Assignment 5

28-Feb-2025

Group_10

Member Names

- **Aung Soe Moe**
- **Yamong Phoo**
- **Nethmi Nayanathara Egodawaththa Arachchige**
- **Ama Nethmini Anjana Wathukarage**
- **Ayesha Subhashanee Hallala Hewage**
- **Tharindu Kavindya Silva Hettiyadura**

GitHub Repository link: https://github.com/Ayesha-Subhashanee/Group-project

Group 10 - Trello Board Project link: https://trello.com/b/wfK43SUk/future-project

Tic - Tac - Toe

Descriptions

Tic_Tac_Toe game is two players game. It plays in a 3 x 3 grid.

Each player take marking in the square with their respective symbols like "X" or "O" which are common use while playing this game.

The first player who align of three of marks in a horizontal or vertical or diagonal is the winner.

Rules

- Two player game with taking trun is played in a 3 x 3 grid.
- Each player choose their respective symbol. Example; Player 1 choose "X" and Player 2 choose "O".
- Players take trun and mark their respective symbol in an empty square cell.
- Each player try to mark with repective symbol not to win the other player.
- Players also try to align their symbol in a horizontal or vertical or diagonal.
- If 9 all cells are filled, there is no winning palyer. The game is end in a draw.

1. Import necessaries

In [25]:
```
import tkinter as tk
import random
```

2. Initializing the Board

In [26]:
```
# Initialize the board and choosing respective symbol for current player
board = ["-"] * 9
currentplayer = "X"
winner = None
```

3. Creating the Tkinter Window and button board

In [27]:
```
# Create the main window for game board
root = tk.Tk()
root.title("Tic-Tac-Toe")

# Create buttons for the board
buttons = []
```

4. Restarting the game

In [28]:
```
def restart_game():
    global board, currentplayer, winner
    board = ["-"] * 9
```

```
    currentplayer = "X"
    winner = None
    for button in buttons:
        button.config(text="", state=tk.NORMAL)
```

5. Handling the palyer move

In [29]:
```
# Function to handle button clicks
def player_move(index):
    global currentplayer
    if board[index] == "-" and winner is None:
        board[index] = currentplayer
        buttons[index].config(text=currentplayer)
        if check_winner():
            return
        switch_player()
        computer_move()
```

6. This function is for computer move.**

In [30]:
```
# Function for computer's move
def computer_move():
    global currentplayer
    available_positions = [i for i in range(9) if board[i] == "-"]
    if available_positions and winner is None:
        position = random.choice(available_positions)
        board[position] = "O"
        buttons[position].config(text="O")
        if check_winner():
            return
        switch_player()
```

** 7. This part is for switching player.**

In [31]:
```
# switching the player
def switch_player():
    global currentplayer
    currentplayer = "O" if currentplayer == "X" else "X"
```

8. checking winner according to Rows, Columns and Diagonals.

In [32]:
```
# checking the winner
def check_winner():
    global winner
    win_patterns = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Rows
        [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Columns
        [0, 4, 8], [2, 4, 6]  # Diagonals
    ]
    for pattern in win_patterns:
        if board[pattern[0]] == board[pattern[1]] == board[pattern[2]] and board[pattern[0]] != "-":
            winner = board[pattern[0]]
            result_label.config(text=f"The winner is {winner}!")
            disable_buttons()
            return True
    if "-" not in board:
        result_label.config(text="It's a draw!")
        return True
    return False
```

9. This function is for button disable.

In [33]:
```
# creating disable button after winning
def disable_buttons():
    for button in buttons:
        button.config(state=tk.DISABLED)
```

10. This is for UI Board.

In [34]:
```
# Creating UI board
for i in range(9):
    button = tk.Button(root, text="", font=("Arial", 20), width=5, height=2,
                       command=lambda i=i: player_move(i))
    button.grid(row=i//3, column=i%3)
    buttons.append(button)
```

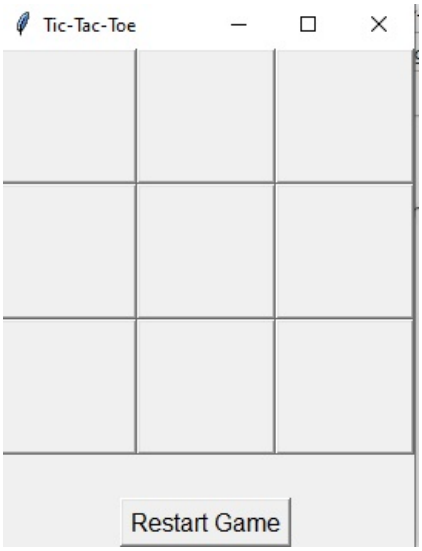
```
# label for displaying results
result_label = tk.Label(root, text="", font=("Arial", 14))
result_label.grid(row=3, column=0, columnspan=3)
```

11. Restarting the game.

In [35]:
```
# Restart button after game end
restart_button = tk.Button(root, text="Restart Game", font=("Arial", 12), command=restart_game)
restart_button.grid(row=4, column=0, columnspan=3)

# Run the main event loop
root.mainloop()
```

```
- | - | -
-----
- | - | -
-----
- | - | -
The game is over and it is a draw.!
X | - | -
-----
- | - | O
-----
- | - | -
The game is over and it is a draw.!
X | - | -
-----
X | O | O
-----
- | - | -
The game is over and it is a draw.!
X | X | -
-----
X | O | O
-----
- | O | -
The winner is X
X | X | O
-----
X | O | O
-----
X | O | -
```



Conclution

In this project, we successfully implemented a Tic-Tac-Toe game using Python and Tkinter. The game allows two players to take turns marking “X” or “O” on a 3x3 grid, with the objective of aligning three marks in a row, column, or diagonal to win. If all spaces are filled without a winner, the game ends in a draw.