Magic Square method

In [1]:

```
import random

def print_board(board):
    for row in board:
        print(" | ".join(row))
        print("-" * 13)

def is_winner(board, player):
    for row in board:
        if all(cell == player for cell in row):
            return True

    for col in range(3):
        if all(board[row][col] == player for row in range(3)):
            return True

    if all(board[i][i] == player for i in range(3)) or all(board[i][2 - i] == player for i in range(3)):
        return True

    return False

def is_board_full(board):
    return all(cell != ' ' for row in board for cell in row)

def get_user_move():
    while True:
        try:
            move = int(input("Enter your move (1-9): "))
            if 1 <= move <= 9:
                return move
            else:
                print("Invalid move. Please enter a number between 1 and 9.")
        except ValueError:
            print("Invalid input. Please enter a number.")

def calculate_computer_move(board, player_symbol, computer_symbol):
    magic_square = [
        [8, 3, 4],
        [1, 5, 9],
        [6, 7, 2]
    ]
```

```
    empty_cells = [(i, j) for i in range(3) for j in range(3) if board[i][j] == ' ']

    for i, j in empty_cells:
        temp_board = [row[:] for row in board]
        temp_board[i][j] = computer_symbol
        if is_winner(temp_board, computer_symbol):
            return i * 3 + j + 1

    for i, j in empty_cells:
        temp_board = [row[:] for row in board]
        temp_board[i][j] = player_symbol
        if is_winner(temp_board, player_symbol):
            return i * 3 + j + 1

    return random.choice(empty_cells)[0] * 3 + random.choice(empty_cells)[1] + 1

def play_tic_tac_toe():
    board = [[' ' for _ in range(3)] for _ in range(3)]
    user_symbol, computer_symbol = 'X', 'O'

    print("Welcome to Tic Tac Toe!")
    print_board(board)

    for move_num in range(1, 10):
        current_player = user_symbol if move_num % 2 == 1 else computer_symbol

        if current_player == user_symbol:
            user_move = get_user_move()
            row, col = divmod(user_move - 1, 3)
        else:
            computer_move = calculate_computer_move(board, user_symbol, computer_symbol)
            row, col = divmod(computer_move - 1, 3)
            print(f"Computer chooses position {computer_move}")

        while board[row][col] != ' ':
            print("ERROR! That position is already taken. Choose a different one.")
            if current_player == user_symbol:
                user_move = get_user_move()
                row, col = divmod(user_move - 1, 3)
            else:
                computer_move = calculate_computer_move(board, user_symbol, computer_symbol)
                row, col = divmod(computer_move - 1, 3)

        board[row][col] = user_symbol if current_player == user_symbol else computer_symbol
        print_board(board)

        if is_winner(board, current_player):
            print(f"{current_player} wins!")
            break
```

```
        if is_board_full(board):
            print("It's a tie!")
            break

if __name__ == "__main__":
    play_tic_tac_toe()
```

```
Welcome to Tic Tac Toe!
  |   |
-------------
  |   |
-------------
  |   |
-------------

  |   |
-------------
  |   |
-------------
  |   | X
-------------
Computer chooses position 2
  | O |
-------------
  |   |
-------------
  |   | X
-------------

X | O |
-------------
  |   |
-------------
  |   | X
-------------
Computer chooses position 5
X | O |
-------------
  | O |
-------------
  |   | X
-------------

X | O |
-------------
  | O |
-------------
  | X | X
-------------
Computer chooses position 7
```

```
X | O |
-------------
  | O |
-------------
O | X | X
-------------

X | O | X
-------------
  | O |
-------------
O | X | X
-------------
Computer chooses position 6
X | O | X
-------------
  | O | O
-------------
O | X | X
-------------

X | O | X
-------------
X | O | O
-------------
O | X | X
-------------
It's a tie!
```

In [ ]: