```
import numpy as np

from src.region import Region


class Rectangle(Region):
    def __init__(self, shape=np.array([[0, 0], [0, 1], [1, 1], [1, 0]])):
        super().__init__(shape)
        self.x_int = [shape[:, 0].min(), shape[:, 0].max()]
        self.y_int = [shape[:, 1].min(), shape[:, 1].max()]

    def _flatten(self, n):
        self.x_coords = np.linspace(self.x_int[0], self.x_int[1], n)
        self.y_coords = np.linspace(self.y_int[0], self.y_int[1], n)

        self.x_coords = np.tile(self.x_coords, n)
        self.y_coords = np.repeat(self.y_coords, n)
        # n += 1
        #
        # def half_step(interval):
        #     return (interval[1] - interval[0]) / (n - 1) / 2
        #
        # def get_x(interval):
        #     first_layer = np.linspace(interval[0], interval[1], n)
        #     hs = half_step(interval)
        #     second_layer = np.linspace(interval[0] + hs, interval[1] + hs, n - 1, endpo
int=False)
        #
        #     second_layer = np.append(second_layer, first_layer[0])
        #     second_layer = np.append(second_layer, first_layer[-1])
        #
        #     segment = np.append(first_layer, second_layer)
        #     net_x = np.tile(segment, n - 1)
        #     return np.append(net_x, first_layer)
        #
        # def get_y(interval):
        #     net_y = np.linspace(interval[0], interval[1], 2*n - 1)
        #     first = np.repeat(net_y[::2], n)
        #     second = np.repeat(net_y[1::2], n + 1)
        #     return np.sort(np.append(first, second))
        #
        # def get_coords(x_int, y_int):
        #     xc = get_x(x_int)
        #     yc = get_y(y_int)
        #     return xc, yc
        #
        # # longest edge should have more points
        # if (self.x_int[1] - self.x_int[0]) > (self.y_int[1] - self.y_int[0]):
        #     self.y_coords, self.x_coords = get_coords(self.y_int, self.x_int)
        # else:
        #     self.x_coords, self.y_coords = get_coords(self.x_int, self.y_int)
```