# Spring 24 Div II Week 2 - Solutions

(taken from editorials of original problems)

## Problem A

(Source: Original Problem)

Sort the integers using an efficient sorting algorithm (the built in sort from your favorite language is perfect). Now all the repeated elements are in consecutive positions. Iterate through the elements one by one and append each to a new list/array if it differs from the previous element.

## Problem B

(Source: Northwestern European Regional Programming Contest 2020 Problem C)

**Problem**

For $n$ numbers between 0 and 100 you are given the average of all numbers ($d$), and the average of a subset of $k$ of those numbers ($s$). Compute the average of the remaining numbers.

**Solution**

- The sum of all numbers is $d \cdot n$.
- So the sum of the remaining numbers is $d \cdot n - s \cdot k$.
- That parts contains $n - k$ numbers, so the average of those numbers is $(d \cdot n - s \cdot k)/(n - k)$.
- When the average is $< 0$ or $> 100$, print impossible.

**Gotchas**

- Precision issues, e.g. answers just below 0 or just above 100

## Problem C

(Source: Nordic Collegiate Programming Contest 2013 Problem A)

**Problem**

Find an optimal planting schedule to minimize the earliest date when all the trees are mature.

**Insight and Solution**

- **Insight:** It is optimal to plant the trees in descending order of growth times.
  - **Proof**: If any two trees are not in this order, then swapping them cannot increase the result.
- **Solution:** Sort the trees by decreasing growth times, and output $\max(\text{position} + \text{growth time} + 1)$.
- **Time complexity:** $O(n \log n)$.

# Problem D

(Source: CSES Problem Set: Sorting and Searching)

Binary search :We check all the possible outcomes between 0 and 1e18 using binary search. Here the possible outcome can't cross 1e9 so we must take the minimum of sum and 1e9.Similar type of problem:get the container

# Problem E

(Source: UK & Ireland Programming Contest 2017 Problem F)

- Dynamic programming state: {**head_count**, **flips_left**}
- If we have no flips left, the answer is the number of heads we have
  - f(h, 0) = h
- If at least one tail is left, flipping it gives a 50% chance of 1 extra head, or a 50% chance of nothing changing.
  - f(h, k) = 0.5 * f(h+1,k-1) + 0.5 * f(h, k-1)
- Otherwise, it's necessary to flip a head and have a 50% chance of *reducing* the score.
  - f(N, k) = 0.5 * f(N,k-1) + 0.5 * f(N-1,k-1)
  - f(N, k) = N - 0.5

# Problem F

(Source: German Collegiate Programming Contest 2018 Problem C)

**Problem**

Given a graph of ski slopes labelled with *condition measures*, find a path with maximal sum of these measures.

**Solution**

- The graph of ski slopes is a *directed acyclic graph*.
- Find the *longest path* between any two nodes.
- Multiple graph algorithms can be used for this:
  - Invert the measures, use Floyd-Warshall ($\mathcal{O}(n^3)$).
  - Add a super source, invert the measures, then use Bellman-Ford for shortest paths ($\mathcal{O}(n * m)$).
  - Find a topological ordering, then find the maximum for each node with dynamic programming ($\mathcal{O}(n + m)$).