NWERC 2024

Solutions presentation

November 24, 2024

The NWERC 2024 Jury

- **Andreas Grigorjew**
  University of Helsinki
- **Arnar Bjarni Arnarson**
  Reykjavík University
- **Arne Alex**
  Ludwig-Maximilians-Universität München
- **Atli Fannar Franklín**
  University of Iceland
- **Christopher Weyand**
  Karlsruhe Institute of Technology / moia.io
- **Doan-Dai Nguyen**
  École normale supérieure - Université Paris Sciences & Lettres
- **Haidar Ismaeel**
  freiheit.com / Tishreen University
- **Khaled Ismaeel**
  Innopolis University
- **Maarten Sijm**
  CHipCie (Delft University of Technology)
- **Michael Zündorf**
  Karlsruhe Institute of Technology
- **Nils Gustafsson**
  KTH Royal Institute of Technology
- **Paul Wild**
  FAU Erlangen-Nürnberg
- **Ragnar Groot Koerkamp**
  ETH Zurich
- **Reinier Schmiermann**
  Utrecht University
- **Takuki Kurokawa**
  Technical University of Munich
- **Thomas Beuman**
  Leiden University
- **Vitaly Aksenov**
  City, University of London
- **Wendy Yi**
  Karlsruhe Institute of Technology

Big thanks to our proofreaders and test solvers

- **David Stangl**
  Hasso Plattner Institute / moia.io
- **Joakim Blikstad**
  KTH Royal Institute of Technology
- **Jorke de Vlas**
  Linköping University
- **Oleksandr Kulkov**
  ETH Zürich
- **Pavel Kunyavskiy**
  JetBrains, Amsterdam
- **Wietze Koops**
  Lund University / University of Copenhagen

A: Alphabetical Aristocrats

Problem author: Maarten Sijm

Problem

Given a list of $n \leq 1000$ allegedly Dutch surnames, sort them according to the part starting from the first capital letter.

Solution

- Use a sort function from the standard library of your favourite programming language.
- Specify a custom comparator or sorting key.
- Default string comparisons are already lexicographic and compare using ASCII values.

Run time: $\mathcal{O}(n \log n)$.

Statistics: 99 submissions, 83 accepted, 1 unknown

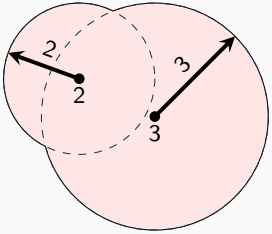
J: Jib Job

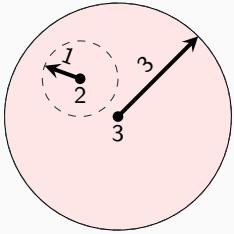
Problem author: Paul Wild, Ragnar Groot Koerkamp

Problem

Given n crane towers with different positions and heights. Place a *jib* on top of each crane such that:

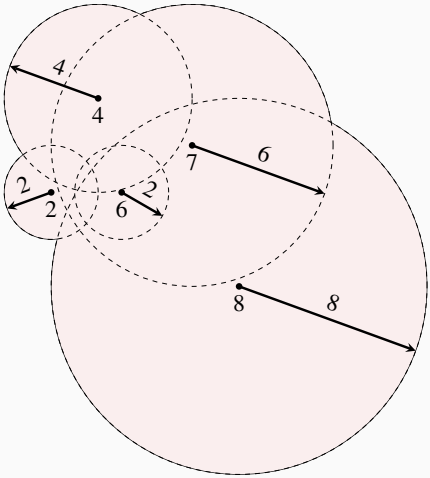
- No jib can collide with any other crane tower.
- No crane is unstable by having a jib longer than the tower's height.
- The ground area that can be reached is maximized.





J: Jib Job

Problem author: Paul Wild, Ragnar Groot Koerkamp



Gotcha

- The solution is *not* unique
- The jib of the crane with height 6 could be larger
- . . . but that does not increase the covered area
- We tried to confuse you :P

J: Jib Job

Problem author: Paul Wild, Ragnar Groot Koerkamp

Observation

- The size of a jib does not influence any other jib since only jibs and cranes can collide.
- Larger jibs cannot cover less area.

  $\Longrightarrow$ Make each jib as large as possible.

Solution

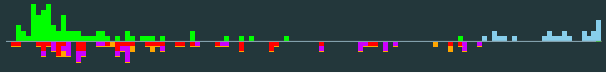
For each tower, determine the closest tower that is larger. Take the minimum of that distance and the height of the tower.

