GREEDY ALGORITHM

CONCEPT

- Greedy v.s. DP
  - Greedy
    - Current state only refers to the current situation
    - Lead to local optimal
  - DP
    - Current state may refer to the previous states
    - Lead to global optimal

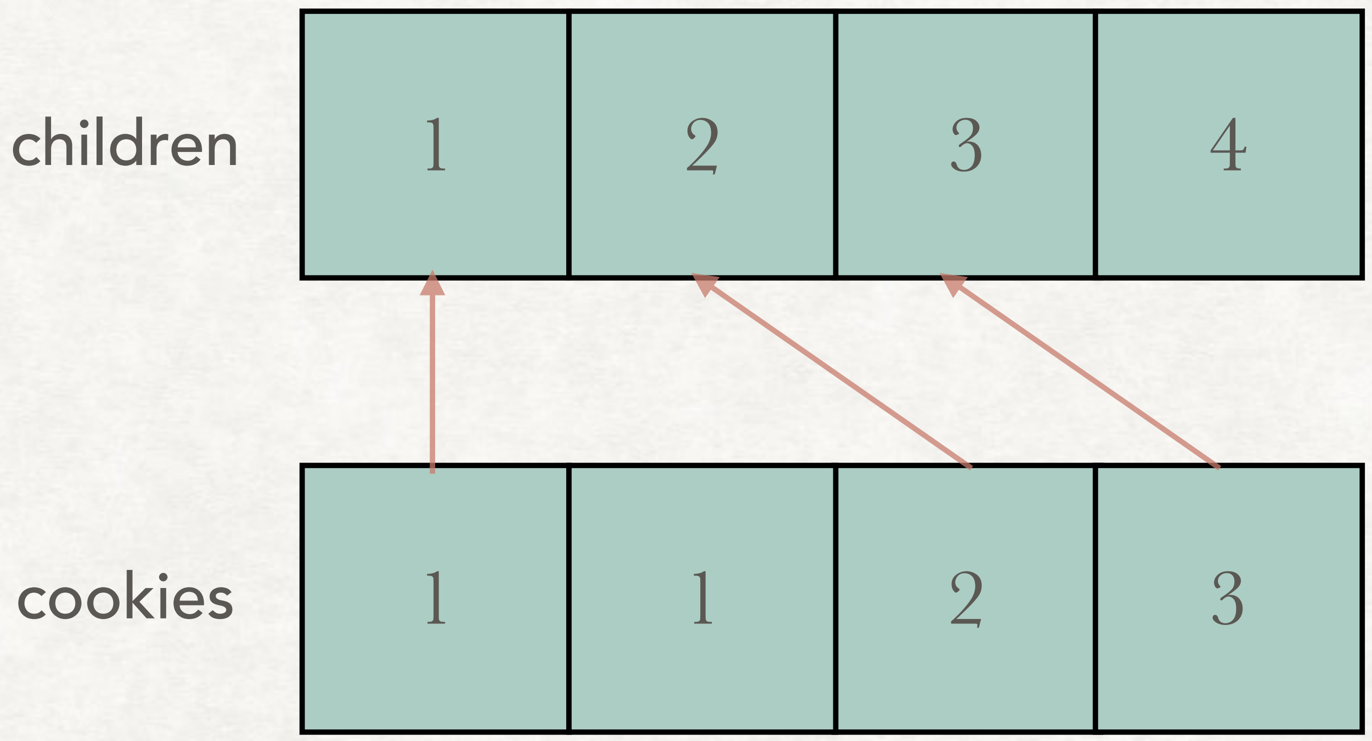
NO. 455 ASSIGN COOKIES

PROBLEM DESCRIPTION

- Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie. Each child i has a greed factor gi, which is the minimum size of a cookie that the child will be content with; and each cookie j has a size sj. If sj >= gi, we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

IDEA

- Sorting
- Maintain index for children



SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/455_Assign_Cookies

NO. 435 NON-OVERLAPPING INTERVALS

PROBLEM DESCRIPTION

- Given a collection of intervals, find the minimum number of intervals you need to remove to make the rest of the intervals non-overlapping.
- Note:
  - You may assume the interval's end point is always bigger than its start point.
  - Intervals like [1,2] and [2,3] have borders "touching" but they don't overlap each other.

