Algolympics 2019

Solution Sketches

Problem I: Wax

- Sol. 1: Lots of if-else cases.
  - Not recommended; prone to bugs.

Problem I: Wax

- Sol. 2: Loops:

```
int d, ans = 0;
for (int i = 0; i < 4; i++)
  for (int j = 0; j < 4; j++)
    d = a[j] - a[i],
    ans = max(ans, d*d);
```

Problem I: Wax

- Sol. 3: Think about it some more.
  - **$(\max - \min)^2$**
- Derivable from parabola-ness of $(x - y)^2$.

Problem A: Quantum...

- No-thinking solution: $n \leq 20$, so just brute-force.
  - $O(n2^n)$

Problem A: Quantum...

- Think about it some more.
- Every selection

  `0 + a[1] + a[2] - a[3] - a[4] + … - a[n]`
- is cancelled by its negation

  `0 - a[1] - a[2] + a[3] + a[4] - … + a[n]`
- so everything cancels!

Problem A: Quantum…

- Think about it some more.
- Every selection

  `0 + a[1] + a[2] - a[3] - a[4] + … - a[n]`
- is cancelled by its negation

  `0 - a[1] - a[2] + a[3] + a[4] - … + a[n]`
- so everything cancels!
- Just **print 0.**

Problem J: Does It Spark Joy?

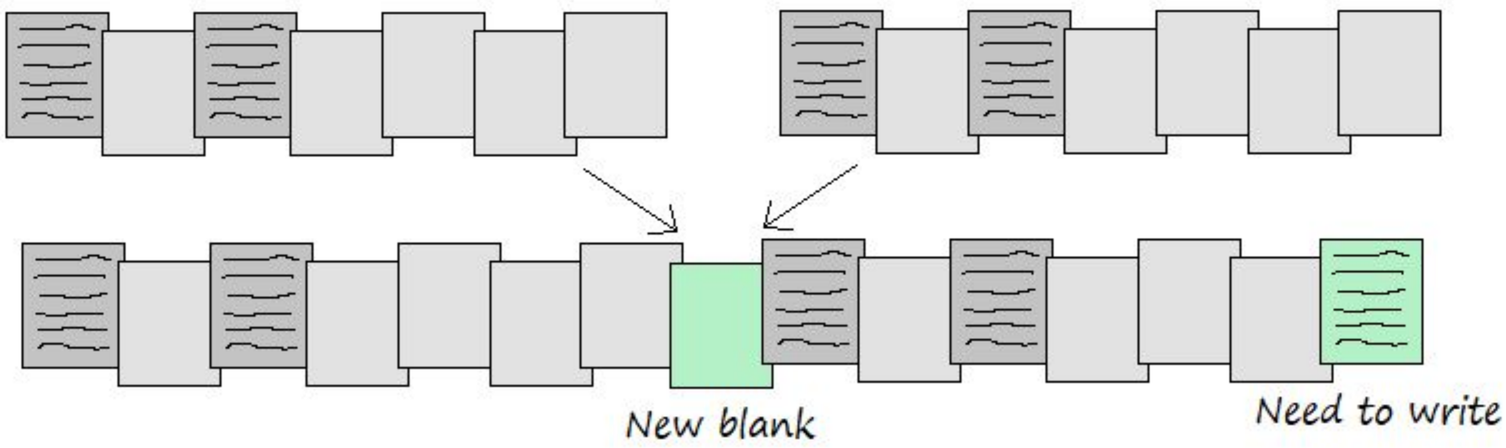
- String parsing
- Just need to be careful
- Don’t parse “and” too early!
  - `And and, and and and, and and.`
- Suggestion: split by comma first, then remove the final “and”.

Problem N: Cursed Printer

- If $k = 0$, possible.
- If $0 < k < x - 1$, impossible.
  - A single print will give at least $x - 1$ blanks.
- If $k \geq x - 1$, possible.

Problem N: Cursed Printer

- Observation: Only one print job needed.



- If necessary, write on some pages, or decrease, to get exact blanks. Always possible.

Problem N: Cursed Printer

- Given print job with $n \geq x$ with at least x non-blanks:
  - There are $\leq (n - 1) + (n - x) = 2n - 1 - x$ non-blanks.
  - We want $2n - 1 - x \geq k$.
  - $n > (k+x)/2$
- Thus, $\mathbf{n = \lfloor (k+x)/2 \rfloor + 1}$
  - Can always write on some blank pages to get exactly k blanks.

Problem C: Best Grill Contest

- For each leaf,
  - Go up the tree, greedily add votes to the grill so it wins.
  - Alternatively, binary search the correct number of votes.

Problem C: Best Grill Contest

- Simulating 2^n times too slow.
- Optimize: Keep final vote and number of "petty" votes for each subtree.
  - Dynamic programming, $O(2^n)$.
- Solving for a leaf only requires going up the tree.
  - O(n) (greedy) or O(n log ans) (binsearch).
- $O(n2^n)$ or $O(n2^n \log ans)$ overall.

Problem C: Best Grill Contest

- Who is the best grill?

1. 夜刀神 十香
2. 七罪
3. 時崎 狂三
4. 四糸乃
5. 五河 琴里
6. 誘宵 美九
7. 七輪指♡
8. 八舞

Problem D: Extraordinary Machine

- WLOG assume connected.
- If $(r_1\ r_2\ \ldots\ r_n)\ (s_1\ s_2\ \ldots\ s_n)$ is valid,
- then $(\alpha r_1\ \alpha r_2\ \ldots\ \alpha r_n)\ (\beta s_1\ \beta s_2\ \ldots\ \beta s_n)$ is also valid.
- We can assume $r_1 = s_1 = 1$.
- Thus, $r_i s_i = r_1 s_1 = 1$.

Problem D: Extraordinary Machine

- $r_i/r_j = s_j/s_i$, so we can unify constraints.
- $s_i = 1/r_i$, so we can just consider r_i's.
- Valid iff every cycle has product 1. (Why?)
- We can just do a single BFS/DFS.

Problem D: Extraordinary Machine

- Problem: Numbers grow too large!
  - Consider a cycle with costs 2, 2, 2, 2, ..., ½, ½, ½, ...
- Floats too inaccurate for checking.
  - Will also get overflow and underflow.

Problem D: Extraordinary Machine

- Fix: Only check modulo large primes p.
- Need to randomize selection of p to avoid getting hacked.
- Choose several p to make it less likely to fail.

Problem F: Biko

- Idea 1:
  - 1/q + 1/q + ... + 1/q. (q - p times)
- Not enough! Can only have up to 120.
  - This only gets us until 120/q.

Problem F: Biko

- Idea 2:
  - $1/q + 1/q + \ldots + 1/q + (1/2q + 1/2q) + (1/2q + 1/2q) + \ldots$
  - until it sums to $(q - p)/q$.
- Still not enough.
  - This only gets us until $180/q$.

Problem F: Biko

- Idea 3:
  - Generalize: once we run out, go to the next one.
    - $1/q + \ldots + (1/2q + 1/2q) + \ldots + (1/3q + 1/3q + 1/3q) + \ldots$
- Goes up to $602/q$, which is enough.
  - Solves up to $q = 603$.

Problem F: Biko

- Follow-up: Can you squeeze more out of this scheme to reach $q < 659$?

Problem B: ABCD Paths

- If has cycle of same letters: INFINITE.
- Otherwise, finite.
- Topological sort per letter, then DP.
- Alternatively, make a graph with 26n nodes, then find longest path in it.

Problem B: ABCD Paths

- For lexicographically smallest, string comparison too slow.
- Key: Strings are nondecreasing, so compress to frequency counts.
  - $O(\alpha)$ comparison (α = alphabet size)
- $O(\alpha^2(n + e))$ overall.
  - or $O(\alpha(\alpha n + e))$

Problem B: ABCD Paths

- Challenge: Can you still solve it if alphabet size is huge, say, $\alpha = O(e)$?