Runtime: $\mathcal{O}(n^2)$

Statistics: 88 submissions, 78 accepted, 3 unknown

E: Evolving Etymology

Problem author: Nils Gustafsson



Problem

Given a word s, take every second letter of $s + s$, starting with the first. Do this $k \leq 10^{18}$ times.

Observation

A solution that simply applies the method k or $\sqrt{k}$ times, is too slow. We need something $\mathcal{O}(n \log k)$.

Solution

- After each time of applying the method, the resulting indices multiply by 2.
- Thus, the ith character in the resulting word is the $(i \cdot 2^k \bmod n)$th character in the original word (using 0-based indexing).

Run time is $\mathcal{O}(n + \log k)$ when pre-calculating $2^k \bmod n$, but $\mathcal{O}(n \log k)$ is also accepted.

Statistics: 168 submissions, 70 accepted, 23 unknown

L: Limited Library

Problem author: Vitaly Aksenov

Problem

Fit m books with height b_i into n shelves of height a_j.

- Constraint: a book of height b_i only fits into a shelf of height $a_j \geq b_i$.
- Choice: shelves have capacity for either x or $y < x$ books.

Find the maximum number of shelves with capacity y so that all boos fit, or print `impossible` if the items do not fit even when all shelves have capacity x.

Observations

- When a book fits into a shelf of height a_i, it fits into all shelves of height $a_j \geq a_i$.
- When choosing shelves to narrow down to capacity y, we do not lose anything by choosing the shelves with the smallest height.
- If we can fit all books when k shelves have capacity y, we can always fit all books when fewer than k shelves have capacity y.

L: Limited Library

Problem author: Vitaly Aksenov

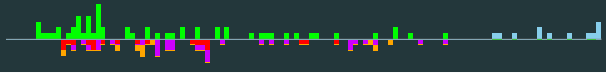
Solution

- Sort shelves by height.
- Sort books by height.
- For a given number k of shelves with capacity y, greedily match books to shelves to decide if they fit.
- Binary-search for the maximum number of shelves.

Statistics: 142 submissions, 67 accepted, 17 unknown

D: Dutch Democracy

Problem author: Thomas Beuman



Problem

Given n parties and their number of seats, how many subsets are there such that

- the cumulative number of seats of the subset is larger than half the total number of seats, and
- dropping any party lowers the cumulative number of seats to at most half the total number of seats?

Solution

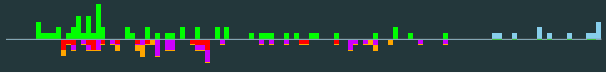
Categorise valid subsets by their party with the lowest number of seats. Without it, the cumulative number of seats drops below the threshold. By transitivity, this is true for all parties in the subset.

- Sort parties by their number of seats.
- Dynamic programming on the number of subsets that include party k and any party with more seats, and that has a cumulative number of seats of at least m.
- Runs in $\mathcal{O}(\text{number of parties} \times \text{total number of seats})$.

Statistics: 131 submissions, 59 accepted, 12 unknown

D: Dutch Democracy

Problem author: Thomas Beuman



Fun fact

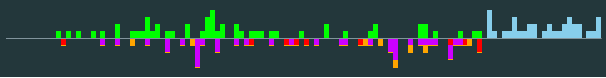
The current Dutch coalition has a majority, but does not meet the minimality criterion.

Fun fact 2

The German coalition met both criteria until the 6th of November, 2024. They dropped the smallest party.

F: Flowing Fountain

Problem author: Michael Zündorf



Problem

Simulate how a multi-level fountain fills up as liquid is poured into levels multiple times.

Naive solution

Just do it, iterating to find next level each time while some poured amount remains. Run time: $\mathcal{O}(qn)$, too slow!

Solution

We need to find the Next Greater Element (NGE) for each index. Can be done with one linear pass and a stack. Use Union-Find with path compression to track where overflow ends up. When level i fills up, unite i with $\mathrm{NGE}(i)$. Run time: $\mathcal{O}(n + q\alpha(n))$.

