Daily Coding Problem #187

Problem

This problem was asked by Google.

You are given given a list of rectangles represented by min and max x- and y-coordinates. Compute whether or not a pair of rectangles overlap each other. If one rectangle completely covers another, it is considered overlapping.

For example, given the following rectangles:

```
{
    "top_left": (1, 4),
    "dimensions": (3, 3) # width, height
},
{
    "top_left": (-1, 3),
    "dimensions": (2, 1)
},
{
    "top_left": (0, 5),
    "dimensions": (4, 3)
}
```

return `true` as the first and third rectangle overlap each other.

Solution

We can check every pairing of rectangles to see if they overlap, and return `true` if we find one. If not, then return `false`.

```
def overlapping(rectangles):
    for i, rec1 in enumerate(rectangles):
```

```
        for rec2 in rectangles[i + 1:]:
            if is_overlapping(rec1, rec2):
                return True

    return False
```

How do we calculate whether two rectangles overlap? It's easier to check if they *don't* intersect:

- If `rec1`'s left border is right of `rec2`'s right border
- Or `rec1`'s right border is left of `rec2`'s left border
- Or `rec1`'s top border is below `rec2`'s bottom border
- Or `rec1`'s bottom border is above of `rec2`'s top border

If any of these cases are met, then the rectangles don't overlap, and we can return `false`:

```
def is_overlapping(rec1, rec2):
    if rec1["top_left"][0] >= rec2["top_left"][0] + rec2["dimensions"][0]:
        return False

    if rec1["top_left"][0] + rec1["dimensions"][0] <= rec2["top_left"][0]:
        return False

    if rec1["top_left"][1] <= rec2["top_left"][1] - rec2["dimensions"][1]:
        return False

    if rec1["top_left"][1] - rec1["dimensions"][1] >= rec2["top_left"][1]:
        return False

    return True
```

Since each `is_overlapping` call is O(1), this whole code takes $O(n^2)$ time.