Problem Explanation

Huazhong Agricultural University Problem Setting Team

The 14th Huazhong Agricultural University Programming Contest

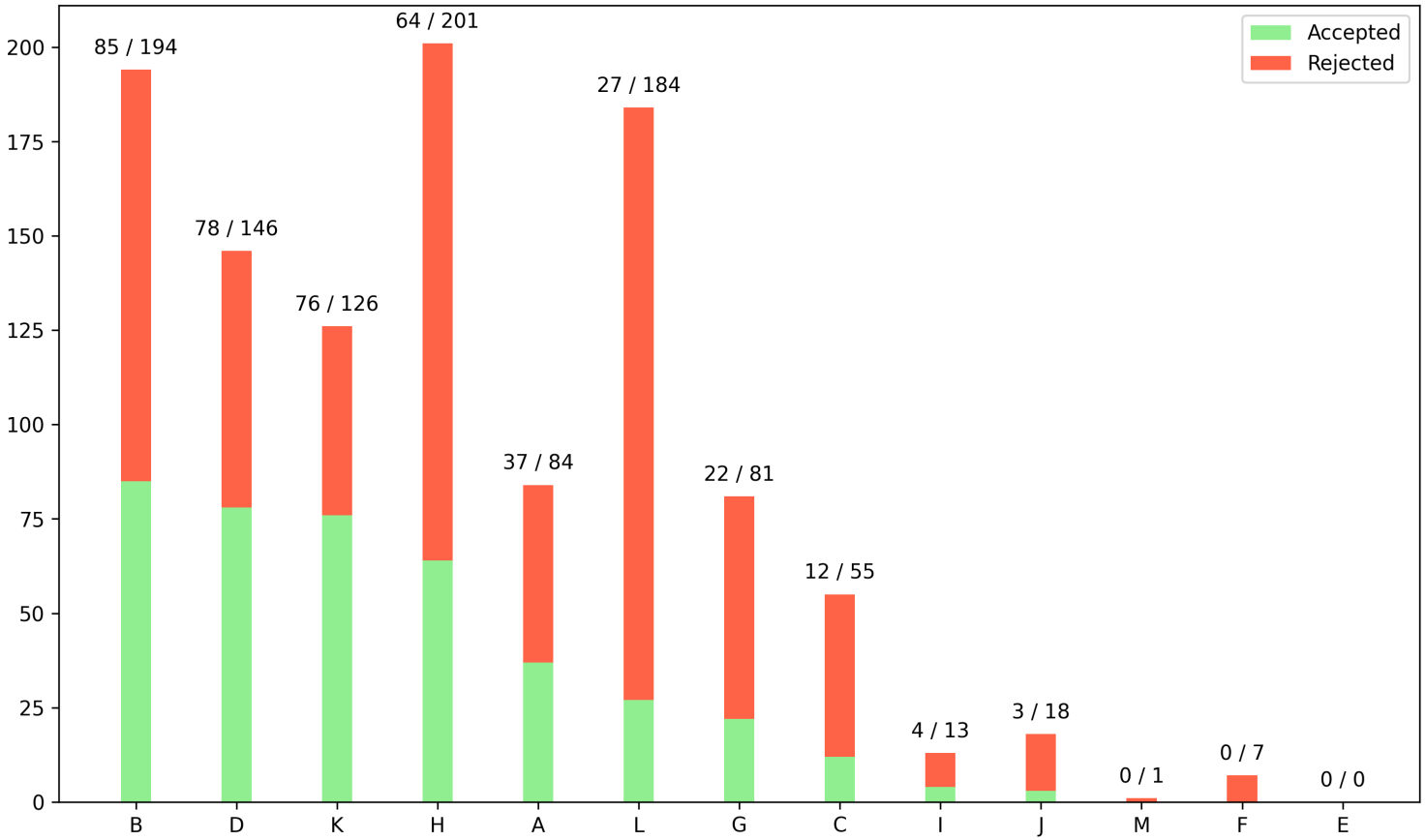
April 13, 2025

Overview

- Easy: B D
- Medium-Easy: H K
- Medium: A C G
- Medium-Hard: E I J
- Hard: F M
- Guess: L

Overview

Before the scoreboard frozen, the submission status for each problem is as follows:



B. Lunch!

Problem Description

Given a_1, a_2, a_3, perform swap operations to make sure that $a_1 > a_2$ and $a_2 < a_3$.