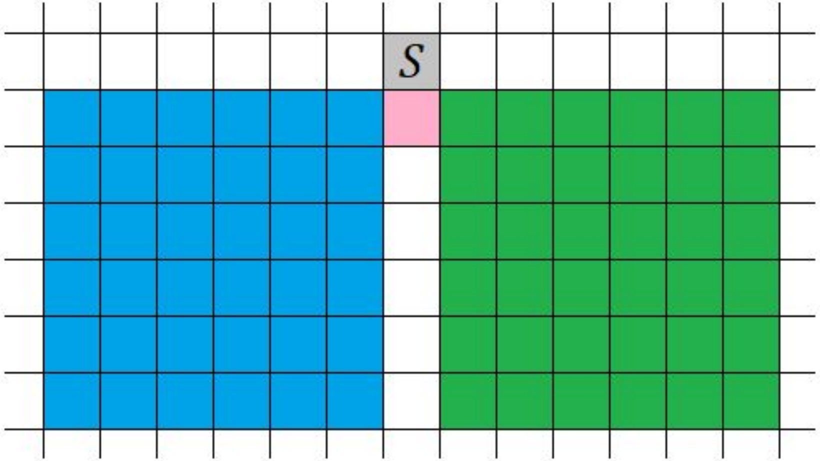
Problem H: Pokémain

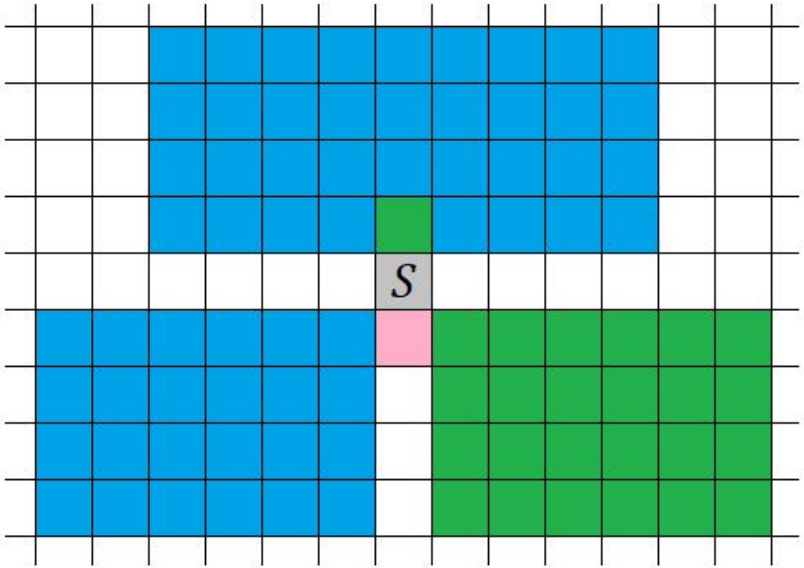
- WLOG assume $0 \leq x \leq y \leq z$.
  - Be careful with output.
- Small cases:
  - If $x = 0$, impossible. (Starting point visitable)
  - If $x = y = z = 1$, possible. (Just the starting point)
  - If $x = y = 1$, $z > 1$, impossible (Extra cell visitable by at least two guys).
  - If $x = y = z = 2$, impossible. (Why?)

Problem H: Pokémain

- Remaining cases (both possible):

  $x = 1, z \geq y \geq 2$ and $y \geq x \geq 2, z \geq 3$



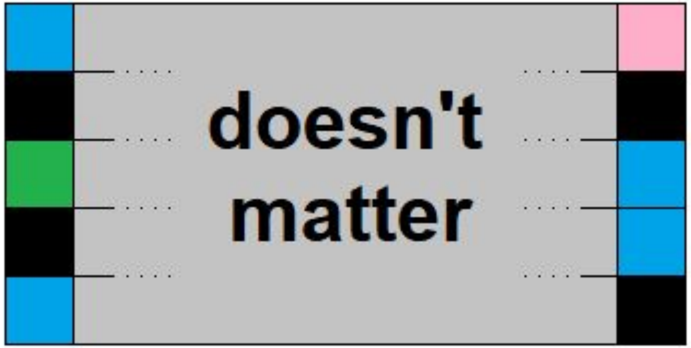


Problem G: Frozen

- Brute force too slow!
- Let’s swap r, c. Easier to draw. Thus, assume $r \leq 10$.
- Need efficient connectivity queries on a huge distance, with some updates.
- **Segment tree** might help...

