Tic-Tac-Toe Game using Pygame

Introduction:

The Tic-Tac-Toe game is a simple game that can be implemented using the Pygame library in Python. Pygame is a set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.

Packages Used:

Pygame: Pygame is a set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.

Algorithm:

1. Initialize the Pygame library and create a window with a size of 300x300 pixels and caption of "Tic-Tac-Toe".
2. Load the images of "X" and "O" that will be used to display the player's moves.
3. Create a 2D list called "squares" to store the state of the squares, which will be used to check for a win or a draw.
4. Create a variable called "turn" to store the current turn and initialize it to 0.
5. Create a function called "draw_x_o" that takes the position of the square as an argument and draws the image of the "X" or "O" depending on the value of the "turn" variable.
6. Create a function called "on_mouse_click" that handles mouse clicks, converts the mouse position to grid coordinates, checks if the square is empty, calls the "draw_x_o" function, updates the state of the square and calls the "check_win_or_draw" function to check if there's a winner or a draw.
7. Create a function called "check_win_or_draw" that checks if either player has won the game or if it's a draw.
8. Create a main game loop that handles events and updates the display.
9. Close the game using pygame.quit().

Built-in functions:

- **pygame.init():** Initialize all pygame modules (in this case only pygame)

- **pygame.display.set_mode(size) :** create a window of the given size.
- **pygame.display.set_caption(title) :** sets the caption of the window to the given title
- **pygame.image.load(path) :** loads an image from the specified file path
- **screen.blit(image, pos) :** draw an image on the screen at the given position
- **pygame.event.get() :** get all the events that have occurred since the last time this function was called
- **pygame.quit() :** uninitialize all pygame modules

Working Procedure:

- The code starts by importing the Pygame library, initializing it, and creating a window with a size of 300x300 pixels and a caption of "Tic-Tac-Toe".
- Next, the images of "X" and "O" are loaded to be used to display the player's moves.
- A 2D list called "squares" is created to store the state of the squares, which will be used to check for a win or a draw.
- A variable called "turn" is created to store the current turn and it is initialized to 0.
- A function called "draw_x_o" is created that takes the position of the square as an argument and draws the image of the "X" or "O" depending on the value of the "turn" variable.
- A function called "on_mouse_click" is created that handles mouse clicks, converts the mouse position to grid coordinates, checks if the square is empty, calls the "draw_x_o" function, updates the state of the square and calls the "check_win_or_draw" function to check if there's a winner or a draw.
- A function called "check_win_or_draw" is created that checks if either player has won the game or if it's a draw.
- A main game loop is created that handles events and updates the display.
- The game is closed using pygame.quit().

Sample output:

"Player 1 wins!" if player 1 wins.

"Player 2 wins!" if player 2 wins.

"It's a draw!" if the game is a draw.

In this script, Pygame library is utilized to create a basic Tic-Tac-Toe game. The game allows two players to play the game. The game state is stored in a 2D list called "squares", and the "turn" variable is used to keep track of which player's turn it is. The "draw_x_o" function takes the position of the square as an argument and draws the image of the "X" or "O" depending on the value of the "turn" variable. The "on_mouse_click" function handles mouse clicks and converts the mouse position to grid coordinates, it then checks if the square is empty, if it is empty it calls the "draw_x_o" function and updates the state of the square and calls the check_win_or_draw function to check if there's a winner or a draw. The function "check_win_or_draw" is used to check if either player has won the game or if it's a draw. It uses a nested for loop to check the rows, columns and diagonals for a win. If a win condition is met it prints a message on the console "Player 1 wins" or "Player 2 wins" and if it's a draw it prints "It's a draw!". The main game loop is used to handle events and update the display. Finally, the game is closed using pygame.quit().

It is important to note that the above code is a simplified version of a Tic-Tac-Toe game, and it might not include all the features you would want to include in your game. You should add more features such as a menu, sound effects, and a GUI for the game. Additionally, it is recommended to add more error handling, comments and make the code more readable and test the game before the final implementation.

Output Screenshot: