```python
# Title: Programming Assignment 2
# Due date: Wednesday, September 9, 2021 at 11:59pm
# Author: Sotheanith Sok
# Description: Remove connected components whos area is less than P pixels. Mathlab version of bwareaopen assumes that 1 is connected components and 0 is background.

# ------------------------------------------------------------------------------
# imports
import numpy as np

def bwareaopen(BW, P):
    """Removes all connected components (objects) that have fewer than P pixels from the binary image BW.

    Args:
        BW (array): binary image.
        P (int): maximum number of pixels in objects, specified as a nonnegative integer.

    Returns:
        [array]: binary image
    """
    # Find connected components by looping through all pixels that hasn't been visited.
    rows = np.shape(BW)[0]
    cols = np.shape(BW)[1]
    tag = 2
    for row in range(rows):
        for col in range(cols):
            if BW[row, col] == 1:
                BW = _find_connected_components(BW, row, col, tag)
                tag = tag + 1

    # Remove connected componets that contains less than P pixels.
    for component in range(2, tag):
        pixels = np.count_nonzero(BW == component)
        if pixels < P:
            BW[BW == component] = 0
        else:
            BW[BW == component] = 1

    return BW


def _find_connected_components(BW, initial_row, initial_col, tag):
    """Perform non-recursive flooding algorithm to find all pixels connected to a component.

    Args:
        BW (array): binary image.
        initial_row (int): starting row index.
        initial_col (int): starting column index.
        tag (int): tag used to identify this connected component.

    Returns:
        [array]: binary image with tagged area of this connected component
    """
    #Add initial row and col to a set of unvisted pixels (set is desired since we don't want duplicated unvisted pixels).
    unvisted_pixels = set()
    unvisted_pixels.add((initial_row, initial_col))

    #Loop through all unvisted pixels
    while len(unvisted_pixels) > 0:

        #Remvove the first unvisited pixel from the set
        row, col = unvisted_pixels.pop()

        #Tag the pixel
        BW[row, col] = tag

        # Add unvisted neighboring pixels to the set
        # # Top left
        if row > 0 and col > 0 and BW[row - 1, col - 1] == 1:
            unvisted_pixels.add((row - 1, col - 1))
        # Top
        if row > 0 and BW[row - 1, col] == 1:
            unvisted_pixels.add((row - 1, col))
        # Top right
        if row > 0 and col < np.shape(BW)[1] - 1 and BW[row - 1, col + 1] == 1:
            unvisted_pixels.add((row - 1, col + 1))
        # Left
        if col > 0 and BW[row, col - 1] == 1:
            unvisted_pixels.add((row, col - 1))
        # Right
        if col < np.shape(BW)[1] - 1 and BW[row, col + 1] == 1:
            unvisted_pixels.add((row, col + 1))
        # Bottom left
        if row < np.shape(BW)[0] - 1 and col > 0 and BW[row + 1, col - 1] == 1:
            unvisted_pixels.add((row + 1, col - 1))
        # Bottom
        if row < np.shape(BW)[0] - 1 and BW[row + 1, col] == 1:
            unvisted_pixels.add((row + 1, col))
        # Bottom right
        if (
            row < np.shape(BW)[0] - 1
            and col < np.shape(BW)[1] - 1
            and BW[row + 1, col + 1] == 1
        ):
            unvisted_pixels.add((row + 1, col + 1))

    return BW
```