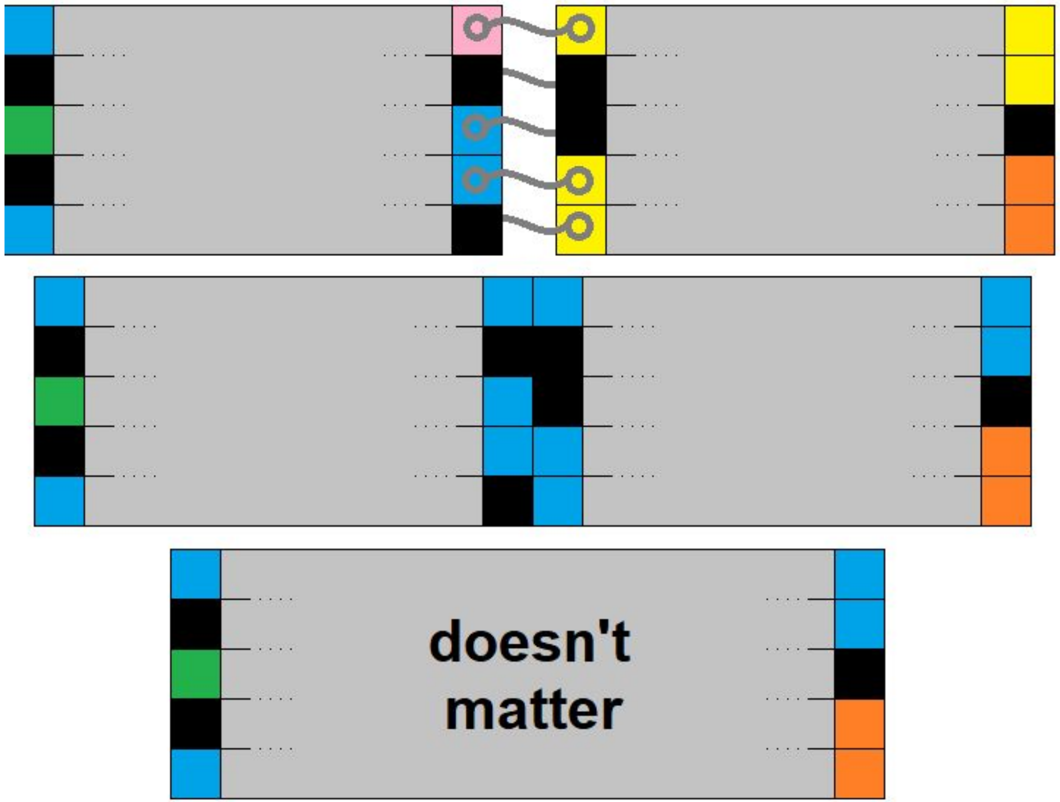
Problem G: Frozen

- Segment tree: for each block, store connected components of left and right columns:



Problem G: Frozen

- On merging two blocks, “stitch” them and do BFS/union find.
    - O(r).

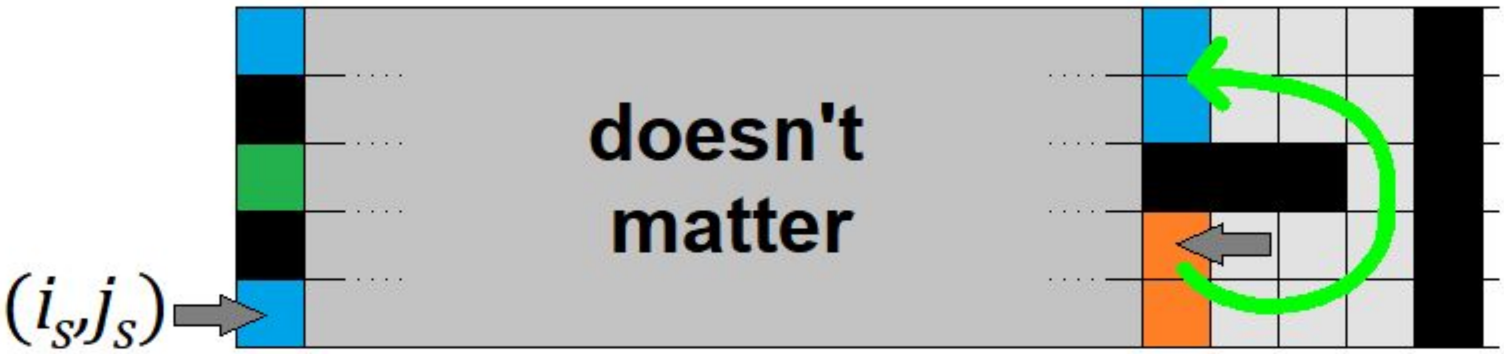


Problem G: Frozen

- On query (i_s, j_s) (i_d, j_d) (assume $i_s \le i_d$), range query on block $[i_s, i_d]$. Check if same component.

Problem G: Frozen

- **Wrong!**



Problem G: Frozen

- Range query on **3 blocks** $[1, i_s - 1]$, $[i_s, i_d]$, $[i_d + 1, c]$, and "stitch them"

- Correct now. O(r log c) per query/update.

Problem L: Ryu Quezacotl's Birthday

- Brute force too slow!
- Observation: For most starting values, the product goes to 0 early.
  - Reason: Digits are close to purely random.
  - Specifically, after < 10 steps. Thus, assume $i_L \leq i_R < 10$.
  - Exception: 1, 11, 111, etc.
    - But there are only a few of these. Can handle individually

Problem L: Ryu Quezacotl's Birthday

- Observation 2: After one step, < 4400 distinct values remain.
- Collect all such distinct values at the beginning.
- For each one,
  - count the starting points in $[n_L, n_R]$ that go to it.
  - Digit DP. $O(\log n_R)$

Problem L: Ryu Quezacotl's Birthday

- Everything can be done quickly now.
- Single test case: 4400*(log n_R + 10) steps.