- In fact, sufficient swap operations mean we can arrange the sequence freely. From $a_1 > a_2$ and $a_2 < a_3$, we can deduce that a_2 is the smallest number among a_1, a_2, a_3.
- We can sort the set $\{a_1, a_2, a_3\}$, and let the sorted sequence be $\{b_1, b_2, b_3\}$. We only need to check whether $b_1 < b_2$ and $b_1 < b_3$.

D. Cowardly Lizard V

Problem Description

Given a sequence $a_1, a_2, \ldots, a_n$. Find $x(1 \leq x \leq n)$ such that there is no a_i equal to x.

- Simply check whether all elements in $a_1, a_2, \ldots, a_n$ are distinct. If $a_1, a_2, \ldots, a_n$ are all distinct, then there is no solution.
- Otherwise, use a memoization array to record numbers that do not appear in $a_1, a_2, \ldots, a_n$, and output the result.
- Time complexity is $O(n)$.

H. Defense Deployment

Problem Description

Find the number of pairs of points such that either their x-coordinates or y-coordinates are equal.

- For a given x-coordinate x_t, suppose there are m points with the same x-coordinate, then the contribution of this x-coordinate to the answer is $\frac{m(m-1)}{2}$; similarly for the y-coordinate.
- For implementation, you can use `std::map` or directly use an array to count the occurrences.

K. ruskal's Reconstruction Number

Problem Description

Given a number. Perform at most one operation to swap adjacent digits, in order to maximize the result.

- Treat the number as a string s, find the smallest i such that $s_i < s_{i+1}$, and swap s_i and s_{i+1}.
- If no such i exists, do nothing.

K. ruskal's Reconstruction Number

Proof

First, if $s_i \geq s_{i+1}$, then swapping s_i and s_{i+1} will not result in a greater value.

Let x be the number of indices i where $s_i \geq s_{i+1} (1 \leq i < n)$.

Case 1

If $x = 0$, clearly, not performing the swap is optimal.

Case 2

If $x = 1$, we only need to find the smallest i such that $s_i < s_{i+1}$ and swap s_i and s_{i+1}.

K. ruskal's Reconstruction Number

Proof

Case 3

If $x \geq 2$, there exist $i_1, i_2, \ldots, i_x$ such that $s_{i_p} < s_{i_p+1} (1 \leq p \leq x)$.
Let us assume that $\{i_p\}$ is increasing. It is easy to see that for all $2 \leq p \leq x$, we have $i_1 < i_p$.
If we swap s_{i_1} and s_{i_1+1}, then the i_1-th digit becomes s_{i_1+1}; if we swap s_{i_p} and s_{i_p+1}, then the i_1-th digit remains s_{i_1}.
Since $s_{i_1} < s_{i_1+1}$, the optimal solution is to swap s_{i_1} and s_{i_1+1}.

Proof complete.

A. A New Journey

Problem Description

The selection consists of two aspects: training session results and problem-solving performance. You need to calculate the score for each contestant in these two aspects.

- Simply simulate according to the problem description. Be mindful of precision during the simulation: when calculating the score for a single training session, use integer operations instead of floating-point operations; when calculating the arithmetic mean of the top k training session scores, first sum the k scores and then divide by k.
- Time complexity is $O(n(m + k + n \log n))$.

C. Cowardly Lizard IV

Problem Description

Given q queries, for each query, given l and r, find the value of $\min_{k=l}^{r-1}\left\{\frac{1}{\sum_{i=l}^{k} a_i}+\frac{1}{\sum_{i=k+1}^{r} a_i}\right\}$.

- Notice that: $\frac{1}{\sum_{i=l}^{k} a_i}+\frac{1}{\sum_{i=k+1}^{r} a_i}=\frac{\sum_{i=l}^{r} a_i}{\sum_{i=l}^{k} a_i \times \sum_{i=k+1}^{r} a_i}$
- For a given l and r, $\sum_{i=l}^{r} a_i$ is fixed, so we only need to maximize $\sum_{i=l}^{k} a_i \times \sum_{i=k+1}^{r} a_i$.