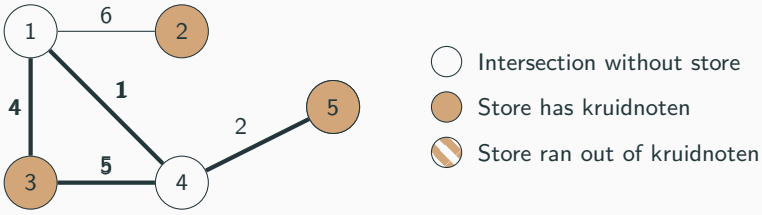
Statistics: 142 submissions, 54 accepted, 35 unknown

K: Kruidnoten

Problem author: Andreas Grigorjew

Problem

Given a graph and the probability $p(v)$ for each store v to have kruidnoten, output the expected length of a shortest path from vertex 1 to n if we get kruidnoten.

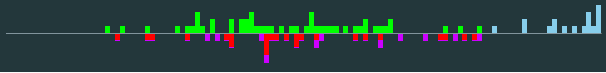


Idea

- We get kruidnoten at a store if it has kruidnoten and no better store has kruidnoten.
- A store is better if there is a shorter path that visits the store.

K: Kruidnoten

Problem author: Andreas Grigorjew



Solution

- For each store v, compute the length $d(v)$ of a shortest path if we visit the store.
- Sort all stores by $d(v)$ in ascending order, assume $d(v_1) \leq d(v_2) \leq \ldots \leq d(v_n)$.
- The probability that v_k has kruidnoten and no better store has them is

$$p_{\text{best}}(v_k) = p(v_k) \cdot \prod_{i=1}^{k-1} (1 - p(v_i)) \ .$$

- The expected length of a shortest path if we get kruidnoten is

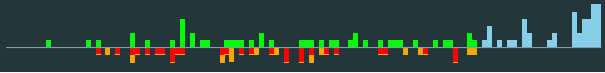
$$\sum_{k=1}^{n} p_{\text{best}}(v_k) \cdot d(v_k) \ .$$

Running time: $\mathcal{O}(m \log(n))$

Statistics: 102 submissions, 50 accepted, 17 unknown

H: Hash Collision

Problem author: Reinier Schmiermann



Problem

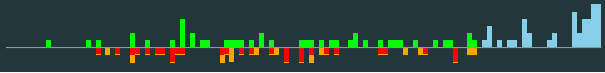
For some hidden function $f : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$, use at most 1000 queries retrieving $f^c(r) = f(\ldots f(r) \ldots)$, to find c and r such that $f^c(r) = c$.

Observations

- When repeatedly applying f, you eventually end up in a cycle.
- When we know the length m of this cycle, and c is part of the cycle, we can effectively invert f, e.g. $f(x) = c$ for $x = f^{-1}(c) = f^{m-1}(c) = f^{2m-1}(c) = \ldots$.
- Likewise: $f^c(r) = c$ for $r = f^y(c)$ where y is any integer $y \equiv -c \mod m$.

H: Hash Collision

Problem author: Reinier Schmiermann



Finding the cycle length

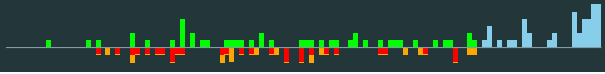
- Starting from an arbitrary value x, compute $c = f^n(x)$ to make sure c is in the cycle.
- Get $f^k(c)$ for various values of k.
- When you get the same value for k_1 and k_2, the cycle length is (a divisor of) $m = |k_1 - k_2|$.
- Compute $i = -c \mod m$ and get $r = f^i(c)$.
- We now have $f^c(r) = c$.

Suitable values of k

- Let $s = \lceil \sqrt{n} \rceil$. Use $k \in \{1, 2, 3, \ldots, s-1, s, 2s, 3s, \ldots, s^2\}$.
- For any m, there are guaranteed to be k_1, k_2 such that $m = k_1 - k_2$.
- This requires about $2s$ queries, which is comfortably less than 1000.

H: Hash Collision

Problem author: Reinier Schmiermann



Solution 2

- Take an arbitrary value x.
- Get $c = f^n(x)$.
- Get $r = f^{n-c}(x)$ (mind the case $n - c = 0$).
- We now have $f^c(r) = f^c(f^{n-c}(x)) = f^n(x) = c$.
- Done!

Statistics: 128 submissions, 44 accepted, 43 unknown

C: Connect Five

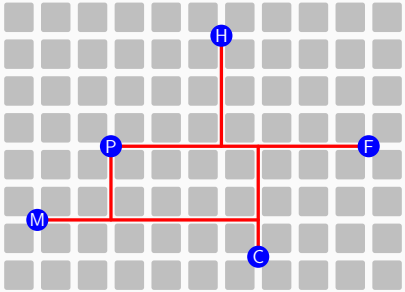
Problem author: Thomas Beuman

Problem

Given are 5 locations in a city grid with square blocks. Refurbish a set of streets in the grid such that

- for any two locations there exists a shortest path between them that only uses refurbished streets;
- the total length of the refurbished streets is minimized.

