In [1]:
```
import numpy as np
import pygame
import sys
import math
```

```
pygame 2.1.2 (SDL 2.0.18, Python 3.10.0)
Hello from the pygame community. https://www.pygame.org/contribute.html
```

In [2]:
```
PINK=(0,0,255)
GREEN=(0,0,0)
ORANGE =(255,0,0)
PURPLE = (255,255,0)

ROW_NO=6
COLUMN_NO =7
```

In [3]:
```
def make_board():
    board = np.zeros((ROW_NO,COLUMN_NO))
    return board
```

In [4]:
```
def drop_piece(board, row, col, piece):
    board[row][col] = piece
```

In [5]:
```
def accurate_location(board, col):
    return board[ROW_NO-1][col] == 0
```

In [6]:
```
def get_next_open_row(board, col):
    for r in range(ROW_NO):
        if board[r][col] == 0:
            return r
```

In [7]:
```
def print_board(board):
    print(np.flip(board, 0))
```

In [8]:
```
def winning_move(board, piece):
    # Need to check the horizontal location to  decide the winner of the  game
    for c in range(COLUMN_NO-3):
        for r in range(ROW_NO):
            if board[r][c] == piece and board[r][c+1] == piece and board[r][c+2] == piece and board[r][c+3] == piece:
                return True


    # Need to check the positively sloped diaganols to decide the game winner
    for c in range(COLUMN_COUNT-3):
        for r in range(ROW_COUNT-3):
            if board[r][c] == piece and board[r+1][c+1] == piece and board[r+2][c+2] == piece and board[r+3][c+3] == piece:
                return True

     # Need to check the vertical location to decide the winner of the game
    for c in range(COLUMN_NO):
        for r in range(ROW_NO-3):
            if board[r][c] == piece and board[r+1][c] == piece and board[r+2][c] == piece and board[r+3][c] == piece:
                return True

    # Need to check the Negatively sloped diaganols to decide the game winner
    for c in range(COLUMN_COUNT-3):
        for r in range(3, ROW_COUNT):
            if board[r][c] == piece and board[r-1][c+1] == piece and board[r-2][c+2] == piece and board[r-3][c+3] == piece:
                return True
```

In [9]:
```
def draw_board(board):
    for c in range(COLUMN_NO):
        for r in range(ROW_NO):
            pygame.draw.rect(screen, PINK, (c*SQUARESIZE, r*SQUARESIZE+SQUARESIZE, SQUARESIZE, SQUARESIZE))
            pygame.draw.circle(screen, GREEN, (int(c*SQUARESIZE+SQUARESIZE/2), int(r*SQUARESIZE+SQUARESIZE+SQUARESIZE/2)), RADIUS)
    for c in range(COLUMN_NO):
        for r in range(ROW_NO):
            if board[r][c] == 1:
                pygame.draw.circle(screen, ORANGE, (int(c*SQUARESIZE+SQUARESIZE/2), height-int(r*SQUARESIZE+SQUARESIZE/2)), RADIUS)
            elif board[r][c] == 2:
                pygame.draw.circle(screen, PURPLE, (int(c*SQUARESIZE+SQUARESIZE/2), height-int(r*SQUARESIZE+SQUARESIZE/2)), RADIUS)
    pygame.display.update()
```

In [10]:
```
board = make_board()
print_board(board)
game_over = False
turn = 0
```

```
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]]
```

In [11]:
```
# In this we initalise pygame
pygame.init()
# defiine our screen size
SQUARESIZE = 200
# Defined the width and height of thre board
width = COLUMN_NO * SQUARESIZE
height = (ROW_NO+1) * SQUARESIZE
size = (width, height)
RADIUS = int(SQUARESIZE/2 - 5)
screen = pygame.display.set_mode(size)
```

In [12]:
```
# Calling function is used to draw_board again
draw_board(board)
pygame.display.update()
myfont = pygame.font.SysFont("monospace", 75)
```

In [ ]:
```
while not game_over:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
        if event.type == pygame.MOUSEMOTION:
            pygame.draw.rect(screen, GREEN, (0,0, width, SQUARESIZE))
            posx = event.pos[0]
            if turn == 0:
                pygame.draw.circle(screen, ORANGE, (posx, int(SQUARESIZE/2)), RADIUS)
            else:
                pygame.draw.circle(screen, PURPLE, (posx, int(SQUARESIZE/2)), RADIUS)
        pygame.display.update()
```

In [ ]:
```
if event.type == pygame.MOUSEBUTTONDOWN:
            pygame.draw.rect(screen, GREEN, (0,0, width, SQUARESIZE))

            # print(event.pos)
             # Player 1 need to give input
            if turn == 0:
                posx = event.pos[0]
                col = int(math.floor(posx/SQUARESIZE))

                if accurate_location(board, col):
                    row = get_next_open_row(board, col)
                    drop_piece(board, row, col, 1)

                    if winning_move(board, 1):
                        label = myfont.render("Player 1 wins!!", 1, ORANGE)
                        screen.blit(label, (40,10))
                        game_over = True
```

In [ ]:
```
 # Player 2 needs to give input
            else:
                posx = event.pos[0]
                col = int(math.floor(posx/SQUARESIZE))

                if accurate_location(board, col):
                    row = get_next_open_row(board, col)
                    drop_piece(board, row, col, 2)

                    if winning_move(board, 2):
                        label = myfont.render("Player 2 wins!!", 1, PURPLE)
                        screen.blit(label, (40,10))
                        game_over = True
            print_board(board)
            draw_board(board)

            turn += 1
            turn = turn % 2

            if game_over:
                 pygame.time.wait(3000)
```