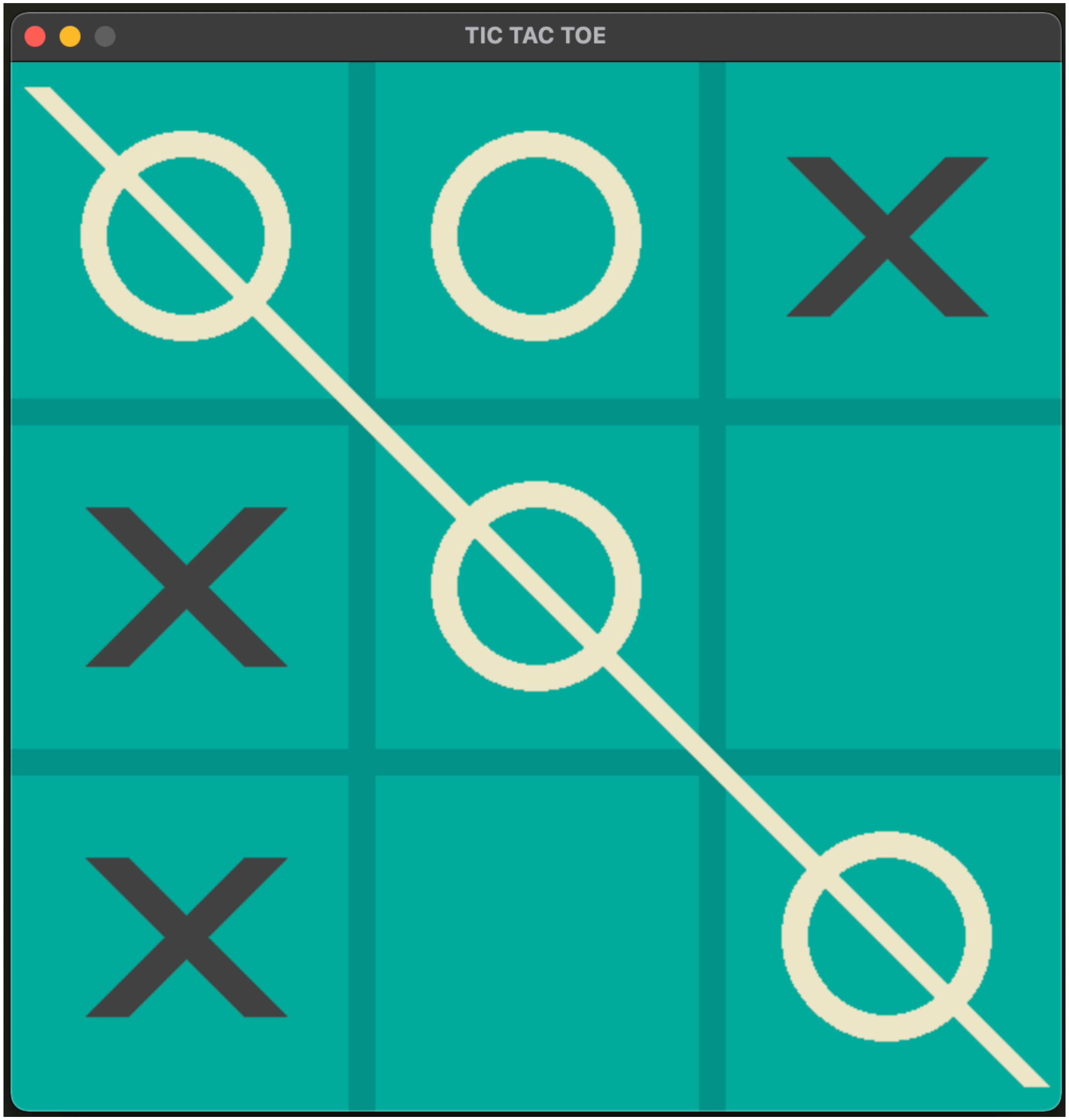
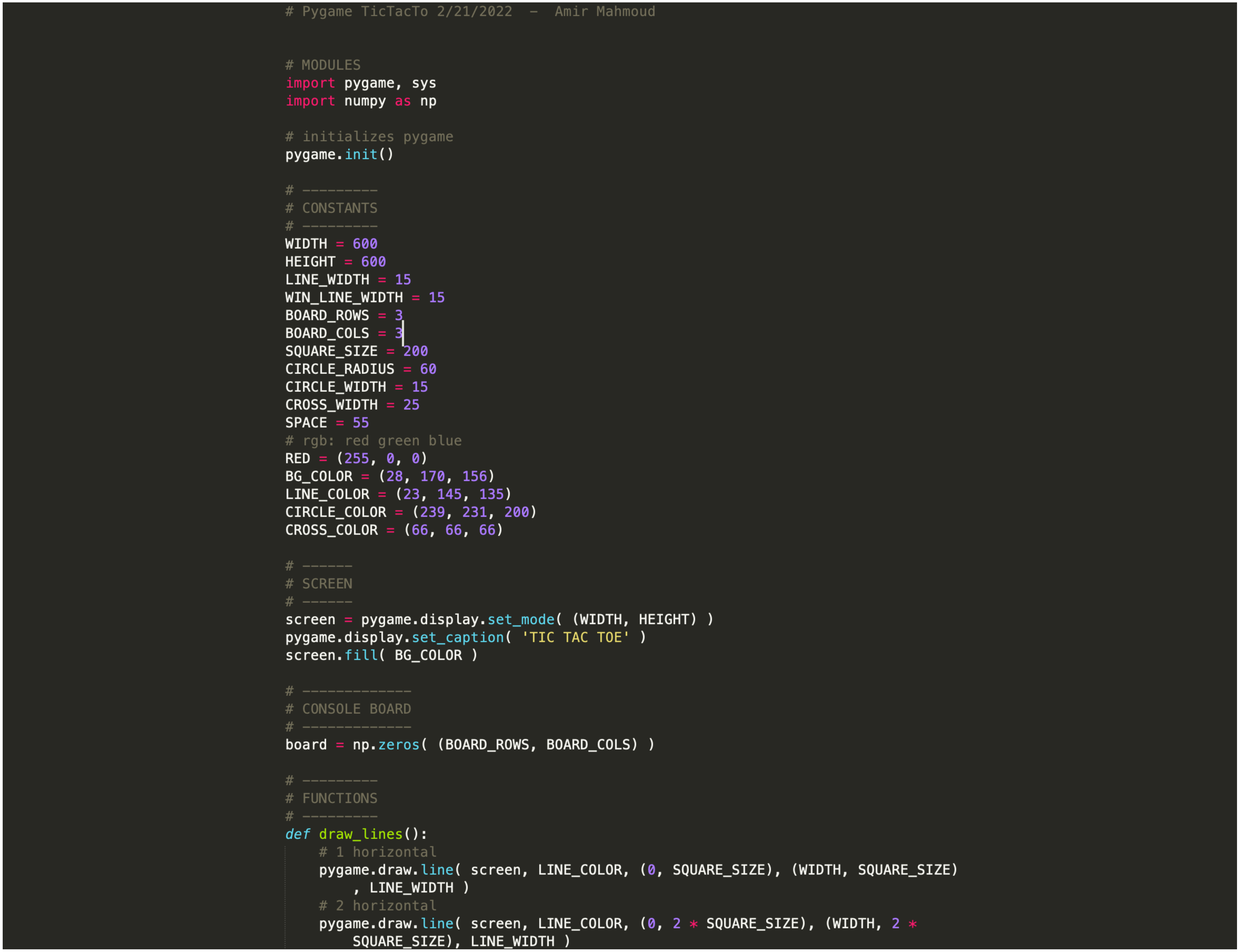
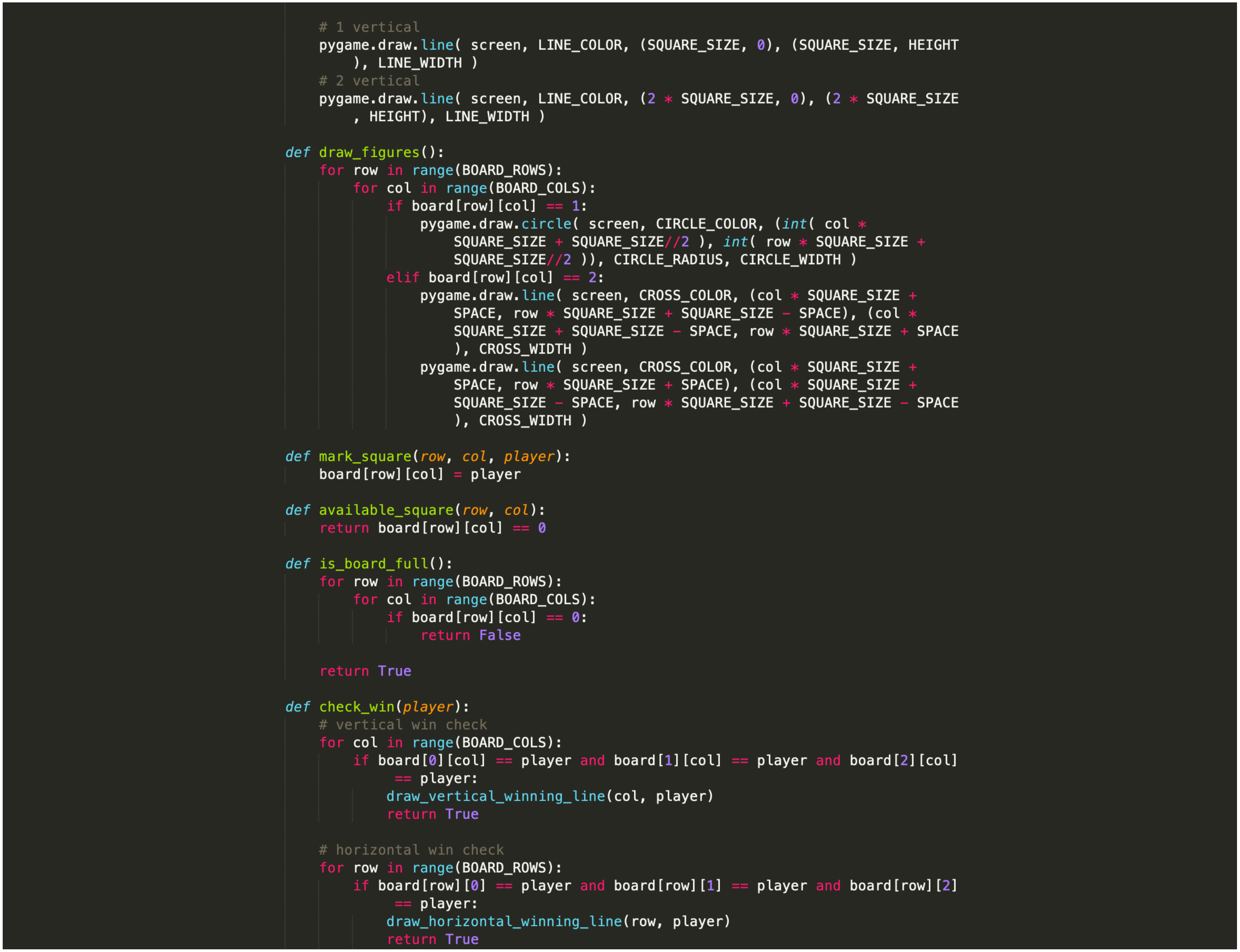
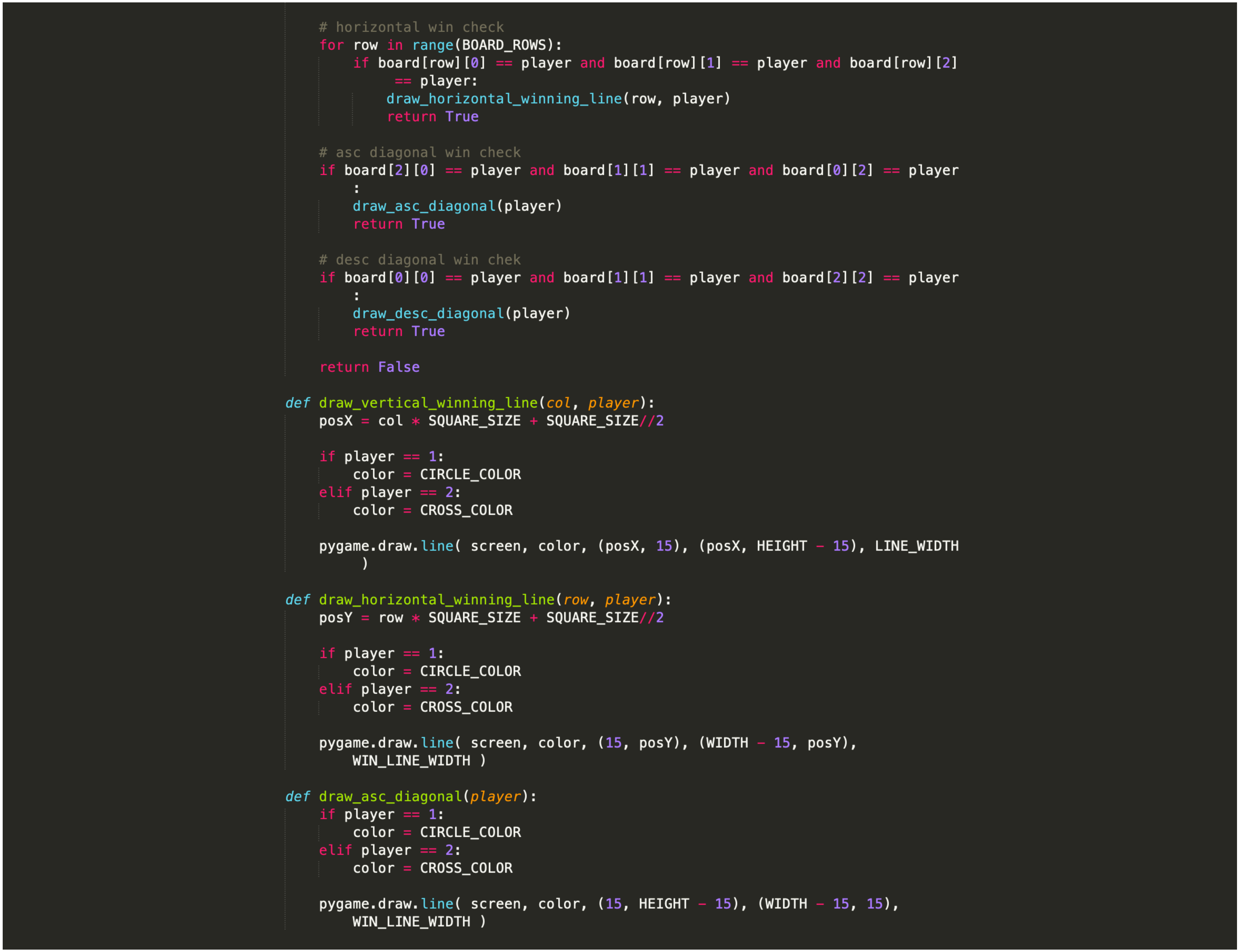

TIC TAC TOE



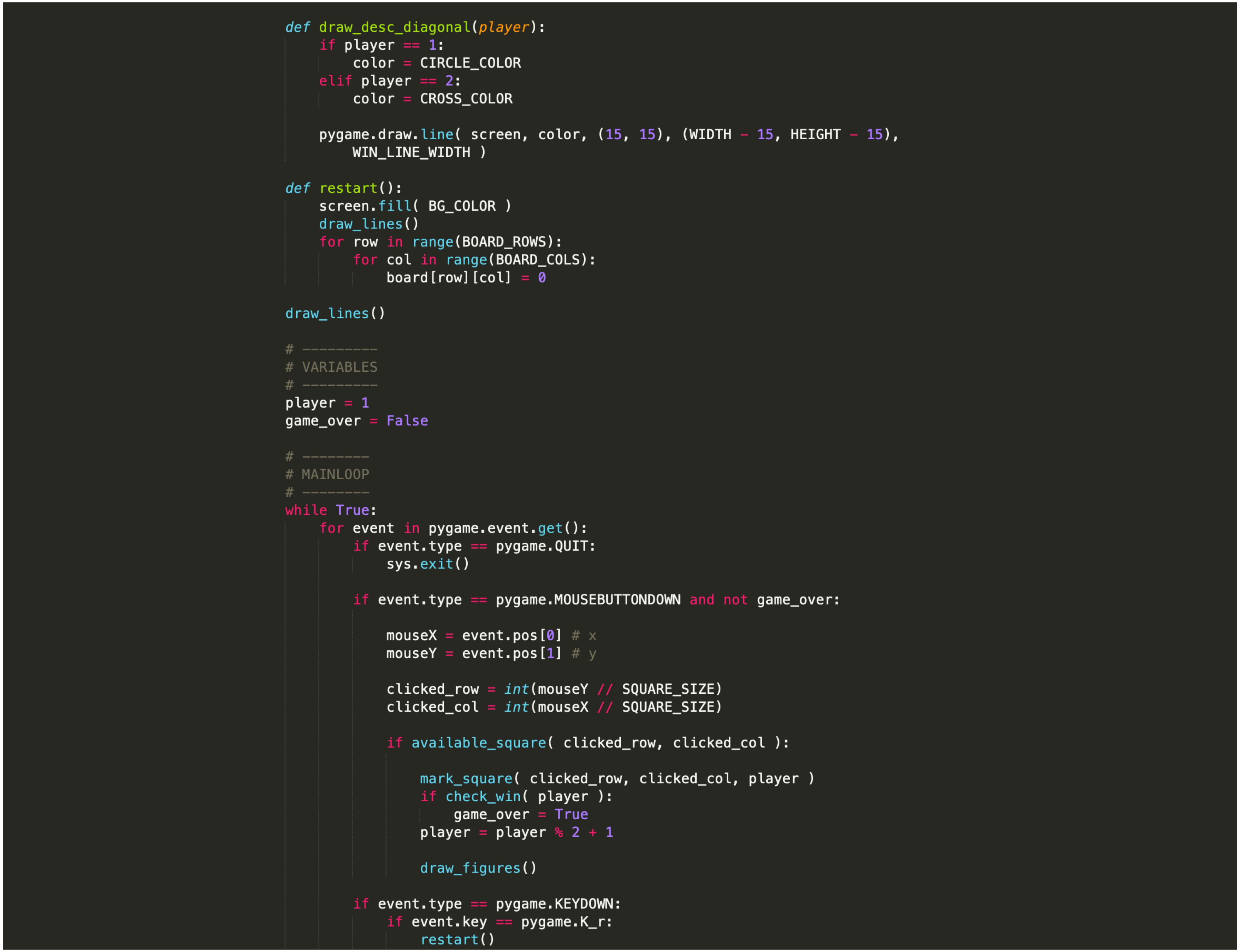
```python
# Pygame TicTacTo 2/21/2022  -  Amir Mahmoud


# MODULES
import pygame, sys
import numpy as np

# initializes pygame
pygame.init()

# ---------
# CONSTANTS
# ---------
WIDTH = 600
HEIGHT = 600
LINE_WIDTH = 15
WIN_LINE_WIDTH = 15
BOARD_ROWS = 3
BOARD_COLS = 3
SQUARE_SIZE = 200
CIRCLE_RADIUS = 60
CIRCLE_WIDTH = 15
CROSS_WIDTH = 25
SPACE = 55
# rgb: red green blue
RED = (255, 0, 0)
BG_COLOR = (28, 170, 156)
LINE_COLOR = (23, 145, 135)
CIRCLE_COLOR = (239, 231, 200)
CROSS_COLOR = (66, 66, 66)

# ------
# SCREEN
# ------
screen = pygame.display.set_mode( (WIDTH, HEIGHT) )
pygame.display.set_caption( 'TIC TAC TOE' )
screen.fill( BG_COLOR )

# -------------
# CONSOLE BOARD
# -------------
board = np.zeros( (BOARD_ROWS, BOARD_COLS) )

# ---------
# FUNCTIONS
# ---------
def draw_lines():