IDEA

- Sorting
- interval1[start:end] and interval2[start:end]
  - interval2.start must >= interval1.end *// previous end*
  - Special cases
    - Given interval1[start: end] and interval2[start, end]
    - interval1.start = interval2.start
    - interval1.end > interval2.end

update previous end!

SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/435_Non-overlapping_Intervals

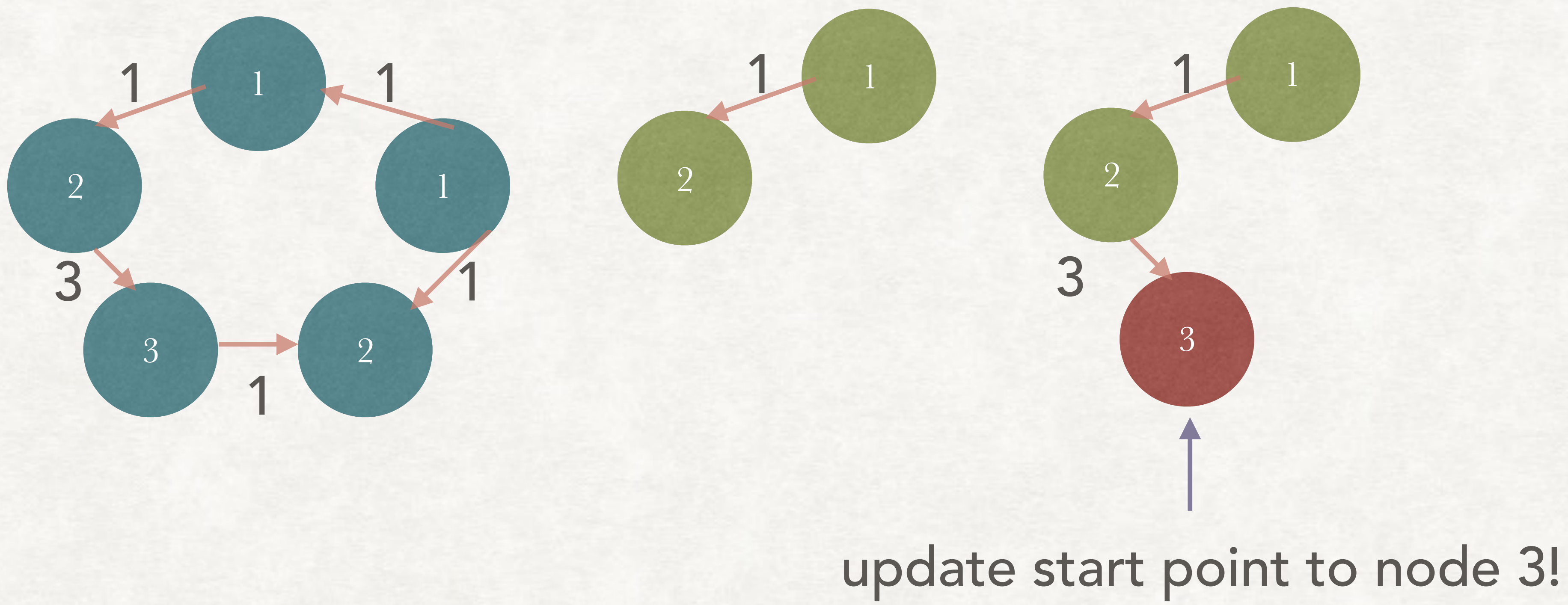
NO. 134
GAS STATION

PROBLEM DESCRIPTION

- There are N gas stations along a circular route, where the amount of gas at station i is gas[i].
- You have a car with an unlimited gas tank and it costs cost[i] of gas to travel from station i to its next station (i+1). You begin the journey with an empty tank at one of the gas stations.
- Return the starting gas station's index if you can travel around the circuit once, otherwise return -1.

