ICPC Asia Seoul Regional Contest 2026

Solutions Presentation

Gyojun Youn

November 14, 2026



K: Stepping Numbers

Proposer: Seung Yong Lee **Setter:** Young-Min Kang



Problem: Find the n-th smallest positive integer whose adjacent digits differ by 1.

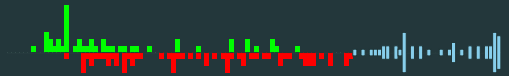
Solution:

- Start a queue with $1, 2, \cdots, 9$.
- Pop v; let $d = (v \bmod 10)$.
- Append $10v + (d - 1)$ if $d > 0$, then $10v + (d + 1)$ if $d < 9$.
- Repeat until find all stepping numbers $< 10^{15}$. There are about $160\,000$.

Complexity: $\mathcal{O}(n)$ time and space.

G: Glowing Devices

Proposer: Jae-Hoon Kim **Setter:** Jae-Hoon Kim



Problem: Pay to force some devices on; let the rest activate for free. Minimize the payment.

Observation: Work backwards. A device left unforced is a unit-time job with deadline $n - t_i$ and saved cost w_i.

Solution:

- Process devices by increasing deadline.
- Add w_i to a min-heap of devices we hope to keep unforced.
- If heap size exceeds the deadline, remove its smallest cost and pay it.

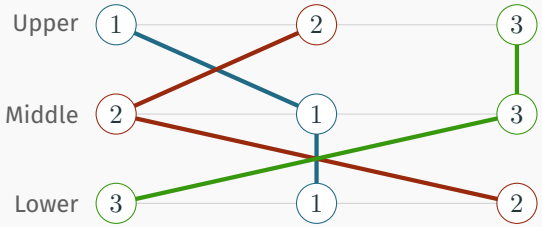
Complexity: $\mathcal{O}(n \log n)$ time, $\mathcal{O}(n)$ space.

D: Circuit X

Proposer: Sang Won Bae **Setter: Sang Won Bae**



Problem: Two permutations label the pins in the upper and lower layers. Connect equal labels through a **freely chosen middle order**.



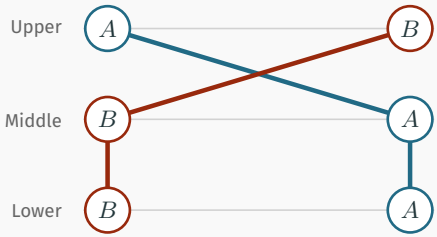
Goal: Minimize $a\chi^{\wedge} + b\chi^{\vee}$, where $\chi^{\wedge}$ and $\chi^{\vee}$ count crossings above and below the middle layer.

D: Circuit X

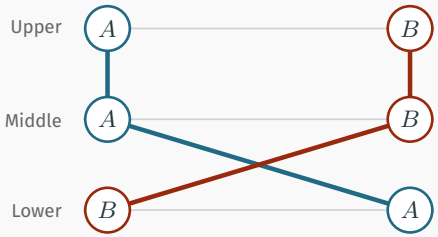
Proposer: Sang Won Bae **Setter: Sang Won Bae**

Observation: Fix two pins A and B. Compare their order in the upper and lower layers.

Opposite orders



Cross **only above**: cost a



Cross **only below**: cost b

Exactly one crossing. Best cost: $\min(a, b)$.

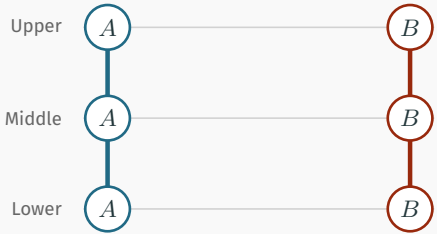
D: Circuit X

Proposer: Sang Won Bae **Setter:** Sang Won Bae

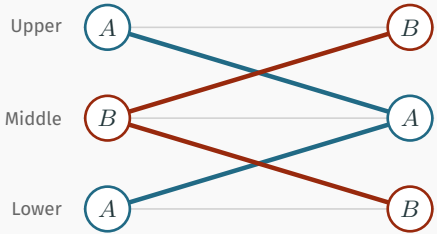


Observation: Fix two pins A and B. Compare their order in the upper and lower layers.

Same order



No crossings: cost 0



Both crossings: cost $a + b$

Zero or two crossings. Best cost: $\min(0, a + b)$.

D: Circuit X

Proposer: Sang Won Bae **Setter: Sang Won Bae**



Solution: For each pair of pins, greedily take its cheaper crossing pattern.

