ICPC International Collegiate Programming Contest // 2024-2025

The 2025 ICPC Latin America Championship

icpc global sponsor
programming tools

icpc diamond
multi-regional sponsor

Problems and Statistics

March 16, 2025

Problem C - Coatless in Yakutsk

Author: Giovanna Kobus

Problem C - Coatless in Yakutsk

Author: Giovanna Kobus

Description

Find the largest temperature t such that there are no more than C consecutive days with a temperature smaller than t.

Problem C - Coatless in Yakutsk

Author: Giovanna Kobus

Description

Find the largest temperature t such that there are no more than C consecutive days with a temperature smaller than t.

Solution Idea

Many possible solutions: two pointers, binary search on the answer, sliding windows...

One idea: iterate through each possible temperature and check the condition

Problem C - Coatless in Yakutsk

Author: Giovanna Kobus

Description

Find the largest temperature t such that there are no more than C consecutive days with a temperature smaller than t.

Solution Idea

Many possible solutions: two pointers, binary search on the answer, sliding windows...

One idea: iterate through each possible temperature and check the condition

Complexity

Scan the temperatures: $T_{max} - T_{min} \to 100$

Check the array: $N \to 10^5$

Problem C - Coatless in Yakutsk: Statistics

Author: Giovanna Kobus

First AC

Team	[UFCG] CACHE OVERFLOW
Time	5 minutes

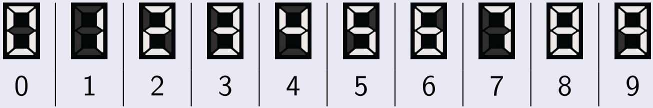
Number of Submissions	53
Number of Accepts	40 (75%)
Teams that attempted	40
Teams that got it	40 (100%)
Submissions after frozen	0

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Description

- Seven-segment display

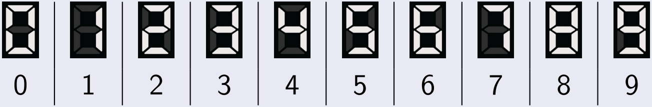


Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Description

- Seven-segment display



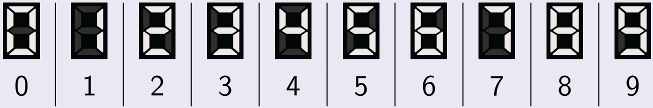
- Some segments are not working properly: always on/off

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Description

- Seven-segment display



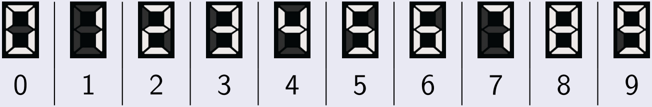
- Some segments are not working properly: always on/off
- Figure out what digit was meant to appear in the display (or print $*$ if there is more than one option)

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Description

- Seven-segment display



- Some segments are not working properly: always on/off
- Figure out what digit was meant to appear in the display (or print $*$ if there is more than one option)
- At most 10^5 digits, a 7-character string for each digit:
  - `G`: Good LED turned on
  - `g`: Good LED turned off
  - `+`: Always-on LED
  - `-`: Always-off LED

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Examples

- Ex: `+++gG--` could be a 0 (GGGgGGG), but not an 8 (GGGGGGG).
- Ex2: `-+-+-+-` could be anything

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Examples

- Ex: `+++gG--` could be a 0 (GGGgGGG), but not an 8 (GGGGGGG).
- Ex2: `-+-+-+-` could be anything

Solution Idea

- Brute force on the 10 possible digits: ignore + and −, see if the rest match

Problem L - LED Counter

Author: Alejandro Strejilevich de Loma

Examples

- Ex: +++gG-- could be a 0 (GGGgGGG), but not an 8 (GGGGGGG).
- Ex2: -+-+-+- could be anything

Solution Idea

- Brute force on the 10 possible digits: ignore + and −, see if the rest match
- Worst case $10^5 \cdot 10 \cdot 7 = 7 \cdot 10^6$ (cheap) operations

Problem L - LED Counter: Statistics

Author: Alejandro Strejilevich de Loma

First AC

Team	[UFMG] Torcida Pão de Alho™
Time	10 minutes

Number of Submissions	44
Number of Accepts	40 (90%)
Teams that attempted	40
Teams that got it	40 (100%)
Submissions after frozen	0

Problem G - Game of Pieces

Author: Rafael Grandsire

Description

You are being asked to simulate a different version of the *Tetris game*, where pieces are only horizontal or vertical, but you have no limit for the number of columns in the game or the size of the pieces.