Problem E: NCIS 2 Experts 1 Keyboard

- Root the tree arbitrarily, say at x.
- Recursively compute effects of paths not containing x.
- Compute effects of paths containing x.
    - Needs several DPs, but can be done.
    - O(size(x) log size(x))
- Overall $O(n^2 \log n)$, too slow.

Problem E: NCIS 2 Experts 1 Keyboard

- Optimize: Don’t choose x arbitrarily. Choose x to be the **centroid**, i.e., every child size is ≤ size(x)/2.
  - Can be found in O(size(x)).
- Now, recursion depth ≤ lg n, so $O(n \log^2 n)$ overall.

Problem M: Unlimited Powah

- $(x - 1)(y - 1)(z - 1) = xyz - 1$ is equivalent to
- $(x + z - 1)(y + z - 1) = z^2 - z + 1$
- Every powah point can be derived from a factorization of $z^2 - z + 1$.
- Factorize all $z^2 - z + 1$ for all $-n \le z \le n$, and collect all possible (x, y, z), precompute answers, etc.
- The only bottleneck is **factorizing all $z^2 - z + 1$**.

Problem M: Unlimited Powah

- Method 1: Sieve powered by number theory.
- If $z^2 - z + 1 = 0 \pmod{p}$, then $(2z - 1)^2 = -3 \pmod{p}$.
- Need to find modular sqrt of -3 modulo p for every $p \le \mathrm{sqrt}(\max(z^2 - z + 1)) = \mathrm{sqrt}(n^2 + n + 1)$.
  - Tonelli-Shanks or Cipolla's algorithm.
- This gives us all z divisible by p.

Problem M: Unlimited Powah

- Normal sieve up to sqrt(n^2 + n + 1) ≤ 10^6 + 1.
- Let f[z] = z^2 - z + 1 for -n ≤ z ≤ n.
- For every p, compute z_1 and z_2 such that
  - $(2z - 1)^2$ = -3 (mod p).
- For every z = z_1 or z_2 (mod p), factor all p out of f[z].
- The remaining f[z] are now prime or 1.
- We now get all prime factors of all z^2 - z + 1.

Problem M: Unlimited Powah

- Method 2: Alternative sieve.
- Doesn't need advanced stuff like Tonelli-Shanks.
- Technique popular in Project Euler. Omitting details.
- Instead, solve PE 216 and read its tutorial doc (written by yours truly) for details.

Problem K: A Song of Stacks and Queues

- Each transform gives an (m-k)-permutation.
  - Actually, queue transform is just the (m-k)-prefix.
- Compute both permutations and match them.
- This gives us m-k constraints.
- Form a directed graph with m-k edges.
- Key: Per node, at most one in-edge and out-edge.
- Thus, each component is either path or cycle.

Problem K: A Song of Stacks and Queues

- Fix scale factor s.
- For every path, first value determines the rest.
- For every cycle, any value determines the rest.
- And the cycle length gives a constraint on the scale factor.

Problem K: A Song of Stacks and Queues

- More precisely, for a cycle length c, we must have scale factor $s^c = 1 \pmod{p}$.
- Equivalent to $s^{\gcd(p - 1, c)} = 1 \pmod{p}$.
  - Can be proven from Fermat’s and Bézout's
- If s doesn’t satisfy this, then the whole cycle must be 0.

Problem K: A Song of Stacks and Queues

- Compute cycle lengths c, reduce to gcd(p - 1, c), and collect possible scale factors that allow this cycle to be nonzero.
- Many more details (DP and combinatorics) but at this point, you have the most important insights. You can figure out the rest. :)

Thank you!

- **Kevin Charles Atienza**
  - Setting, Testing, Judging, Solution Slides, Nice Leaderboard
- **Rene Josiah Quinto**
  - Setting, Testing, Judging
- **Manuel Antonio Rufino**
  - Setting, Judging
- **Marc Patrick Celon**
  - Setting, Judging

- `A`: **Quantum ...** - Atienza+Quinto
- `B`: **ABCD Paths** - Quinto
- `C`: **Best Grill Contest** - Rufino+Atienza
- `D`: **Extraordinary Machine** - Quinto
- `E`: **NCIS 2 Experts 1 Keyboard** - Quinto
- `F`: **Biko** - Celon+Atienza
- `G`: **Frozen** - Atienza
- `H`: **Pokémain** - Atienza
- `I`: **Wax** - Atienza
- `J`: **Does It Spark Joy?** - Atienza
- `K`: **A Song of Stacks and Queues** - Atienza
- `L`: **Ryu Quezacotl's Birthday** - Atienza
- `M`: **Unlimited Powah** - Atienza
- `N`: **Cursed Printer** - Quinto