This is a simple Tic Tac Toe game written in Python. Let's break down the code:

1. \*\*Import Statements\*\*:

```python

import os

import time

import random

```

These lines import necessary modules for clearing the screen (`os`), introducing delays (`time`), and generating random numbers (`random`).

2. \*\*Board Representation\*\*:

```python

board = ["", " ", " ", " ", " ", " ", " ", " ", " ", " "]

```

The Tic Tac Toe board is represented using a list. Each element of the list corresponds to a position on the board. Empty positions are represented by a space.

3. \*\*Function Definitions\*\*:

- `print\_board()`: This function prints the current state of the board.

- `is\_winner(board, player)`: Checks if a player has won by examining all possible win conditions.

- `is\_board\_full(board)`: Checks if the board is full (no more spaces left).

- `get\_computer\_move(board, player)`: Generates a random move for the computer. The computer prioritizes taking the center square if it's empty.

4. \*\*Game Loop\*\*:

The main loop of the game runs until the game is won or tied.

```python

while True:

```

- Clears the screen.

- Prints the current state of the board.

- Gets player X's input.

- Checks for a win by X.

- Prints the board again.

- Checks for a tie.

- Gets player O's input (computer's move).

- Checks for a win by O.

- Checks for a tie.

5. \*\*Player X's Move\*\*:

```python

choice = input("Please choose an empty space for X. ")

choice = int(choice)

```

- Asks the player for their move.

- Converts the input to an integer.

- If the chosen space is empty, it fills it with "X", else it prompts the player to choose again.

6. \*\*Checking for X's Win and Board Fullness\*\*:

```python

if is\_winner(board, "X"):

print("X wins! Congratulations")

break

```

- Checks if player X has won.

- If true, prints a victory message and exits the loop.

- Checks if the board is full (tie).

- If true, prints a tie message and exits the loop.

7. \*\*Player O's (Computer's) Move\*\*:

```python

choice = get\_computer\_move(board, "O")

```

- Calls `get\_computer\_move()` to get the computer's move.

8. \*\*Checking for O's Win and Board Fullness\*\*:

```python

if is\_winner(board, "O"):

print("O wins! Congratulations")

break

```

- Checks if player O has won.

- If true, prints a victory message and exits the loop.

- Checks if the board is full (tie).

- If true, prints a tie message and exits the loop.

The game continues until one player wins or there is a tie. The board is continuously updated after each move.