452. Minimum Number of Arrows to Burst Balloons

Medium 6.5K 187

Companies

There are some spherical balloons taped onto a flat wall that represents the XY-plane. The balloons are represented as a 2D integer array `points` where `points[i] = [x_start, x_end]` denotes a balloon whose **horizontal diameter** stretches between `x_start` and `x_end`. You do not know the exact y-coordinates of the balloons.

Arrows can be shot up **directly vertically** (in the positive y-direction) from different points along the x-axis. A balloon with `x_start` and `x_end` is **burst** by an arrow shot at `x` if `x_start <= x <= x_end`. There is **no limit** to the number of arrows that can be shot. A shot arrow keeps traveling up infinitely, bursting any balloons in its path.

Given the array `points`, return *the **minimum** number of arrows that must be shot to burst all balloons.*

Code

```swift
class Solution {
    func findMinArrowShots(_ intt: [[Int]]) -> Int {

        let points = intt.sorted{ $0[1] < $1[1] }

        var arrows = 0
        var currentEnd = Int.min

        for point in points {
            if currentEnd < point[0] {
                arrows += 1
                currentEnd = point[1]
            }
        }

        return arrows
```

```
    //     let intervals = intt.sorted(by: {$0[0] < $1[0]})
    //     print(intervals)
    // var res:[[Int]] = [intervals[0]]
    // for i in (1..<intervals.count) {
    //     let temp = res.last!
    //     if checkIfOverlap(temp,intervals[i]) {
    //         let b = intervals[i]
    //         let front = min(temp[0],b[0])
    //         let back = min(temp[1],b[1])
    //         res[res.count-1] = [front,back]
    //         // res.append(intervals[i+1])
    //     } else {
    //         res.append(intervals[i])
    //     }
    // }
    // print(res)
    // return res.count

    }

func checkIfOverlap(_ a:[Int], _ b:[Int]) -> Bool {
        let front = max(a[0],b[0])
        let back = min(a[1],b[1])
        return back-front >= 0 ? true : false
    }
}
```