IDEA

- If A cannot reach C, then every B that A can reach cannot reach C
- If sum(gas) > sum(cost), there is MUST at least one solution



SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/134_Gas_Station

NO. 122
BEST TIME TO BUY AND SELL STOCK II

PROBLEM DESCRIPTION

- Say you have an array for which the ith element is the price of a given stock on day i.
- Design an algorithm to find the maximum profit. You may complete as many transactions as you like (ie, buy one and sell one share of the stock multiple times). However, you may not engage in multiple transactions at the same time (ie, you must sell the stock before you buy again).

IDEA

- No cool down, we don’t have to maintain 2 sequences (buy & sell)
- Incremental add the profit (price[i] - price[i-1])

SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/122_Best_Time_to_Buy_and_Sell_Stock_II

NO. 452
MINIMUM NUMBER OF ARROWS TO BURST BALLOONS

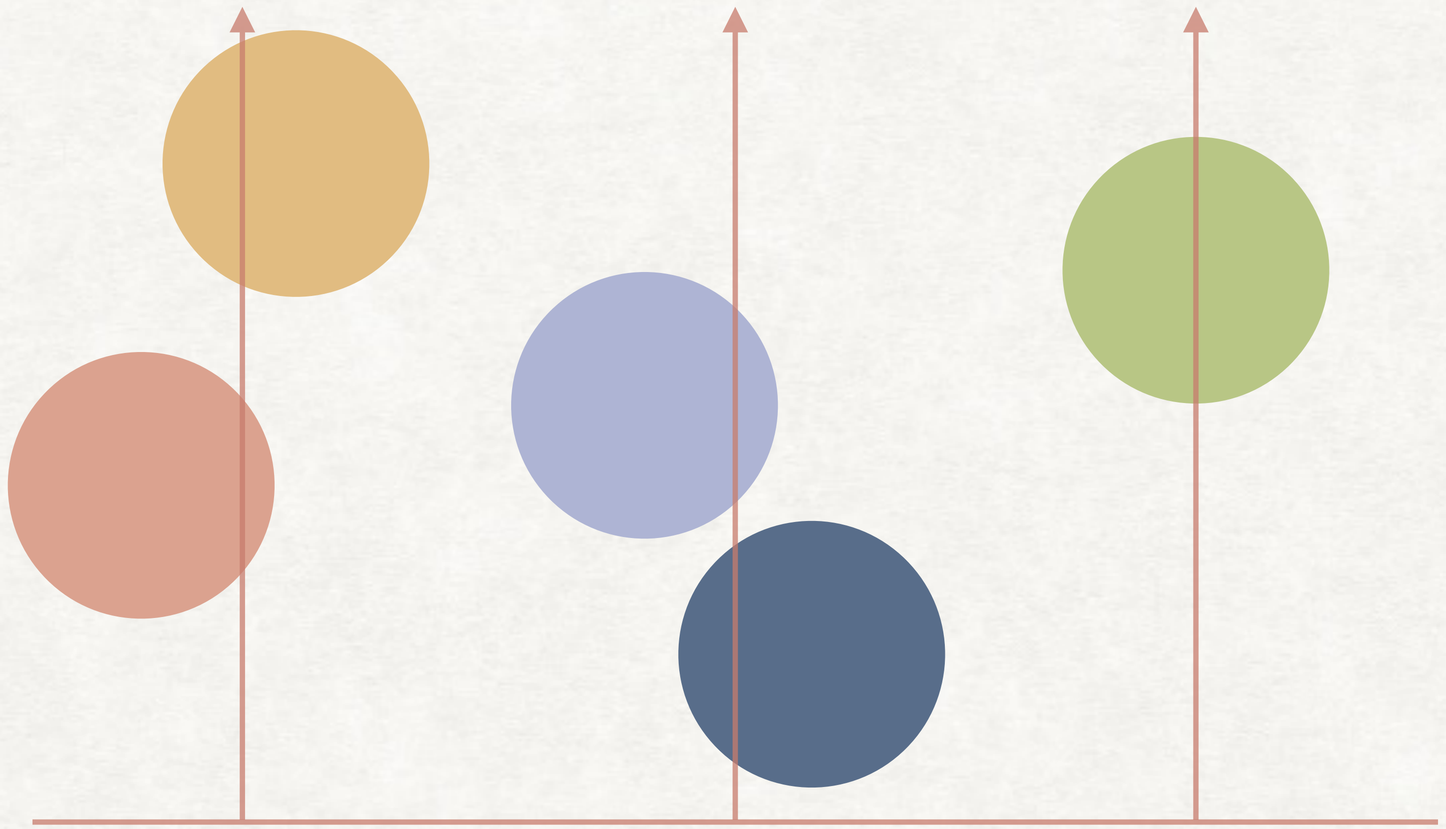
PROBLEM DESCRIPTION

- There are a number of spherical balloons spread in two-dimensional space. For each balloon, provided input is the start and end coordinates of the horizontal diameter. Since it's horizontal, y-coordinates don't matter and hence the x-coordinates of start and end of the diameter suffice. Start is always smaller than end. There will be at most 104 balloons.

- An arrow can be shot up exactly vertically from different points along the x-axis. A balloon with xstart and xend bursts by an arrow shot at x if xstart ≤ x ≤ xend. There is no limit to the number of arrows that can be shot. An arrow once shot keeps travelling up infinitely. The problem is to find the minimum number of arrows that must be shot to burst all balloons.

