```
####  TIC TAC TOE    ####

# START;


# FUNCTIONS;


def default():
    # To be printed as Default;
    print("\nWelcome! Let's play TIC TAC TOE!\n")


def rules():
    print("The board will look like this!")
    print(
        "The positions of this 3 x 3 board is same as the right side
of your key board.\n"
    )
    print(" 7 | 8 | 9 ")
    print("-----------")
    print(" 4 | 5 | 6 ")
    print("-----------")
    print(" 1 | 2 | 3 ")
    print("\nYou just have to input the position(1-9).")


def play():
    # Asking if the player is ready;
    return (
        input("\nAre you ready to play the game? Enter [Y]es or
[N]o.\t")
        .upper()
        .startswith("Y")
    )


def names():
```

```
    # Player names input;

    p1_name = input("\nEnter NAME of PLAYER 1:\t").capitalize()
    p2_name = input("Enter NAME of PLAYER 2:\t").capitalize()
    return (p1_name, p2_name)


def choice():
    # Player choice input;
    p1_choice = " "
    p2_choice = " "
    while (
        p1_choice != "X" or p1_choice != "O"
    ):  # while loop; if the entered value isn't X or O;
        # WHILE LOOP STARTS

        p1_choice = input(f"\n{p1_name}, Do you want to be X or
O?\t")[0].upper()
        # The input above has [0].upper() in the end;
        # So the user can enter x, X, xxxx or XXX; the input will
always be taken as X;
        # Thereby, increasing the user input window;

        if p1_choice == "X" or p1_choice == "O":
            # if entered value is X or O; get out of the loop;
            break
        print("INVALID INPUT! Please Try Again!")
        # if the entered value isn't X or O, re-run the while loop;

        # WHILE LOOP ENDS
    # Assigning the value to p2 and then diplaying the values;
    if p1_choice == "X":
        p2_choice = "O"
    elif p1_choice == "O":
        p2_choice = "X"

    return (p1_choice, p2_choice)
```

```
def first_player():
    # This function will randomly decide who will go first;
    import random

    return random.choice((0, 1))


def display_board(board, avail):
    print(
        "    "
        + " {} | {} | {} ".format(board[7], board[8], board[9])
        + "            "
        + " {} | {} | {} ".format(avail[7], avail[8], avail[9])
    )
    print("    " + "-----------" + "            " + "-----------")
    print(
        "    "
        + " {} | {} | {} ".format(board[4], board[5], board[6])
        + "            "
        + " {} | {} | {} ".format(avail[4], avail[5], avail[6])
    )
    print("    " + "-----------" + "            " + "-----------")
    print(
        "    "
        + " {} | {} | {} ".format(board[1], board[2], board[3])
        + "            "
        + " {} | {} | {} ".format(avail[1], avail[2], avail[3])
    )


def player_choice(board, name, choice):
    position = 0
    # Initialising position as 0^; so it passes through the while
loop;
    while position not in [1, 2, 3, 4, 5, 6, 7, 8, 9] or not
space_check(
        board, position
    ):
        position = int(
```

```
            input(f"\n{name} ({choice}), Choose your next position:
(1-9) \t")
        )

        if (
            position not in [1, 2, 3, 4, 5, 6, 7, 8, 9]
            or not space_check(board, position)
            or position == ""
        ):
            # To check whether the given position is in the set [1-9]
or whether it is empty or occupied;
            print(f"INVALID INPUT. Please Try Again!\n")
    print("\n")
    return position


# THIS IS THEFUNCTION WHERE AI IS ADDED:
def CompAI(board, name, choice):
    position = 0
    possibilities = [x for x, letter in enumerate(board) if letter ==
" " and x != 0]

    # including both X and O, since if computer will win, he will
place a choice there, but if the component will win --> we have to
block that move
    for let in ["O", "X"]:
        for i in possibilities:
            # Creating a copy of the board everytime, placing the
move and checking if it wins;
            # Creating a copy like this  and not this boardCopy =
board, since changes to boardCopy changes the original board;
            boardCopy = board[:]
            boardCopy[i] = let
            if win_check(boardCopy, let):
                position = i
                return position

    openCorners = [x for x in possibilities if x in [1, 3, 7, 9]]
```

```
    if len(openCorners) > 0:
        position = selectRandom(openCorners)
        return position

    if 5 in possibilities:
        position = 5
        return position

    openEdges = [x for x in possibilities if x in [2, 4, 6, 8]]

    if len(openEdges) > 0:
        position = selectRandom(openEdges)
        return position


def selectRandom(board):
    import random

    ln = len(board)
    r = random.randrange(0, ln)
    return board[r]


def place_marker(board, avail, choice, position):
    # To mark/replace the position on the board list;
    board[position] = choice
    avail[position] = " "


def space_check(board, position):
    # To check whether the given position is empty or occupied;
    return board[position] == " "


def full_board_check(board):
    # To check if the board is full, then the game is a draw;
    for i in range(1, 10):
        if space_check(board, i):
            return False
```

```
    return True


def win_check(board, choice):
    # To check if one of the following patterns are true; then the
respective player has won!;

    # HORIZONTAL CHECK;
    return (
        (board[1] == choice and board[2] == choice and board[3] ==
choice)
        or (board[4] == choice and board[5] == choice and board[6] ==
choice)
        or (board[7] == choice and board[8] == choice and board[9] ==
choice)
        # VERTICAL CHECK;
        or (board[1] == choice and board[4] == choice and board[7] ==
choice)
        or (board[2] == choice and board[5] == choice and board[8] ==
choice)
        or (board[3] == choice and board[6] == choice and board[9] ==
choice)
        # DIAGONAL CHECK;
        or (board[1] == choice and board[5] == choice and board[9] ==
choice)
        or (board[3] == choice and board[5] == choice and board[7] ==
choice)
    )


def delay(mode):
    if mode == 2:
        import time

        time.sleep(2)


def replay():
    # If the users want to play the game again?
```

```
    return (
        input("\nDo you want to play again? Enter [Y]es or [N]o: ")
        .lower()
        .startswith("y")
    )


# MAIN PROGRAM STARTS;

print("\n\t\t NAMASTE! \n")
input("Press ENTER to start!")

