# #LeetCode57

https://leetcode.com/problems/insert-interval/description/

Yifeng Zeng

# #题目描述

Insert Interval

# #思路报告

因为intervals已经排了序了，所以我们只需要把newInterval的start和end在这些intervals里面对应的位置找到就行了。由于是从小到大排序的，我们先看newInterval.start，我们遍历intervals，找到某一个intervals.get(i)，当i.end < newInterval.start是我们把直接放到result里面，因为不会跟newInterval有overlap。这时我们有了第一个newInterval.start < i.end的情况，那么我们要考虑newInterval.start跟i.start的大小，两者取更小的作为newInterval的start即可。然后我们要找到这个newInterval跟哪些intervals.get(i)相overlap，那么其实只需要找到最后一个i.start小于等于newInterval.end的interval即可，那么更新newInterval.end为newInterval.end和i.end两者更大的即可。剩下后面的i.start都比newInterval.end要大，也不会有overlap，所以先把newInterval放进result里面，再把后面所有剩下的intervals放进result里面就可以了。

代码如下：

```
public List<Interval> insert(List<Interval> intervals, Interval newInterval) {
    List<Interval> result = new ArrayList<>();

    int i = 0;
    while (i < intervals.size() && intervals.get(i).end < newInterval.start) {
        result.add(intervals.get(i));
        i++;
    }
    if (i < intervals.size()) {
        newInterval.start = Math.min(newInterval.start, intervals.get(i).start);
    }
    while (i < intervals.size() && intervals.get(i).start <= newInterval.end) {
        newInterval.end = Math.max(newInterval.end, intervals.get(i).end);
        i++;
    }
    result.add(newInterval);
```

```
        while (i < intervals.size()) {
            result.add(intervals.get(i));
            i++;
        }
        return result;
    }
```

# #套路总结

- 其实这个题就是对于一个复杂的问题拆分成多个简单问题的例子，每一个步骤单独看起来其实很简单，但是不容易从复杂问题想到，而且有一些corner case需要考虑，比如intervals的size为0的情况。