Output the minimal total length.

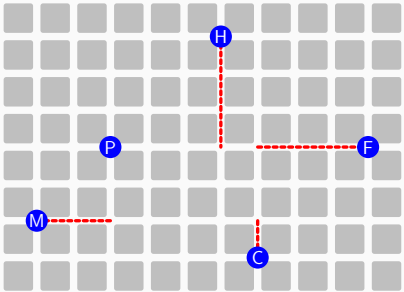


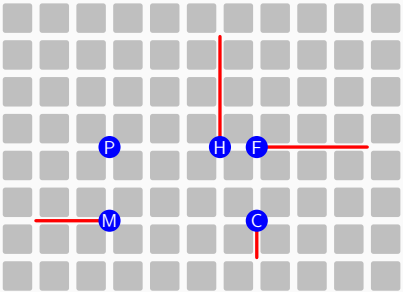
C: Connect Five

Problem author: Thomas Beuman

Insight

If there is a unique leftmost point, we may refurbish a street towards the right, then relocate that point along that street. The same reasoning applies on the right, top or bottom.



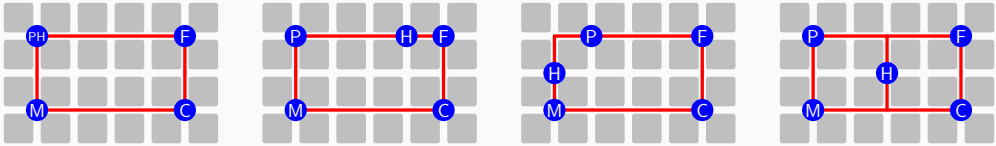


C: Connect Five

Problem author: Thomas Beuman

Solution

- Repeat this action until this is no longer possible. Remove duplicates after each step.
- This leaves either a single point or one of the scenarios below (up to symmetry).
- The remaining solution is now a rectangle, maybe with an extra chord.



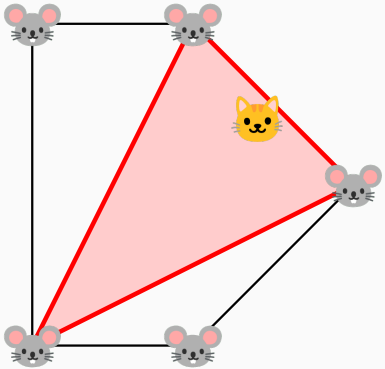
Statistics: 105 submissions, 14 accepted, 72 unknown

M: Mouse Trap

Problem author: Khaled Ismaeel

Problem

You are given a convex polygon. A point in that polygon is chosen uniformly at random. Find the expected number of triangles that can be formed by taking three vertices of the polygon which contain the chosen point.



M: Mouse Trap

Problem author: Khaled Ismaeel

Solution

- By linearity of expectation the answer is equal to the sum of the areas of all triangles, divided by the area of the polygon.
- The latter can be found in linear time. For the former we compute:

$$\begin{aligned}
&\textstyle\sum_{i<j<k} \frac{1}{2}(p_j - p_i) \times (p_k - p_j) \\
&= \textstyle\frac{1}{2} \sum_j (\sum_{i<j} (p_j - p_i)) \times (\sum_{j<k} (p_k - p_j)) \\
&= \textstyle\frac{1}{2} \sum_j ((j-1)p_j - \sum_{i<j} p_i) \times (\sum_{j<k} p_k - (n-1-j)p_j)
\end{aligned}$$

- The inner sums can be precomputed in linear time, so the overall running time is linear as well.

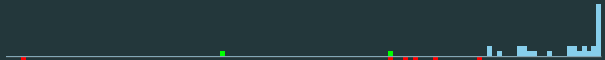
Possible Pitfall

All summations can be done in integers, but the sums may overflow 64-bit integers.

Statistics: 102 submissions, 12 accepted, 78 unknown

B: Binary Search

Problem author: Nils Gustafsson



Problem

Given is an undirected graph G of n vertices with binary vertex labels.

- Find the length of a shortest binary sequence that cannot be obtained by walking in the graph.
- If such binary sequence does not exist, print "`infinity`".

When is the answer infinite?