default()
rules()


while True:

##########################################################################
###############

    # Creating the board as a list; to be kept replacing it with user
input;
    theBoard = [" "] * 10

    # Creating the available options on the board:
    available = [str(num) for num in range(0, 10)]  # a List
Comprehension
    # available = '0123456789'

    print("\n[0]. Player vs. Computer")
    print("[1]. Player vs. Player")
    print("[2]. Computer vs. Computer")
    mode = int(input("\nSelect an option [0]-[2]: "))
    if mode == 1:
        # Asking Names;
        p1_name, p2_name = names()
        # Asking Choices; Printing choices; X or O;
        p1_choice, p2_choice = choice()
```

```
        print(f"\n{p1_name}:", p1_choice)
        print(f"{p2_name}:", p2_choice)

    elif mode == 0:
        p1_name = input(
            "\nEnter NAME of PLAYER who will go against the
Computer:\t"
        ).capitalize()
        p2_name = "Computer"
        # Asking Choices; Printing choices; X or O;
        p1_choice, p2_choice = choice()
        print(f"\n{p1_name}:", p1_choice)
        print(f"{p2_name}:", p2_choice)

    else:
        p1_name = "Computer1"
        p2_name = "Computer2"
        p1_choice, p2_choice = "X", "O"
        print(f"\n{p1_name}:", p1_choice)
        print(f"\n{p2_name}:", p2_choice)

    # Printing randomly who will go first;
    if first_player():
        turn = p2_name
    else:
        turn = p1_name

    print(f"\n{turn} will go first!")

    # Asking the user, if ready to play the game; Output will be True
or False;
    if mode == 2:
        ent = input(
            "\nThis is going to be fast! Press Enter for the battle
to begin!\n"
        )
        play_game = 1
    else:
        play_game = play()
```

```
    while play_game:
        ############################
        # PLAYER1
        if turn == p1_name:
            # Displaying the board;
            display_board(theBoard, available)

            # Position of the input;
            if mode != 2:
                position = player_choice(theBoard, p1_name,
p1_choice)
            else:
                position = CompAI(theBoard, p1_name, p1_choice)
                print(f"\n{p1_name} ({p1_choice}) has placed on
{position}\n")

            # Replacing the ' ' at *position* to *p1_choice* in
*theBoard* list;
            place_marker(theBoard, available, p1_choice, position)

            # To check if Player 1 has won after the current input;
            if win_check(theBoard, p1_choice):
                display_board(theBoard, available)

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                if mode:
                    print(f"\n\nCONGRATULATIONS {p1_name}! YOU HAVE
WON THE GAME!\n\n")
                else:
                    print("\n\nTHE Computer HAS WON THE GAME!\n\n")

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                play_game = False

            else:
                # To check if the board is full; if yes, the game is
a draw;
                if full_board_check(theBoard):
```

```
                    display_board(theBoard, available)
                    print("~~~~~~~~~~~~~~~~~~~")
                    print("\nThe game is a DRAW!\n")
                    print("~~~~~~~~~~~~~~~~~~~")
                    break
                # If none of the above is possible, next turn of
Player 2;
                else:
                    turn = p2_name

        ############################
        # PLAYER2
        elif turn == p2_name:
            # Displaying the board;
            display_board(theBoard, available)

            # Position of the input;
            if mode == 1:
                position = player_choice(theBoard, p2_name,
p2_choice)
            else:
                position = CompAI(theBoard, p2_name, p2_choice)
                print(f"\n{p2_name} ({p2_choice}) has placed on
{position}\n")

            # Replacing the ' ' at *position* to *p2_choice* in
*theBoard* list;
            place_marker(theBoard, available, p2_choice, position)

            # To check if Player 2 has won after the current input;
            if win_check(theBoard, p2_choice):
                display_board(theBoard, available)

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                if mode:
                    print(f"\n\nCONGRATULATIONS {p2_name}! YOU HAVE
WON THE GAME!\n\n")
                else:
                    print("\n\nTHE Computer HAS WON THE GAME!\n\n")
```

```
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                play_game = False

            else:
                # To check if the board is full; if yes, the game is
a draw;
                if full_board_check(theBoard):
                    display_board(theBoard, available)
                    print("~~~~~~~~~~~~~~~~~~~")
                    print("\nThe game is a DRAW!\n")
                    print("~~~~~~~~~~~~~~~~~~~")
                    break
                # If none of the above is possible, next turn of
Player 2;
                else:
                    turn = p1_name

    # If the users want to play the game again?
    if replay():
        # if Yes;
        continue
    else:
        # if No;
        break


##########################################################################
###############

print("\n\n\t\t\tTHE END!")


# END
```