C. Cowardly Lizard IV

- Also, since $\sum_{i=l}^{k} a_i + \sum_{i=k+1}^{r} a_i = \sum_{i=l}^{r} a_i$, the smaller $\left|\sum_{i=l}^{k} a_i - \sum_{i=k+1}^{r} a_i\right|$ is, the larger $\sum_{i=l}^{k} a_i \times \sum_{i=k+1}^{r} a_i$ becomes.

C. Cowardly Lizard IV

- Since $\sum_{i=l}^{k} a_i - \sum_{i=k+1}^{r} a_i$ is monotonic with respect to k, we only need to find k_1 and k_2 such that:
  1. $\sum_{i=l}^{k_1} a_i \le \sum_{i=k_1+1}^{r} a_i$ and $\sum_{i=l}^{k_1} a_i - \sum_{i=k_1+1}^{r} a_i$ is as close to 0 as possible.
  2. $\sum_{i=l}^{k_2} a_i \ge \sum_{i=k_2+1}^{r} a_i$ and $\sum_{i=l}^{k_2} a_i - \sum_{i=k_2+1}^{r} a_i$ is as close to 0 as possible.

  Then, compare the results of selecting k_1 and k_2.
- Use prefix sums to handle the range sums of $\{a\}$, with a time complexity of $O(n + q(\log n + \log V))$.

G. Who Likes Mathematics is not Boki-chan

Problem Description

Given the interval $[L, R]$, find the number of numbers in this interval where the absolute difference between every two adjacent digits is 1.

- Note that the valid value range does not exceed 10^{18}, meaning the length of valid numbers is at most 19 digits. We can enumerate all numbers that satisfy the condition where the absolute difference between every two adjacent digits is 1, and check if they fall within the given interval.
- The time complexity is $O(9 \times 2^k)$, where k is the number of digits in the enumerated value range.

E. Creative Boki-chan

Problem Description

Given $b_1, b_2, \ldots, b_{n+k}$, determine if there exists an original sequence $a_1, a_2, \ldots, a_n$.

- Let dp_i represent whether the subsequence $b_1, b_2, \ldots, b_i$ has a corresponding original sequence. Consider how dp_i can be updated for the next state.

E. Creative Boki-chan

- If $dp_i = 1$, consider $dp_j (i+1 < j \leq n+k)$. If the following condition holds:

$$b_{i+1} = \sum_{t=i+2}^{j} (t-i-1)^2 \times b_t$$

or

$$b_j = \sum_{t=i+1}^{j-1} (t-i)^2 \times b_t$$

Then set $dp_j = 1$. Finally, you only need to check the value of dp_{n+k}.

E. Creative Boki-chan

- Notice that $\sum_{i=l}^{r}(i-l+1)^2 \times a_i \geq \sum_{i=l}^{r}(i-l+1)^2$.
- Let $x = r - l + 1$, the length of the sub-segment.
- Thus, $\sum_{i=l}^{r}(i-l+1)^2 = \dfrac{x(x+1)(2x+1)}{6} \leq 10^8$. This gives $x \leq 669$.
- This means that for a given i, when enumerating possible updates to dp_j, we only need to consider sub-segments with a length of at most 669.
- When implementing, note that the given sequence length in the problem is n.
- The time complexity is $O(669n)$.

I. We Must Be Together No Matter How Far

Problem Description

Given an undirected connected graph with n points and n edges. There are multiple queries, and for each query, given k, you need to answer whether it is possible to walk exactly k steps from the start point to the end point.

- First, an undirected connected graph with n points and n edges is a tree with one cycle (a fundamental cycle tree).

I. We Must Be Together No Matter How Far

- Analyze the shortest path length from the start point to the end point. If the shortest path length from the start point to the end point is greater than k, the answer is definitely no.
- Otherwise, consider the following cases:
  1. The length of the cycle is even.
  2. The length of the cycle is odd.

I. We Must Be Together No Matter How Far

Case 1

If the cycle length is even, since there are no odd cycles, if the parity of the shortest path length from the start point to the end point is different from k, the answer is definitely no.

Case 2

If the cycle length is odd, since there are odd cycles, there are two paths on the cycle from the start point to the end point, one odd and one even in length. Among them, there is at least one path where the parity of the path length from the start point to the end point matches k.

- If k is greater than the current chosen path length and both have the same parity, we can repeat walking along one edge to make the total path length exactly k.

I. We Must Be Together No Matter How Far

- In addition, note the case where the start point and the end point are in the same tree of the fundamental cycle tree. In this case, the shortest path can be calculated using LCA (Lowest Common Ancestor).

