# 2024 Pacific Northwest Division 1 Solutions

The Judges

November 16, 2024

# Auto-Coin-o-Matic

## Problem

- You are given a set of $n$ coin denominations and $m$ queries. Each query involves either counting the minimum number of coins needed to make exactly $v$, or removing a coin denomination from all future queries.

## Observation

- Consider the reverse process. The problem turns into adding a coin denomination and counting the minimum number of coins.
- Adding a new denomination is significantly easier!

# Auto-Coin-o-Matic

## Solving the Static Version

- Define $f(i, v)$ to be the minimum number of coins needed to make exactly $v$ given that you only use the first $i$ denominations available. This can be computed with dynamic programming. If we can use the coin denomination, then $f(i, v) \leq 1 + f(i, v - c_i)$ where $c_i$ is the value of the $i$-th coin. If we do not use it, then $f(i, v) \leq f(i - 1, v)$.

# Auto-Coin-o-Matic

## Solving the Original Version

- Reverse the order of the queries and sort the coins by when they are added.
- Solve the static version of the problem for all coin denominations.
- At query time, compute how many denominations are available and reference that row in the DP table.

# Big Integers

## Problem

- Given two base-62 integers, compute which one is smaller.

## Solution

- If the integers are different lengths, the one that is shorter is smaller.
- Otherwise, compare the characters by their base-62 values and find the first mismatch, then compare those directly.

# Champernowne Subsequence

## Problem

- Compute the smallest positive integer $k$ such that, if you write the first $k$ positive integers in order, a given string of $n$ digits appears as a subsequence.

## Solution

- The answer for a string of length $n$ is at most $10n$.
- If $k$ is valid, then $k+1$ is valid, so we can binary search for the answer.
- Alternatively, we can just iteratively write the integers from 1 in order and keep track of how many of the first $d$ digits we have written, and stop once we have written down all $n$ digits.

# Connecting Computers

## Problem

- You are given a connected cactus graph on $n$ vertices where each edge is colored one of $k$ colors.
- Count the number of subsets of colors such that, considering all edges in the subset, the graph is still connected. On top of that, compute the cardinality of the smallest such subset.

## Cactus Observations

- Since the graph is a cactus, either an edge is part of a cycle, or it is not.
- If an edge is not part of a cycle, that color must be used.
- If two or more edges of the same color are in the same cycle, that color must be used.
- If an edge of a given color appears exactly once in the same cycle, then if that color is not used, all other colors in that cycle must be used.

# Connecting Computers

## Solution

- Identify whether each edge is part of a cycle or not and do the above analysis. Precompute for each color, if it is not included, what other colors must be used.
- For a given subset of colors, we can now validate in $\mathcal{O}(k)$ time whether it is valid.
- We can actually compute this for all subsets in $\mathcal{O}(2^k)$ by looping over subsets in nonincreasing size order.

# Exact Change

## Problem

- Given two integers $a$ and $b$, compute the smallest cardinality multiset of integer powers of two such that every integer from $a$ to $b$ can be expressed as the sum of some subset of integers from the multiset.

## Observation

- It is never optimal to have two coins of exactly the same denomination.

## Solution

- If $a = b = 0$, the answer is zero.
- Consider the largest power of two less than or equal to $b$, $2^x$.
- If $2^x \geq a$, then we can recursively solve this problem for the range $a - 2^x$ to $b - 2^x$.
- Otherwise, $2^x - 1$ must be representable, so the answer is $x + 1$ and the set $2^0, \ldots, 2^x$ suffices.

# Free Solo

## Problem

- Given $n$ unit circles in the plane, find the length of the shortest path from the origin to a specific unit circle.

## Solution

- We reduce this to solving a shortest path on critical points.
- We define a point to be critical if it is an intersection point of two circles, or if it is an intersection point between the destination unit circle and a line segment from an existing critical point to the center of the destination unit circle.
- There are $\mathcal{O}(n^2)$ vertices to consider, and $\mathcal{O}(n^2)$ candidate edges to consider. Each edge's validity can be confirmed in $\mathcal{O}(n \log n)$ by line sweeping over the edge.