Key fact: All these pairwise choices can be achieved *simultaneously*: the middle order is one of

$$\boxed{\text{Upper}, \quad \text{Lower}, \quad \text{reverse}(\text{Upper}), \quad \text{reverse}(\text{Lower}).}$$

Solution: Let I be the number of pairs with opposite orders in the upper and lower layers. Then

$$\boxed{I \min(a, b) + \left(\binom{n}{2} - I \right) \min(0, a + b)}$$

Complexity: Count I by inversion counting: $\mathcal{O}(n \log n)$ time.

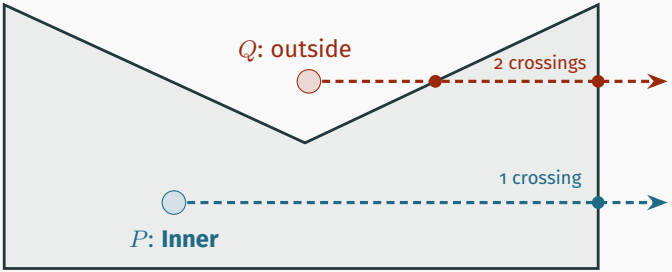
E: Drones

Proposer: Hwan-Gue Cho **Setter: Hwan-Gue Cho**



Problem: Classify each drone as **Inner**, **Boundary**, or **Outer** for a polygon and distance d.

Inner: An edge point is Inner. Otherwise, cast a horizontal ray: **odd** crossings mean inside, **even** mean outside.

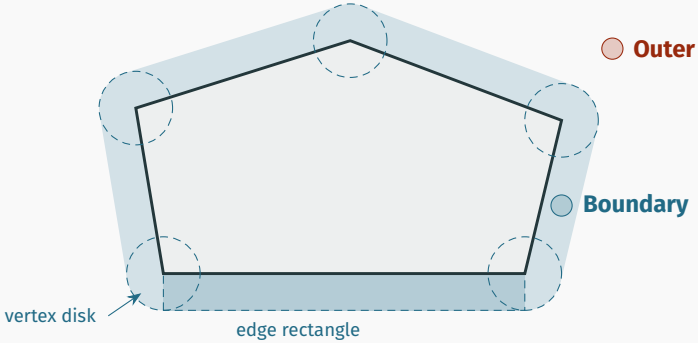


E: Drones

Proposer: Hwan-Gue Cho **Setter: Hwan-Gue Cho**

Boundary: For an outside drone, is its distance to the polygon at most d?

Observation: Union of **radius-d vertex disks** and **width-$2d$ edge rectangles**.



E: Drones

Proposer: Hwan-Gue Cho **Setter:** Hwan-Gue Cho

Solution: Vertex disks are easy: $\|P - A\|^2 \le d^2$.

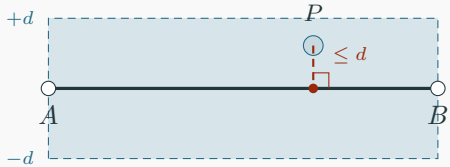
Edge rectangle: For segment AB, let $u = P - A$ and $v = B - A$.

1. The projection lies on segment AB:

$$\boxed{0 \le u \cdot v \le v \cdot v.}$$

2. Distance to the line is at most d:

$$\boxed{(u \times v)^2 \le d^2 (v \cdot v).}$$



Complexity: Check all $s \le 5$ edges per drone: $\mathcal{O}(sn)$ time.

Pitfalls: No square roots; use 64**-bit integers** for cross-product comparisons.

B: Boss Raid Log

Proposer: Seungbum Jo **Setter:** Seungbum Jo



Problem: Does every log entry belong to some occurrence of P, with consecutive selected positions at least d apart?

Solution:

- Scan left to right: L_j is the earliest position for P_j.
- Scan right to left: R_j is the latest position for P_j.
- If either greedy match fails, answer `NO`.

Key fact: Position i can play role j exactly when

$$\boxed{S_i = P_j \quad \text{and} \quad L_j \le i \le R_j.}$$

Use an earliest prefix, position i, and a latest suffix.

B: Boss Raid Log

Proposer: Seungbum Jo **Setter:** Seungbum Jo



Solution: Sweep the log once more.

- Activate interval $[L_j, R_j]$ when $L_j \leq i$.
- For each spell c, maintain the furthest active endpoint $\mathrm{reach}[c]$.
- Entry i is covered iff $\mathrm{reach}[S_i] \geq i$.

Complexity: $\mathcal{O}(n + m + k)$ time and space.

J: Safe

Proposer: Jeong Seop Sim **Setter:** Jeong Seop Sim



Problem: Choose one edit-cost table and split Y into **at least two nonempty pieces**. Minimize the largest edit cost from X to a piece.

