[Meeting Rooms II](https://leetcode.com/problems/meeting-rooms-ii/)

**import** java.util.Arrays;

**import** java.util.PriorityQueue;

**public** **class** MeetingRooms {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int**[][] intervals = {{0 , 30},{5, 10},{15, 20}};

System.***out***.println(*minMeetingRooms*(intervals));

}

**public** **static** **int** minMeetingRooms(**int**[][] intervals) {

**if**(intervals == **null** || intervals.length == 0 || intervals[0].length == 0) {

**return** 0;

}

PriorityQueue<Integer> queue = **new** PriorityQueue<>((p , q) -> p - q); // min heap for maintaining interval end times

Arrays.*sort*(intervals, (a , b) -> a[0] - b[0]); //sorting start time in ascending order

queue.add(intervals[0][1]);

**for**(**int** i = 1 ; i < intervals.length ; i++) {

**if**(queue.peek() <= intervals[i][0]) { //check if the current interval's start time is after the min end time of any interval

queue.poll();

}

queue.add(intervals[i][1]);

}

**return** queue.size();

}

}

Time complexity : O(nlogn + n). n is length of intervals

Space Complexity : O(n), to store end time of each interval