Problem G - Game of Pieces

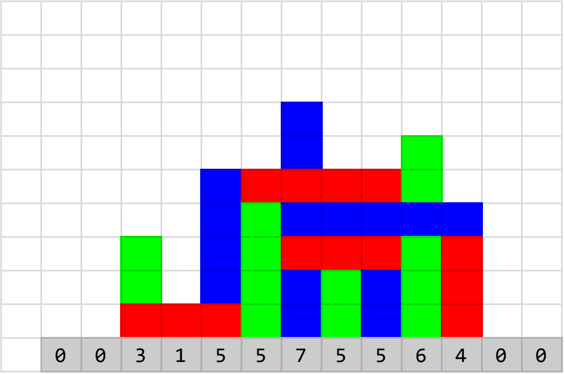
Author: Rafael Grandsire

Description

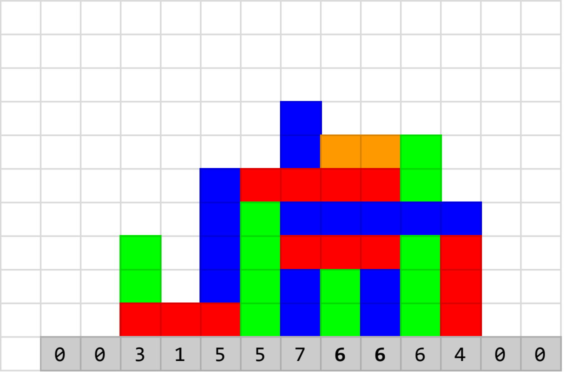
You are being asked to simulate a different version of the *Tetris game*, where pieces are only horizontal or vertical, but you have no limit for the number of columns in the game or the size of the pieces.

The special thing about this simuilation is that we won't allow pieces to create any gap.

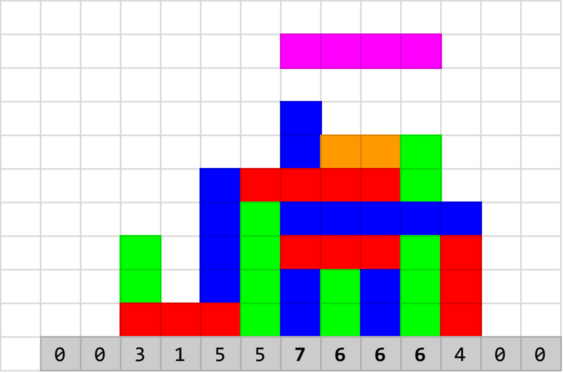
Once it happens, this piece is taken out of the board.



0	0	3	1	5	5	7	5	5	6	4	0	0



0	0	3	1	5	5	7	6	6	6	4	0	0



0 0 3 1 5 5 7 6 6 6 4 0 0

Solution Idea

When the game is always safe, the board can be represented by the height of the columns.

From this point of view, a piece makes the game unsafe if it ranges over two (or more) different heights.

Solution Idea

When the game is always safe, the board can be represented by the height of the columns.

From this point of view, a piece makes the game unsafe if it ranges over two (or more) different heights.

We can run the simulation with three approaches:

- Representing ranges as triples [*left*, *right*, *value*] and supporting the operations using a set;
- Using a sparse segment tree supporting queries to check for equality and updating heights according to the shape of the pieces;
- Offline: using coordinate compression and usual segtree.

Complexity is $\mathcal{O}(n\log(n))$ or $\mathcal{O}(n\log(P))$.

Problem G - Game of Pieces: Statistics

Author: Rafael Grandsire

First AC

Team	[USP] Faça o WM
Time	63 minutes

Number of Submissions	87
Number of Accepts	24 (27%)
Teams that attempted	33
Teams that got it	24 (72%)
Submissions after frozen	37

Problem D - Dangerous City

Author: Luan Arcanjo

Problem D - Dangerous City

Author: Luan Arcanjo

Description

Given a connected graph with weights in the nodes. Find, for each node u, find the sum of the minimum cost to go from u to every other node in the graph, Where the cost of a path is defined by maximum weight of a node contained in the path.

Problem D - Dangerous City

Author: Luan Arcanjo

Solution Idea

We are only interested in the MST of the graph.
Build the reachability tree while performing DSU to build the MST. The weight of a created node in the reachability tree is the maximum weight of the nodes in its subtree.