Solution: Binary search the answer T. For each cost table, ask:

- Can we split Y so **every piece** has edit cost at most T?
- Scan Y once, sharing the usual edit-distance DP across all possible starts of the current piece.

Upper bound: $0 \le T \le 9|X|$. Split Y into pieces of length at most $|X|$.

J: Safe

Proposer: Jeong Seop Sim **Setter: Jeong Seop Sim**

DP state: After reading $Y[1..j]$, i is the number of characters of X already processed in the **current piece**. Keep its minimum edit cost in two rows:

- $f[i]$: we are still working on the **first piece**.
- $g[i]$: **at least one earlier piece** is complete (and costs at most T).

Edit step: For the next digit y, update either row $h \in \{f, g\}$:

$$h'[i] = \min \begin{cases} h[i] + C_{0,y} & \text{insert } y, \\ h'[i-1] + C_{x_i,0} & \text{delete } x_i, \\ h[i-1] + C_{x_i,y} & \text{match/replace.} \end{cases}$$

Start: Before the first digit, $f[i] = \sum_{r=1}^{i} C_{x_r,0}$; all $g[i] = \infty$.

J: Safe

Proposer: Jeong Seop Sim **Setter:** Jeong Seop Sim



Cut: After reading a digit of Y, a piece may end if $f[|X|] \le T$ or $g[|X|] \le T$.

Restart: When the **next digit** arrives, start a new piece in g from the deletion-only row $\delta[i] = \sum_{r=1}^{i} C_{x_r,0}$. Merge it with the ongoing g states by taking minima.

Pitfalls:

- **One piece is not allowed:** accept only if the final $g[|X|] \le T$, not $f[|X|]$.
- **Empty pieces are not allowed:** cut only after consuming a digit, and restart only when another digit arrives.

Complexity: $\mathcal{O}(k|X||Y|\log|X|)$ time; $\mathcal{O}(|X|)$ DP space.

M: Workbench

Proposer: Seung Yong Lee **Setter: Seung Yong Lee**



Problem: Partition the steps into batches. Respect the space for intermediate parts; minimize total batch cost.

DP: Let $F_i = \sum_{t=1}^{i} f_t$ and D_i be the best cost for the first i steps.

$$D_i = \min_{j \text{ valid}} \left[D_j + \max\big(b(F_i - F_j),\ a(s_j + s_i)\big) \right].$$

Split the maximum: Set $u_j = bF_j + as_j$ and $v_i = bF_i - as_i$.

$$D_i = \min \left\{ bF_i + \min_{u_j < v_i} (D_j - bF_j), \quad as_i + \min_{u_j \geq v_i} (D_j + as_j) \right\}.$$

Next step: Two range minima over the active values of u_j.

M: Workbench

Proposer: Seung Yong Lee **Setter: Seung Yong Lee**



Solution:

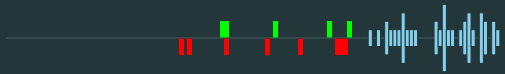
$$D_i = \min\left\{ bF_i + \min_{u_j < v_i}(D_j - bF_j),\quad as_i + \min_{u_j \geq v_i}(D_j + as_j) \right\}.$$

- Coordinate-compress all u_*.
- Store both $D_j - bF_j$ and $D_j + as_j$ in a segment tree.
- Insert $j = i - 1$. Remove old j while $\sum_{t=j+1}^{i-1} s_t > m$. Since valid j form a contiguous interval, maintain only this interval in the segment tree.
- Query the prefix $u_j < v_i$ and the suffix $u_j \geq v_i$.

Complexity: $\mathcal{O}(n \log n)$ time, $\mathcal{O}(n)$ space.

I: Potion Shop

Proposer: Junseok Choi Setter: Junseok Choi



Problem: Remove any positive amount from one pile, or fewer than p crystals in total. Maintain the winner after updates.

Solution: Write $a_i = pq_i + r_i$, where $0 \le r_i < p$.

$$X = \bigoplus_i q_i, \qquad R = \sum_i r_i \pmod{p}.$$

$$\boxed{\texttt{SECOND} \iff X = 0 \text{ and } R = 0.}$$

Updates: Remove the old quotient and remainder; insert the new ones.

Complexity: $\mathcal{O}(n + q)$ time, $\mathcal{O}(n)$ space.

I: Potion Shop

Proposer: Junseok Choi Setter: Junseok Choi



Why losing? From $(X, R) = (0, 0)$:

- A one-pile move changes its quotient, or changes R.
- A lesser potion changes the total modulo p by a nonzero amount.

