Daily Coding Problem #77

Problem

This problem was asked by Snapchat.

Given a list of possibly overlapping intervals, return a new list of intervals where all overlapping intervals have been merged.

The input list is not necessarily ordered in any way.

For example, given [(1, 3), (5, 8), (4, 10), (20, 25)], you should return [(1, 3), (4, 10), (20, 25)].

Solution

We can do this by sorting all the intervals by their start time. This way, when looking at the current interval, if it overlaps with the previous one we can just combine them.

```
def merge(intervals):
    result = []
    for start, end in sorted(intervals, key=lambda i: i[0]):
        # If current interval overlaps with the previous one, combine them
        if result and start <= result[-1][1]:
            prev_start, prev_end = result[-1]
            result[-1] = (prev_start, max(end, prev_end))
        else:
            result.append((start, end))
    return result
```

Since we have to sort the intervals, this runs in O(N log N) time.