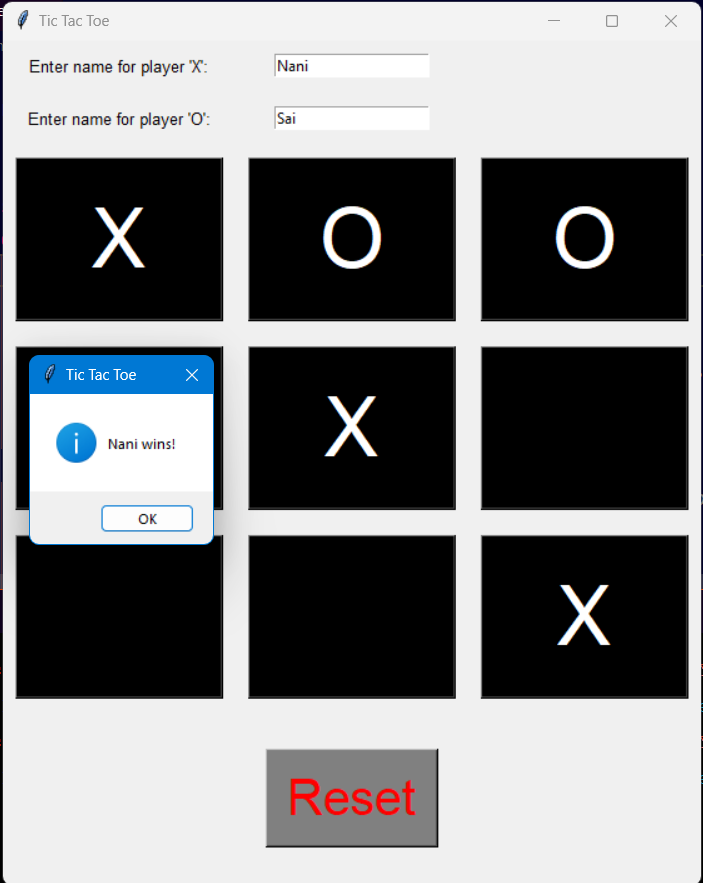
10.**self.check\_draw()**: This is a method that checks if the game is a draw.

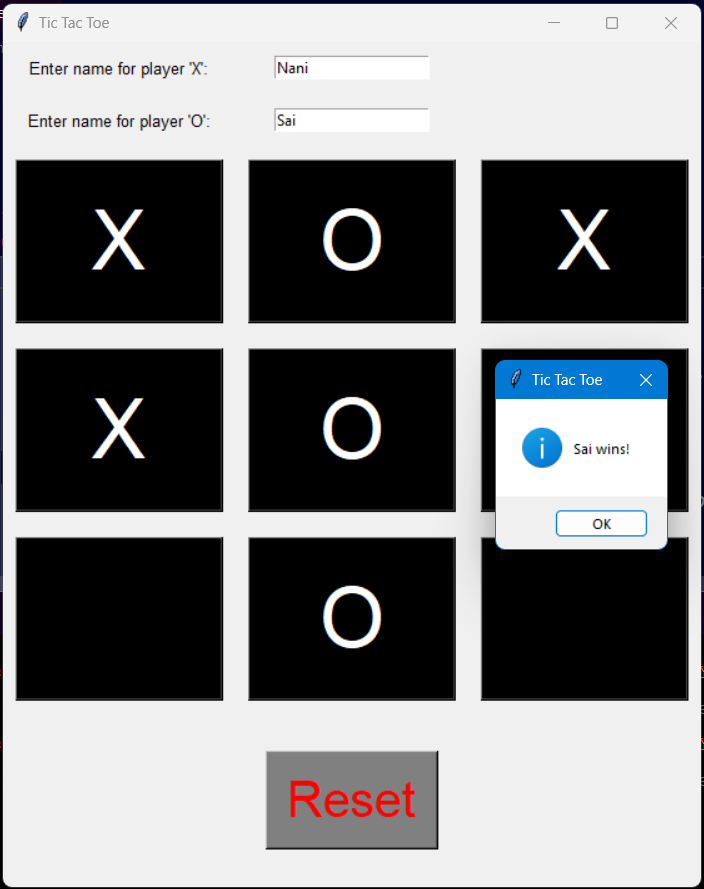
11.**self.reset()**: This is a method that resets the game board and the player names to start a new game.

**Output :**

****

****

****

****

****

**CONCLUSION :**

Tic Tac Toe is a timeless game that has stood the test of time. Its simplicity and strategic depth make it a favourite pastime for people of all ages. Whether you're playing on paper or on a computer, Tic Tac Toe is a great way to challenge your mind and have fun with friends and family.

So the next time you have a few minutes to spare, why not challenge someone to a game of Tic Tac Toe? You never know who might come out on top!

**RESULT :**

Thus the following python code for tic tac toe game was executed successfully and verified successfully .