Problem D - Dangerous City

Author: Luan Arcanjo

Solution Idea

If we compute, for every node in the reachability tree u, $leaf_u$ as the number of leaves in its subtree and look at the path from a leaf l_i to the root of the reachability tree, we can compute the answers by:

$$Answer(l_i) = \sum_{i=1}^{k-1} (leaf_{v_i} - leaf_{v_{i+1}}) w_{v_i} + w_{l_i}$$

Where the $root = v_1, v_2, ..., v_k = l_i$ is the path from the root to l_i. We can compute this formula for all leaves, which are the nodes of the original graph, by performing a BFS/DFS on the reachability tree. Complexity: $O(nlog(n))$.

Problem D - Dangerous City

Author: Luan Arcanjo

Alternative solutions

There are other solutions using *small to large* or building a *treap* while building the MST.

Problem D - Dangerous City: Statistics

Author: Luan Arcanjo

First AC

Team	[UdG CUCEI] Motomomis
Time	70 minutes

Number of Submissions	32
Number of Accepts	17 (53%)
Teams that attempted	22
Teams that got it	17 (77%)
Submissions after frozen	23

Problem A - Ananna

Author: Giovanna Kobus

Problem A - Ananna

Author: Giovanna Kobus

Description

Given a graph, find the number of pair (u, v) such that there is a palindromic path from u to v.

Problem A - Ananna

Author: Giovanna Kobus

Solution Idea

Keep event queue with currently found pairs

Initialize with pairs (u, u)

For each event (u, v) add all pairs (u', v') such that $u' \to^c u$ and $v \to^c v'$.

Problem A - Ananna

Author: Giovanna Kobus

Solution Idea

Keep event queue with currently found pairs

Initialize with pairs (u, u)

For each event (u, v) add all pairs (u', v') such that $u' \to^c u$ and $v \to^c v'$.

Complexity

Each event (u, v) takes $\textsc{indeg}(u) \times \textsc{outdeg}(v)$.

Sum of all events takes $\sum_{u,v} \textsc{indeg}(u) \times \textsc{outdeg}(v) = m^2$.

Problem A - Ananna: Statistics

Author: Giovanna Kobus

First AC

Team	[UNI-FC] UnratedLegendaryGrandMasters
Time	56 minutes

Number of Submissions	34
Number of Accepts	7 (20%)
Teams that attempted	18
Teams that got it	7 (38%)
Submissions after frozen	19

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Description

Find two non-empty disjoint subsets, such that the XOR of the elements of the first is equal to the XOR of the elements of the second, and both subsets have the same size.

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Observation 1

If there is a set with XOR 0, then any partition of this set into two subsets will have the same XOR.

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Observation 1

If there is a set with XOR 0, then any partition of this set into two subsets will have the same XOR.

Observation 2

Find a non empty set with XOR 0 and even size. The solution is any partition of this set into two subsets of the same size.

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Observation 1

If there is a set with XOR 0, then any partition of this set into two subsets will have the same XOR.

Observation 2

Find a non empty set with XOR 0 and even size. The solution is any partition of this set into two subsets of the same size.

Algorithm

Finding an arbitrary non-empty set with XOR 0 can be done using Gaussian Elimination over $\mathbb{Z}_2$.

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Solution

- Add an extra bit set in 1 to all numbers in the array
- Find a set with XOR 0 using the new elements.

Any solution in this updated array will be a solution in the original array, and will have even size.

Problem B - Brazilian FootXOR

Author: Jorge Alejandro Pichardo Cabrera

Complexity

Gaussian elimination: $O(n \cdot k^2)$, where n is the number of elements and k is the number of bits in each element.

Problem B - Brazilian FootXOR: Statistics

Author: Jorge Alejandro Pichardo Cabrera

First AC

Team	[UFMG] Torcida Pão de Alho™
Time	32 minutes

Number of Submissions	19
Number of Accepts	6 (31%)
Teams that attempted	11
Teams that got it	6 (54%)
Submissions after frozen	12

Problem K - Keep Fighting

Author: Luan Arcanjo

Problem K - Keep Fighting

Author: Luan Arcanjo

Description

Given game setup. find the minimum number of cards needed to beat the monster with the restriction that the difference in the number of times a card is played between any two cards is at most 1.