    # 1 horizontal
    pygame.draw.line( screen, LINE_COLOR, (0, SQUARE_SIZE), (WIDTH, SQUARE_SIZE)
        , LINE_WIDTH )
    # 2 horizontal
    pygame.draw.line( screen, LINE_COLOR, (0, 2 * SQUARE_SIZE), (WIDTH, 2 *
        SQUARE_SIZE), LINE_WIDTH )
```



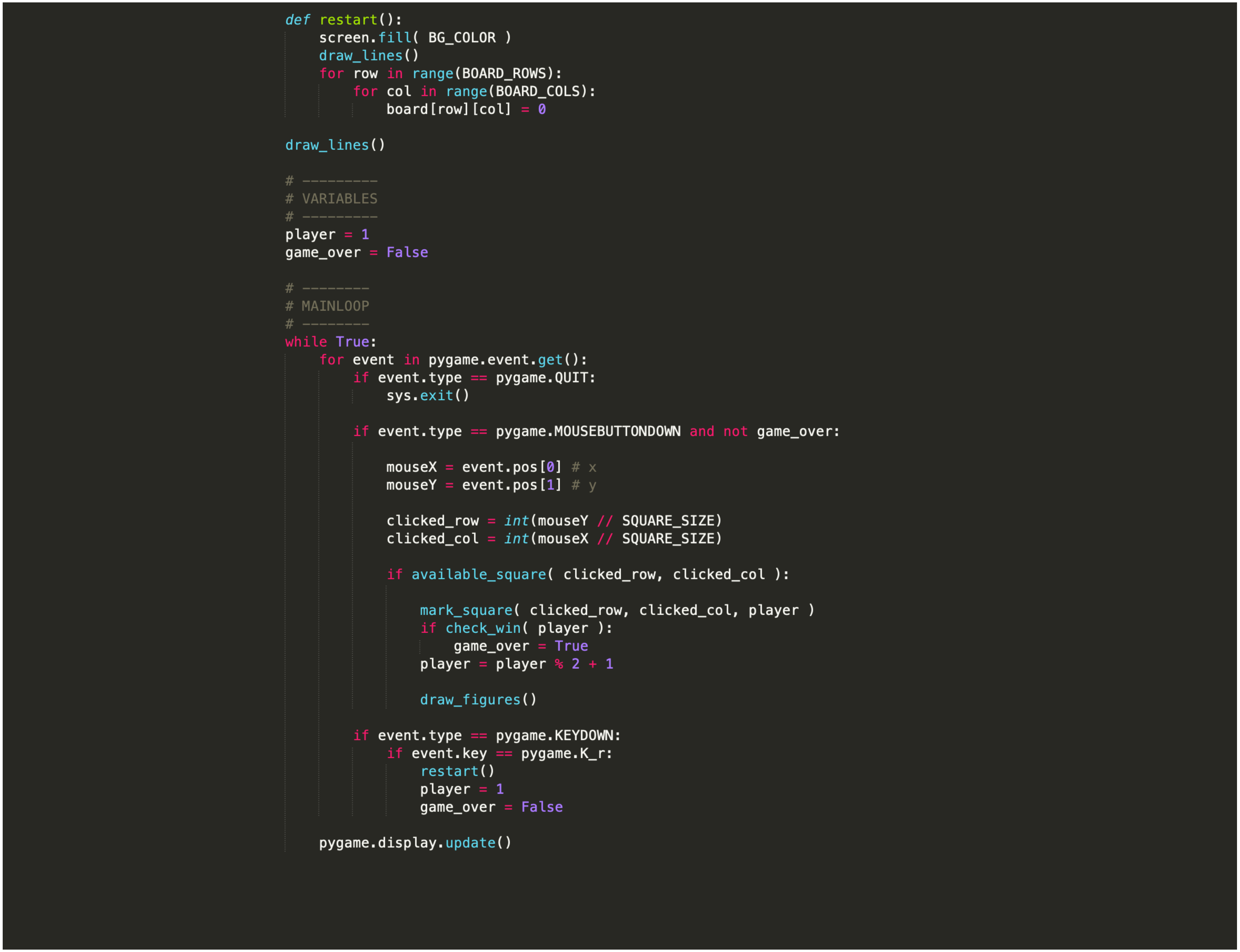
```
    # 1 vertical
    pygame.draw.line( screen, LINE_COLOR, (SQUARE_SIZE, 0), (SQUARE_SIZE, HEIGHT
        ), LINE_WIDTH )
    # 2 vertical
    pygame.draw.line( screen, LINE_COLOR, (2 * SQUARE_SIZE, 0), (2 * SQUARE_SIZE
        , HEIGHT), LINE_WIDTH )

def draw_figures():
    for row in range(BOARD_ROWS):
        for col in range(BOARD_COLS):
            if board[row][col] == 1:
                pygame.draw.circle( screen, CIRCLE_COLOR, (int( col *
                    SQUARE_SIZE + SQUARE_SIZE//2 ), int( row * SQUARE_SIZE +
                    SQUARE_SIZE//2 )), CIRCLE_RADIUS, CIRCLE_WIDTH )
            elif board[row][col] == 2:
                pygame.draw.line( screen, CROSS_COLOR, (col * SQUARE_SIZE +
                    SPACE, row * SQUARE_SIZE + SQUARE_SIZE - SPACE), (col *
                    SQUARE_SIZE + SQUARE_SIZE - SPACE, row * SQUARE_SIZE + SPACE
                    ), CROSS_WIDTH )
                pygame.draw.line( screen, CROSS_COLOR, (col * SQUARE_SIZE +
                    SPACE, row * SQUARE_SIZE + SPACE), (col * SQUARE_SIZE +
                    SQUARE_SIZE - SPACE, row * SQUARE_SIZE + SQUARE_SIZE - SPACE
                    ), CROSS_WIDTH )

def mark_square(row, col, player):
    board[row][col] = player

def available_square(row, col):
    return board[row][col] == 0

def is_board_full():
    for row in range(BOARD_ROWS):
        for col in range(BOARD_COLS):
            if board[row][col] == 0:
                return False

    return True

def check_win(player):
    # vertical win check
    for col in range(BOARD_COLS):
        if board[0][col] == player and board[1][col] == player and board[2][col]
             == player:
            draw_vertical_winning_line(col, player)
            return True

    # horizontal win check
    for row in range(BOARD_ROWS):
        if board[row][0] == player and board[row][1] == player and board[row][2]
             == player:
            draw_horizontal_winning_line(row, player)
            return True
```



```
    # horizontal win check
    for row in range(BOARD_ROWS):
        if board[row][0] == player and board[row][1] == player and board[row][2]
             == player:
            draw_horizontal_winning_line(row, player)
            return True

    # asc diagonal win check
    if board[2][0] == player and board[1][1] == player and board[0][2] == player
        :
        draw_asc_diagonal(player)
        return True

    # desc diagonal win chek
    if board[0][0] == player and board[1][1] == player and board[2][2] == player
        :
        draw_desc_diagonal(player)
        return True

    return False

def draw_vertical_winning_line(col, player):