- If G is a forest, the answer is finite.
- Assume G is a cycle. If the vertex labels follow the pattern "`0011001100...`", the answer is infinite.

When is the answer infinite?

- If G is a forest, the answer is finite.
- Assume G is a cycle. If the vertex labels follow the pattern "0011001100...", the answer is infinite.

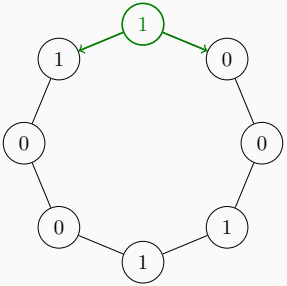
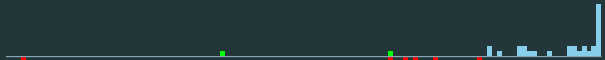


Figure 1: Walking any binary sequence: Both "0" and "1" can be appended to the sequence.

B: Binary Search

Problem author: Nils Gustafsson

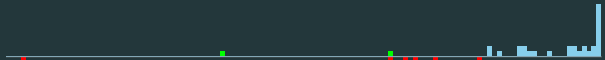


When is the answer infinite?

- If G is a forest, the answer is finite.
- Assume G is a cycle. If the vertex labels follow the pattern "`0011001100...`", the answer is infinite.
- Similarly, if G contains a closed walk whose labels follow this pattern, the answer is infinite.
- Let G contain no closed walk with this pattern. Then there exists a long enough binary sequence that follows this pattern and is not walkable. The answer is thus finite.

B: Binary Search

Problem author: Nils Gustafsson



Solution

- Check if the answer is infinite.
- If not, consider the longest walks of the following four patterns: "0011001100...", "'0110011001...", "1100110011..." and "1001100110...".
- Let the shortest of these walks have length k. **Then the answer is** $k + 1$.
- Proof sketch: the goal is to prove that any pattern of length $\leq k$ can be walked. Any pattern can be "folded" into the form "...00110011 ...", and overlayed on one of the longest walks above.

Use DP to find for each vertex the longest walk of the four patterns above. Runtime: $\mathcal{O}(n)$.

Statistics: 38 submissions, 2 accepted, 30 unknown

I: It's a Kind of Magic

Problem author: Michael Zündorf

Problem

Count the number of multiplicative magic 3×3 squares with product at most n.

Observations

- Given a multiplicative magic square, for every prime p, counting factors p in the entries gives an additive magic square (possibly with duplicates).
- Some linear algebra shows that all additive magic squares have a sum divisible by 3, and take the following form (where the sum is $3a \geq 0$ and $|b| + |c| < a$):

$a+b$	$a-b-c$	$a+c$
$a-b+c$	a	$a+b-c$
$a-c$	$a+b+c$	$a-b$

- The number of additive magic squares (with duplicates) with sum $3a$ equals $2a^2 + 2a + 1$.
- The number of multiplicative magic squares (with duplicates) with product m^3 with $m = p_1^{e_1} \cdots p_k^{e_k}$ equals $(2e_1^2 + 2e_1 + 1) \cdots (2e_k^2 + 2e_k + 1)$.

I: It's a Kind of Magic

Problem author: Michael Zündorf

Idea

- For every $1 \leq m \leq 10^6$, use inclusion-exclusion to compute the number of solutions with distinct numbers.
- There are 5 (up to symmetry) different possible patterns of equal numbers. (Counts include squares with additional duplicates.)

A	B	C
D	E	F
G	H	I

$c_1(m) = \prod(2e_i^2 + 2e_i + 1)$

A	B	A
C	C	C
D	E	D

$c_2(m) = \prod \left(2 \left\lfloor \frac{e_i}{3} \right\rfloor + 1\right)$

A	B	C
C	A	B
B	C	A

$c_3(m) = \prod \left(2 \left\lfloor \frac{e_i}{2} \right\rfloor + 1\right)$

A	A	B
C	D	E
F	G	G

$c_4(m) = \prod(2e_i + 1)$

A	A	A
A	A	A
A	A	A

$c_5(m) = 1$

I: It's a Kind of Magic

Problem author: Michael Zündorf

Solution

- All values $c_i(m)$ for $1 \leq m \leq 10^6$ can be computed with a prime sieve in $\mathcal{O}(N \log \log N)$ time, where $N = 10^6$.
- Use inclusion-exclusion to compute the number of solutions with distinct numbers.
  - The count equals $c_1(m) - 4c_2(m) - 2c_3(m) - 2c_4(m) + 7c_5(m)$.
- For each query, print the sum of all counts with $m \leq \sqrt[3]{n}$.
  - All sums can be precomputed in $\mathcal{O}(N)$ additional time.