Why winning? From any other position:

- If $X \neq 0$, use the usual Nim move to lower one quotient and make $X = 0$. Choose its new remainder to make $R = 0$ too.
- Otherwise, remove R crystals from the remainder parts. No quotient changes, and R becomes 0.

A: Baduk Solitaire

Proposer: Chan-Su Shin **Setter:** Chan-Su Shin



Problem: Repeatedly remove at least two consecutive stones of one color, ignoring empty cells. Can we remove them all?

Observations:

- Ignore empty cells and compress equal-colored stones into **runs**.
- Partial removal can wait. Remove a whole run when it disappears.
- Join the two ends: the line and the circle have the same answer.

Why a circle? A move crossing the cut can be postponed: clear the middle first, then remove the saved same-colored end stones together.

Easy case: One circular run succeeds iff it contains at least two stones.

A: Baduk Solitaire

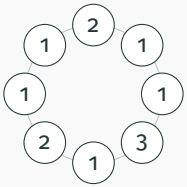
Proposer: Chan-Su Shin **Setter:** Chan-Su Shin



Observation: Otherwise there are $2h$ alternating runs. Call a run of size 1 a singleton.

Criterion:

$$\texttt{YES} \iff \text{the longest circular block of singletons has length} < h.$$



Run sizes: $h = 4$, longest block $= 2$.

Why?

- A block of h singletons remains an obstruction after any legal removal.
- Otherwise, remove a large run beside a longest block if needed. The same condition survives with $h - 1$.

Complexity: $\mathcal{O}(n)$ time and space.

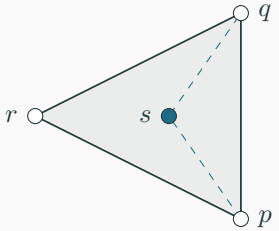
L: Summoning Triangle

Proposer: Sang Duk Yoon **Setter:** Sang Duk Yoon



Problem: Count triangles containing exactly one stone of every color, including their three vertices.

Geometry: Fix a directed side pq. Rotate the picture so $p = (0, 0)$ and q lies above it. Consider only points to the left.



A point s is inside pqr iff

$$\text{slope}(ps) < \text{slope}(pr), \qquad \text{slope}(qs) > \text{slope}(qr).$$

Sweep r in increasing p-slope. The first inequality is now automatic. Only a threshold in the q-order remains.

L: Summoning Triangle

Proposer: Sang Duk Yoon **Setter: Sang Duk Yoon**



Ranks: Give each left-side point a rank t increasing with its q-slope. Larger ranks lie on the inside side of the threshold.

Per color: Maintain the largest and second-largest processed ranks, $A[c]$ and $B[c]$.

- Missing colors start at $A[c] = B[c] = 0$.
- Give the two endpoint colors a sentinel maximum $n - 1$.
- Skip pq if its endpoint colors agree.

At r: Insert its rank $t = t(r)$. The triangle is valid exactly when

$$\boxed{\min_c A[c] = t \quad \text{and} \quad \max_c B[c] < t.}$$

r supplies its own color. Every other color has exactly one rank above t.

L: Summoning Triangle

Proposer: Sang Duk Yoon **Setter:** Sang Duk Yoon



Make it linear per pq:

- Keep a histogram of the current maxima $A[c]$.
- Each $A[c]$ only increases, so a pointer to the smallest occupied rank only moves right.
- If the old maximum is u and the new rank is t, update $M \leftarrow \max(M, \min(u, t))$, where $M = \max_c B[c]$.

Preprocessing: Sort directions around every point once. Extract both half-plane orders in $\mathcal{O}(n)$ per side.

Complexity: $\mathcal{O}(n^3)$ time, $\mathcal{O}(n^2)$ space.

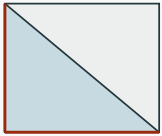
Aside: $\mathcal{O}(n^3 \log k)$ needs an efficient implementation. Random hashing also gives an $\mathcal{O}(n^3)$ approach.

F: Erin Needs Cake

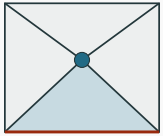
Proposer: Gyojun Youn **Setter: Gyojun Youn**

Problem: **Interactive.** Locate a hidden triangular face with at most $\lceil \log_2 n \rceil$ simple-loop queries.

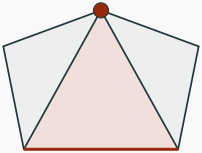
Observation: Peel one boundary triangle while keeping the remaining region a disk.



Two exposed edges: remove an ear.



One exposed edge: opposite vertex must be interior.