Problem K - Keep Fighting

Author: Luan Arcanjo

Solution Idea

- If Bob's power cannot be greater than 0, or if there are no attack cards, it is impossible to beat the monster.
- If it is not possible to increase Bob's power, it is optimal to always use the attack cards when they are available. The number of attack cards required to beat the monster is $\lceil \frac{h}{p} \rceil$. Handle this case separately.

Problem K - Keep Fighting

Author: Luan Arcanjo

Solution Idea

- If it is not possible to finish the game by using each card exactly once, it is optimal to play the attack cards first, then the multiply cards, and finally the add cards. This process will happen $O(\sqrt{h})$ times.
- If it is possible to finish the game without using each card exactly once, the problem's bounds allow brute-forcing the number of each type of card to be played.
- Remember to sort the add and multiply cards by power and to handle the *** 1** card.

  Complexity: $O(n\sqrt{h})$.

Problem K - Keep Fighting: Statistics

Author: Luan Arcanjo

First AC

Team	[UFMG] Torcida Pão de Alho™
Time	81 minutes

Number of Submissions	67
Number of Accepts	5 (7%)
Teams that attempted	22
Teams that got it	5 (22%)
Submissions after frozen	86

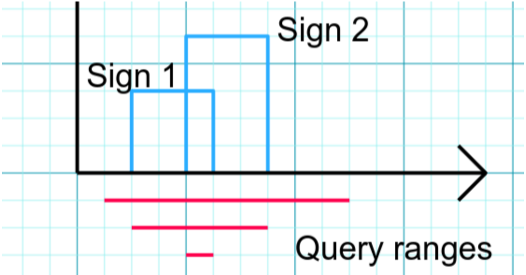
Problem F - Festival Signs

Author: Agustín Santiago Gutiérrez

Process 3 operations:

- Add rectangle
- Remove rectangle
- Query minimum height in a range

For every rectangle: Bottom side is contained in the $y = 0$ line



Problem F - Festival Signs

Author: Agustín Santiago Gutiérrez

Solution Idea

- Add rectangle + Queries:
  - Segment Tree with Lazy Propagation
- Implement rollback capability
  - Option 1: Keep a stack of array modifications
  - Option 2: Persistent Segment Tree
- "Offline dynamic connectivity" to allow removals

Problem F - Festival Signs

Author: Agustín Santiago Gutiérrez

Complexity

- Offline dynamic connectivity → $O(N \lg N)$ operations
- Each operation is a Segment Tree operation or a Rollback
- Total time: $O(N \lg^2 N)$
- $O(N\sqrt{N} \lg N)$ approaches: can get AC if well optimized

Problem F - Festival Signs: Statistics

Author: Agustín Santiago Gutiérrez

First AC

Team	[USP] Faça o WM
Time	180 minutes

Number of Submissions	2
Number of Accepts	2 (100%)
Teams that attempted	2
Teams that got it	2 (100%)
Submissions after frozen	9

Problem I - Infinite Arrays

Author: Luis Santiago Re

Description

Given an array P with **pairwise distinct** values, handle the following types of events:

- Given an array $A = A_1, A_2, \ldots, A_K$, find the length of longest common subarray between P^∞ and A^∞
- Delete a value X from P
- Insert a value Y into P, to the left of another value Z (**pairwise distinct** property remains true all the time)

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.For each position i as right end of a subarray, compute the left most valid beginning L_i.

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.For each position i as right end of a subarray, compute the left most valid beginning L_i.
- $PosOf_{P_i} = i$, i.e. $PosOf_V =$ position in P where value V is located.

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.For each position i as right end of a subarray, compute the left most valid beginning L_i.
- $PosOf_{P_i} = i$, i.e. $PosOf_V =$ position in P where value V is located.
- If $PosOf_{A_i} = (PosOf_{A_{i-1}} + 1)\%|P|$, then $L_i = L_{i-1}$.

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.For each position i as right end of a subarray, compute the left most valid beginning L_i.
- $PosOf_{P_i} = i$, i.e. $PosOf_V =$ position in P where value V is located.
- If $PosOf_{A_i} = (PosOf_{A_{i-1}} + 1)\%|P|$, then $L_i = L_{i-1}$.
- If $PosOf_{A_i} \neq (PosOf_{A_{i-1}} + 1)\%|P|$, then $L_i = i$ (or $A_i \notin P$).