    posX = col * SQUARE_SIZE + SQUARE_SIZE//2

    if player == 1:
        color = CIRCLE_COLOR
    elif player == 2:
        color = CROSS_COLOR

    pygame.draw.line( screen, color, (posX, 15), (posX, HEIGHT - 15), LINE_WIDTH
         )

def draw_horizontal_winning_line(row, player):
    posY = row * SQUARE_SIZE + SQUARE_SIZE//2

    if player == 1:
        color = CIRCLE_COLOR
    elif player == 2:
        color = CROSS_COLOR

    pygame.draw.line( screen, color, (15, posY), (WIDTH - 15, posY),
        WIN_LINE_WIDTH )

def draw_asc_diagonal(player):
    if player == 1:
        color = CIRCLE_COLOR
    elif player == 2:
        color = CROSS_COLOR

    pygame.draw.line( screen, color, (15, HEIGHT - 15), (WIDTH - 15, 15),
        WIN_LINE_WIDTH )
```



```python
def draw_desc_diagonal(player):
    if player == 1:
        color = CIRCLE_COLOR
    elif player == 2:
        color = CROSS_COLOR

    pygame.draw.line( screen, color, (15, 15), (WIDTH - 15, HEIGHT - 15),
        WIN_LINE_WIDTH )

def restart():
    screen.fill( BG_COLOR )
    draw_lines()
    for row in range(BOARD_ROWS):
        for col in range(BOARD_COLS):
            board[row][col] = 0

draw_lines()

# ---------
# VARIABLES
# ---------
player = 1
game_over = False

# --------
# MAINLOOP
# --------
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()

        if event.type == pygame.MOUSEBUTTONDOWN and not game_over:

            mouseX = event.pos[0] # x
            mouseY = event.pos[1] # y

            clicked_row = int(mouseY // SQUARE_SIZE)
            clicked_col = int(mouseX // SQUARE_SIZE)

            if available_square( clicked_row, clicked_col ):

                mark_square( clicked_row, clicked_col, player )
                if check_win( player ):
                    game_over = True
                player = player % 2 + 1

                draw_figures()

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_r:
                restart()
```



```python
def restart():
    screen.fill( BG_COLOR )
    draw_lines()
    for row in range(BOARD_ROWS):
        for col in range(BOARD_COLS):
            board[row][col] = 0

draw_lines()

# ---------
# VARIABLES
# ---------
player = 1
game_over = False

# --------
# MAINLOOP
# --------
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()

        if event.type == pygame.MOUSEBUTTONDOWN and not game_over:

            mouseX = event.pos[0] # x
            mouseY = event.pos[1] # y

            clicked_row = int(mouseY // SQUARE_SIZE)
            clicked_col = int(mouseX // SQUARE_SIZE)

            if available_square( clicked_row, clicked_col ):

                mark_square( clicked_row, clicked_col, player )
                if check_win( player ):
                    game_over = True
                player = player % 2 + 1

                draw_figures()

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_r:
                restart()
                player = 1
                game_over = False

    pygame.display.update()
```