Boundary opposite vertex: do not remove it yet.

A removable triangle always exists. Stop after recording the final triangle.

F: Erin Needs Cake

Proposer: Gyojun Youn **Setter: Gyojun Youn**



Solution: Record the recursive removal order $r_1, r_2, \ldots, r_n$.

$\{r_j, r_{j+1}, \ldots, r_n\}$ always has one simple boundary.

Implementation:
- Build face adjacency from shared roads; mark the initial boundary.
- Keep a queue of candidates. Recheck eligibility when popped.
- A removal exposes at most three neighboring faces. Only these need to be queued again.

Important: A newly exposed vertex is now a boundary vertex. Earlier queue entries may no longer be removable.

Bound: The queue work is linear.

F: Erin Needs Cake

Proposer: Gyojun Youn **Setter: Gyojun Youn**



Binary search: Let the unknown face be r_h. Maintain an interval of possible values of h.

- Query the **whole suffix** $r_j, r_{j+1}, \ldots, r_n$ at the midpoint.
- Reply 1 means $h \geq j$; reply 0 means $h < j$.

Build a query: A road is on the suffix boundary iff exactly one adjacent face belongs to the suffix. Follow the resulting cycle.

Pitfalls: The removed prefix need not be simple. Do not query an arbitrary interval of the permutation.

Complexity: $\mathcal{O}(n \log n)$ time, $\mathcal{O}(n)$ space; at most $\lceil \log_2 n \rceil$ queries.

C: Chemistry Experiment

Proposer: Jeounghui Nah **Setter: Jeounghui Nah**



Problem: Count safe length-n strings over $0, 1, \ldots, k$, using every nonzero symbol. Adjacent nonzero symbols must agree.

Solution: First allow only j chemicals, without requiring all of them. Let the count be $H_n(j)$.

Inclusion–exclusion:

$$\text{Answer} = \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} H_n(j) \pmod{m}.$$

Next question: How do we compute one $H_n(j)$ without iterating up to n?

C: Chemistry Experiment

Proposer: Jeounghui Nah **Setter: Jeounghui Nah**



Two states: Let z_t and w_t count safe prefixes ending in zero and nonzero, respectively.

$$z_{t+1} = z_t + w_t, \qquad w_{t+1} = j z_t + w_t. \qquad \begin{pmatrix} z_n \\ w_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ j & 1 \end{pmatrix}^n \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Solution: $H_n(j) = z_n + w_n$. Use binary exponentiation for each j.

Complexity: $\mathcal{O}(k \log n)$ time, $\mathcal{O}(k)$ space.

H: Last Jump

Proposer: Sung-Hwan Kim **Setter: Sung-Hwan Kim**

Problem: Extend X one character at a time, sometimes jumping to its next occurrence. At the end, the last jumper wins.

Key invariant: The current position i is the **earliest ending position** of X in T.

- An occurrence of Xc cannot end at or before i.
- If $T[i+1] = c$, that earliest end is $i+1$; otherwise we jump to it.

Compress: A suffix automaton groups substrings with the same ending positions. They have the same moves and jump types.

Graph: For transition $v \xrightarrow{c} u$, it is a jump iff $\mathrm{first}[u] > \mathrm{first}[v] + 1$.

H: Last Jump

Proposer: Sung-Hwan Kim **Setter:** Sung-Hwan Kim

DP: $W[v][b]$: can the current player win? $b = 1$ means that player wins if no further jump happens.

Terminal: If $\mathrm{first}[v] = n$, then $W[v][b] = b$.

Transition: Move to u, then give the turn to the opponent.

Move	Opponent's flag	Current player wins if
No jump	$1 - b$	$\neg W[u][1 - b]$
Jump	0	$\neg W[u][0]$

Solution: Take OR over legal transitions, in decreasing SAM length. Start at $W[\mathrm{root}][1]$.

Complexity: $\mathcal{O}(|\Sigma|n)$ per string.

Thanks to:

Problem proposers & setters:

- Jae-Hoon Kim
- Chan-Su Shin
- Hwan-Gue Cho
- Young-Min Kang
- Sung-Hwan Kim
- Soohwan Kim
- Sang Won Bae
- Jeong Seop Sim
- Gyun Woo
- Sang Duk Yoon
- Seungbum Jo
- Sung-Woo Hur
- Yeonghyeon Kim
- Jeounghui Nah
- Gyojun Youn
- Seung Yong Lee
- Junseok Choi

And everyone behind the contest:

- Seokju Hong
- Suhyun Park
- Seungwon Shin
- Jaehong Kim
- Volunteers

We hope you enjoyed the problems. See you next year!