PROBLEM DESCRIPTION

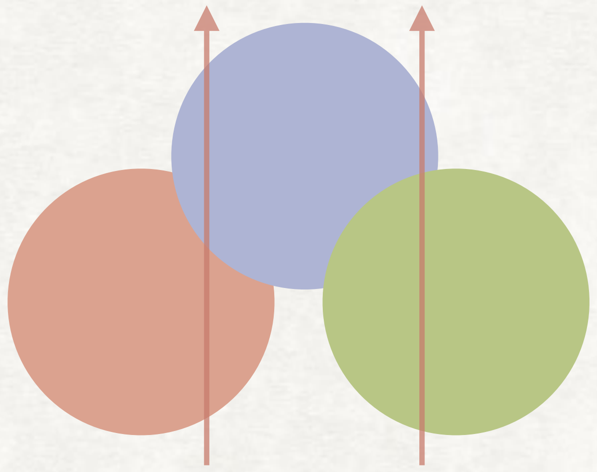
- Figure



return 3

IDEA

- Finding the number of non-overlapping intervals
- Sorting
- number of balloons - overlapping balloons
- Special cases



update end coordinate

dp[i].end = min(dp[i-1].end, dp[i].end)

SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/452_Minimum_Number_of_Arrows_to_Burst_Balloons

NO. 376
WIGGLE SUBSEQUENCE

PROBLEM DESCRIPTION

- A sequence of numbers is called a wiggle sequence if the differences between successive numbers strictly alternate between positive and negative. The first difference (if one exists) may be either positive or negative. A sequence with fewer than two elements is trivially a wiggle sequence.
- For example, [1,7,4,9,2,5] is a wiggle sequence because the differences (6,-3,5,-7,3) are alternately positive and negative. In contrast, [1,4,7,2,5] and [1,7,4,5,5] are not wiggle sequences, the first because its first two differences are positive and the second because its last difference is zero.
- Given a sequence of integers, return the length of the longest subsequence that is a wiggle sequence. A subsequence is obtained by deleting some number of elements (eventually, also zero) from the original sequence, leaving the remaining elements in their original order.

IDEA

- Difference must be positive and negative alternatively and repeatedly
- Using DP concept is more straightforward to me
    - dp[i] = dp[i-1] + 1 if wiggle else dp[i-1]

SOLUTION

- https://github.com/Brady31027/leetcode/tree/master/376_Wiggle_Subsequence