Alternative solution

- There are only around 300 different possible multisets of exponents for numbers limited by 10^6.
- For each multiset, locally precompute the number of multiplicative magic squares using a brute force, and hard-code the results into a lookup table.
- Use a sieve to factor all numbers $1 \leq m \leq 10^6$ and look up all counts in this table.

Statistics: 12 submissions, 1 accepted, 9 unknown

G: Glued Grid

Problem author: Paul Wild

Problem

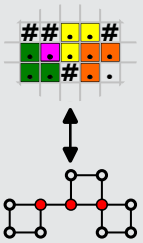
Given an $h \times w$ grid of labelled tiles with one position left empty, arrange the tiles in ascending order of their labels.

- Some tiles glued in place.
- Moving tiles form a simply connected region.
- Only asks for yes/no, not a sequence of moves.

Graph-theoretical Approach

Model moving tiles as a connected undirected graph. Cannot move tiles across articulation points/cut vertices!

- Decompose graph into biconnected components.
- Solve problem on each component individually.
- Pay attention to attribution of articulation points to components.



G: Glued Grid

Problem author: Paul Wild

Solvability of a Sliding Puzzle

Can determine solvability without constructing explicit solution.

- Can solve if and only if the parity of the permutation of the tile labels is even.
- Check that label permutations are contained within each component.
- Compute parity in $\mathcal{O}(n \log n)$ time.

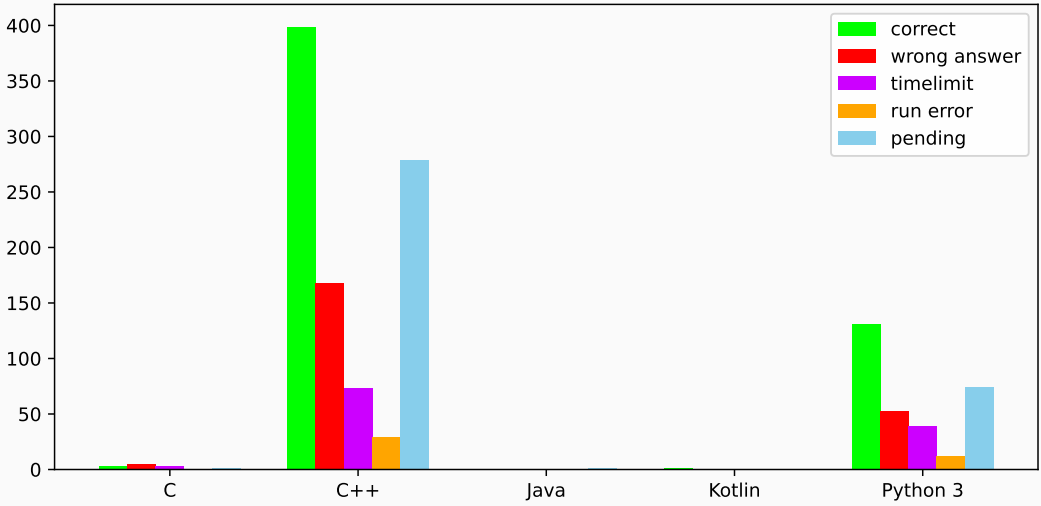
Alternate Approach

Same insights possible from game mechanics.

- Flood-fill areas of moving tiles with a 2×2 stencil.
- Exclude from each area the single tile topologically closest to the bottom right square.
- Compute permutation parity on each area as above.

Statistics: 17 submissions, 0 accepted, 15 unknown

Language stats



Random facts

Jury work

- 922 commits (including test session) (last year: 720)
- 1281 secret test cases (last year: 1424) ($\approx$ 98.5 per problem!)
- 356 jury/proofreader solutions (last year: 239)
- The minimum[1] number of lines the jury needed to solve all problems is:

$$1 + 23 + 17 + 13 + 1 + 21 + 38 + 6 + 37 + 6 + 39 + 21 + 11 = 234$$

  On average 18 lines per problem, up from 13.6 last year.

Jury dedication

- The test cases were finished on the Wednesday before the contest.
- However, the problem slides (for the livestream) and the solution slides were only finished today.

[1] With some code golfing, last year we golfed more