Problem I - Infinite Arrays

Author: Luis Santiago Re

Solution Idea

- For a query: $ans \leq K$, or $ans = *$.
- Iterate over A^2.For each position i as right end of a subarray, compute the left most valid beginning L_i.
- $PosOf_{P_i} = i$, i.e. $PosOf_V =$ position in P where value V is located.
- If $PosOf_{A_i} = (PosOf_{A_{i-1}} + 1)\%|P|$, then $L_i = L_{i-1}$.
- If $PosOf_{A_i} \neq (PosOf_{A_{i-1}} + 1)\%|P|$, then $L_i = i$ (or $A_i \notin P$).
- How to handle insert/delete? Linked list (or treap).

Problem I - Infinite Arrays: Statistics

Author: Luis Santiago Re

First AC

Team	[UNI-FC] UnratedLegendaryGrandMasters
Time	180 minutes

Number of Submissions	1
Number of Accepts	1 (100%)
Teams that attempted	1
Teams that got it	1 (100%)
Submissions after frozen	12

Problem J - Just Look Up

Author: Giovanna Kobus

Problem J - Just Look Up

Author: Giovanna Kobus

Description

Find the biggest empty "circle" in the surface of a sphere. Points are given as 3D coordinates to be projected.

Problem J - Just Look Up

Author: Giovanna Kobus

First Solution Idea

Largest empty circle must have at least three points in its edge.

- Iterate through all triples of points
- Find circle that passes through them
- Check if it is empty

Problem J - Just Look Up

Author: Giovanna Kobus

First Solution Idea

Largest empty circle must have at least three points in its edge.

- Iterate through all triples of points
- Find circle that passes through them
- Check if it is empty

Complexity

$O(n^4) \rightarrow$ too slow!

Problem J - Just Look Up

Author: Giovanna Kobus

First Solution Idea

Let us iterate through the pairs and try to find largest circle!

- Iterate through all triples of points
- Find circle that passes through them
- Check if it is empty

Problem J - Just Look Up

Author: Giovanna Kobus

First Solution Idea

Let us iterate through the pairs and try to find largest circle!

- Iterate through all triples of points
- Find circle that passes through them
- Check if it is empty

Complexity

$O(n^4) \rightarrow$ too slow!

Problem J - Just Look Up

Author: Giovanna Kobus

First Solution Idea

Let us iterate through the pairs and try to find largest circle!

- Iterate through all triples of points
- Find circle that passes through them
- Check if it is empty

Complexity

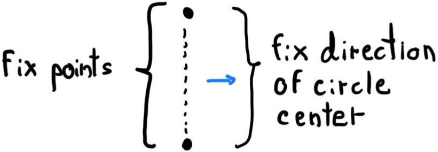
$O(n^4) \rightarrow$ too slow!

Let us iterate through pairs instead.

Problem J - Just Look Up

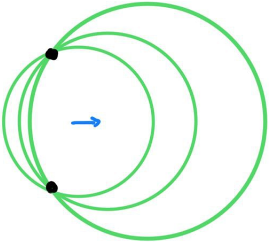
Author: Giovanna Kobus

Similar to finding the 3D convex hull.



Problem J - Just Look Up

Author: Giovanna Kobus



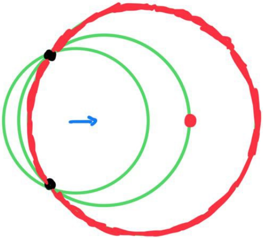
↑ radius

↓ left part

↑ right part

Problem J - Just Look Up

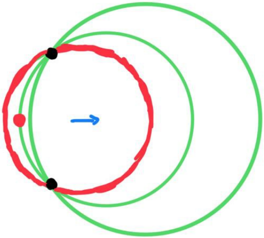
Author: Giovanna Kobus



point to the right
"upper_bounds"
the radius

Problem J - Just Look Up

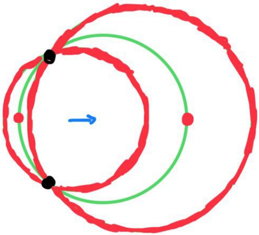
Author: Giovanna Kobus



point to the left
"lower-bounds"
the radius

Problem J - Just Look Up

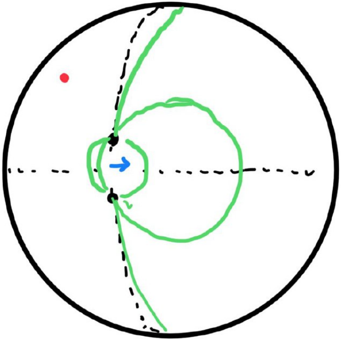
Author: Giovanna Kobus



Answer for pair: largest circle in range

Problem J - Just Look Up

Author: Giovanna Kobus



Of course, we are in a sphere.