# GCD Pairs

## Problem

- Given $n$ integers, each at most $10^{12}$, compute the number of pairs that have a squarefree GCD.

## Solution

- We convert each integer from $n$ to $x$, where $x^2$ is the maximal integer that divides $n$. We can compute $x$ in $\sqrt[3]{10^{12}}$ by naive trial division up to the cube root of the integer, then checking if the remaining part of the integer is a perfect square.
- We then need to check, given these $n$ integers, how many pairs are relatively prime. This is solvable by inclusion-exclusion.

# Homework Help

## Problem

- There is a secret permutation of $n$ integers. In a single query, you can request the number of inversions of a given subarray. Given the ability to make $n$ queries, compute the length of the longest increasing subsequence of the array.

## Solution

- Count the number of inversions for every prefix of the array. With this information, the exact array can be constructed exactly.
- We can then compute the length of the longest increasing subsequence in $\mathcal{O}(n^2)$. This can be done by maintaining the length of the longest increasing subsequence ending at a given index and computing all of these from left to right.

# Intuitive Elements

## Problem

- Given two strings $s$ and $t$, determine if every letter in $t$ appears in $s$.

## Solution

- Due to the low bounds, it's possible to naively loop over every character in $t$ and loop over every character in $s$ to check if it is present in both strings.
- For a faster solution, you can use a data structure like a set to track which characters exist in $s$.

# Rainbow Bowl Ranges

## Problem

- There is a circle of $n$ bowls and you have some balls in one of $m$ colors. For each color, pick a contiguous range of bowls and put one ball in each bowl in the range. A bowl is a rainbow bowl if every color of ball is in the bowl. A rainbow bowl range is a maximal contiguous range of rainbow bowls. Compute the maximum number of rainbow bowl ranges that can be formed given the number of bowls of each colors.

## Solution

- Since the colors must be contiguous, we can instead maintain the number of bowls that do not contain that color.
- If all colors appear $n$ times, then the answer is obviously 1.
- Otherwise, an optimal arrangement involves using the least common color and then the $k$ most common colors, such that the sum of the gaps, plus $k + 1$, is less than or equal to $n$.

# Sleeping on the Train

## Problem

- A train continuously travels from station 1 to station $n$ and back. Given that the train is found between stations $s$ and $s+1$ at different points in time, and someone boarded the train at station $a$ and disembarked at station $b$, compute the minimum number of times the train turns around at either station 1 or station $n$.

## Solution

- Carefully keep track of whether the train is moving towards increasing station numbers or decreasing station numbers.
- Every time there is an inconsistency found, increment the number of times the train must have turned around by 1 and change the direction of the train.

# Training, Round 3

## Problem

- Let $f(n, p)$ be all sets of $p$ distinct integers where each integer is between 1 and $n$.
- Generate $k$ sets independently and uniformly at random from $f(n, p)$. Compute the exact probability that there exists a pair of sets that shares some integer.

## Solution

- Note that $|f(n,p)| = \binom{n}{p}$, so the number of distinct possibilities for all $k$ sets is $\left(\binom{n}{p}\right)^k$.
- We will compute the complementary probability, namely that each integer appears at most once among the $k$ sets.
- There are $\binom{n}{p}$ valid sets for the first set. In general, there are $\binom{n-(i-1)p}{p}$ valid sets for the $i$th set. This gives us a closed form for the number of valid $k$-tuples.
- If we precompute all factorials up to $10^7$, along with their inverses, we can compute binomial coefficients in $\mathcal{O}(1)$.

# Word Game

## Problem

- Given $n$ words, and a merge operation where two strings $s$ and $t$ can be combined if there exists a length-$k$ suffix of $s$ that is a length-$k$ prefix of $t$, compute the lexicographically smallest word that can be generated by merging pairs of words until only one word is left.

## Solution

- Due to the small value of $n$ and the short length of strings, a recursive brute force solution that checks all possible ways of merging is fast enough.
- Note that if multiple values of $k$ are valid, you must try all of them!