Problem J - Just Look Up

Author: Giovanna Kobus

Solution Idea

- Iterate through all pairs of points and center directions.
- Find largest circle that passes through them and has center in the direction.

Problem J - Just Look Up

Author: Giovanna Kobus

Solution Idea

- Iterate through all pairs of points and center directions.
- Find largest circle that passes through them and has center in the direction.

Complexity

$O(n^2) \times O(n) \rightarrow$ good enough! You can also use your own 3D convex hull code to make life easier.

Problem J - Just Look Up: Statistics

Author: Giovanna Kobus

First AC

Team	?
Time	?

Number of Submissions	?
Number of Accepts	? (no submissions)
Teams that attempted	?
Teams that got it	? (no submissions)
Submissions after frozen	17

Problem E - Exciting Business Opportunities

Author: Roberto Sales

Problem E - Exciting Business Opportunities

Author: Roberto Sales

Description

You are given a tree and a list of operations:

- Open a business in node X
- Add a sponsor in node X

A continuous sub-array is *good* if, when considering operations only in this range, all the opened businesses in it are in the path between at least two sponsors.

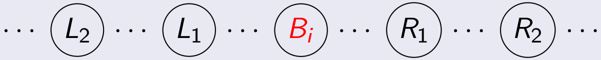
For each $1 \leq i \leq N$, compute the length of the largest good sub-array starting at i.

Problem E - Exciting Business Opportunities

Author: Roberto Sales

Solution Idea

- For a business operation on node B_i, find the two sponsors on each side (left and right separately) such that they belong to different sub-trees of B_i.



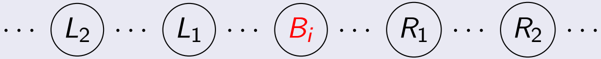
- These positions can be computed for each B_i by using a segment tree.

Problem E - Exciting Business Opportunities

Author: Roberto Sales

Solution Idea

- Given a fixed start for the good sub-array, the node B_i might cause some end positions to become invalid. Be careful of sponsors that are in the same node as a business:
  - If we start in $[0, L_2]$, the node B_i doesn't break any endings.
  - If we start in $(L_2, L_1]$, we can't finish in $[B_i, R_1)$. You should be careful when L_1 and R_1 belong to the same sub-tree.
  - If we start in $(L_1, B_i]$, we can't finish in $[B_i, R_2)$.



Problem E - Exciting Business Opportunities

Author: Roberto Sales

Solution Idea

- Use a sweep line fixing the start of the sub-array while maintaining, for each position, whether it's a valid ending position.
    - Segment tree lazy of range addition, or simple segment tree with minimum prefix sum
- There is an alternative solution using divide and conquer.

Problem E - Exciting Business Opportunities: Statistics

Author: Roberto Sales

First AC

Team	?
Time	?

Number of Submissions	1
Number of Accepts	? (no submissions)
Teams that attempted	1
Teams that got it	? (no submissions)
Submissions after frozen	1

Problem H - Horrible Restaurants

Author: Agustín Santiago Gutiérrez

Description

- N restaurants
- Can assign 0, 1, 2 or 3 stars to each
- Cost matrix $C_{i,j}$ is given $(1 \leq i \leq N, 1 \leq j \leq 3)$
- For every possible K, compute min cost for K total stars

Problem H - Horrible Restaurants

Author: Agustín Santiago Gutiérrez

Solution Idea

- Let S be **any** optimal solution for K stars
- **Some** optimal solution for $K+1$ can be reached from S by using one of these transformations:
  - $+1$ star to a restaurant
  - $+2$ stars to a restaurant, -1 star to another one
  - $+3$, -2 (two restaurants involved)
  - $+3$, -1, -1 (three restaurants involved)
  - $+2$, $+2$, -3 (three restaurants involved)

Complexity

- Use sets / heaps / segtrees to find the best option
- Total time: $O(N \lg N)$

Problem H - Horrible Restaurants: Statistics

Author: Agustín Santiago Gutiérrez

First AC

Team	?
Time	?

Number of Submissions	1
Number of Accepts	? (no submissions)
Teams that attempted	1
Teams that got it	? (no submissions)
Submissions after frozen	1