J. Skill Tree

Problem Description

Given a skill tree with node 1 as the root, each node has a combat power value. To light up a node, its parent node must already be lit (the root node can be lit directly), and each node lighting up costs 1 coin. The combat power gained by lighting up a node is the median of the combat power values of all nodes on the path from the node to the root, sorted in increasing order. Find the maximum combat power that can be obtained with no more than m coins.

Note

The definition of the median in this problem is slightly different from the standard definition.

J. Skill Tree

- First, we can preprocess the combat power that each node can contribute. We only need to perform a DFS traversal of the entire tree while maintaining a weighted segment tree.
- When visiting a node, insert the combat power value of this node into the weighted segment tree, and delete it when backtracking.

J. Skill Tree

- Next, we define $dp_{u,k}$ as the maximum combat power that can be obtained by selecting k nodes in the subtree rooted at u.
- For each child node v of u, we have:

$$dp_{u,i+j} = \max_{v}\{dp_{u,i} + dp_{v,j}\}$$

- For each u, we initialize $dp_{u,1} = v_u$, which means the combat power obtained by selecting only u.
- During this process, for each node u, we ensure that u is selected, and then perform a 0-1 knapsack on its child nodes. The final answer is $\max_{k}\{dp_{u,k}\}$.
- The time complexity is $O(n \log n + nm)$.

F. Boki-chan Who Dislikes Mathematics

Problem Description

Define $Bq(x, y, v) = \mathrm{d}(\gcd(x, y))^{[\mathrm{d}(\gcd(x,y))\le v]}$, and find $\prod_{i=1}^{n}\prod_{j=1}^{m} Bq(i, j, v)$.

We have:

$$\prod_{i=1}^{n}\prod_{j=1}^{m}\mathrm{d}(\gcd(i,j)) = \prod_{i=1}^{n}\prod_{j=1}^{m}\mathrm{d}(x)\,[\gcd(i,j)=x]$$

$$= \prod_{x=1}^{\min(n,m)} \mathrm{d}(x)^{\sum_{i=1}^{n}\sum_{j=1}^{m}[\gcd(i,j)=x]}.$$

F. Boki-chan Who Dislikes Mathematics

Consider the exponent part:

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{m}[\gcd(i,j)=x] &= \sum_{i=1}^{\lfloor \frac{n}{x} \rfloor}\sum_{j=1}^{\lfloor \frac{m}{x} \rfloor}[\gcd(i,j)=1] \\
&= \sum_{i=1}^{\lfloor \frac{n}{x} \rfloor}\sum_{j=1}^{\lfloor \frac{m}{x} \rfloor}\sum_{k=1}^{\min(\lfloor \frac{n}{x} \rfloor,\lfloor \frac{m}{x} \rfloor)}\mu(k)[k \mid i][k \mid j] \\
&= \sum_{k=1}^{\min(\lfloor \frac{n}{x} \rfloor,\lfloor \frac{m}{x} \rfloor)}\mu(k)\sum_{i=1}^{\lfloor \frac{n}{x} \rfloor}[k \mid i]\sum_{j=1}^{\lfloor \frac{m}{x} \rfloor}[k \mid j] \\
&= \sum_{k=1}^{\min(\lfloor \frac{n}{x} \rfloor,\lfloor \frac{m}{x} \rfloor)}\mu(k)\left\lfloor \frac{n}{kx} \right\rfloor\left\lfloor \frac{m}{kx} \right\rfloor.
\end{aligned}$$

F. Boki-chan Who Dislikes Mathematics

Let $kx = T$, so $k = \dfrac{T}{x}$, we have:

$$\sum_{k=1}^{\min(\lfloor \frac{n}{x} \rfloor, \lfloor \frac{m}{x} \rfloor)} \mu(k) \left\lfloor \frac{n}{kx} \right\rfloor \left\lfloor \frac{m}{kx} \right\rfloor = \sum_{\substack{T|x \\ T \le \min(n,m)}} \mu\left(\frac{T}{x}\right) \lfloor \frac{n}{T} \rfloor \lfloor \frac{m}{T} \rfloor$$

F. Boki-chan Who Dislikes Mathematics

Therefore:

$$\prod_{x=1}^{\min(n,m)} \mathrm{d}(x)^{\sum_{i=1}^{n}\sum_{j=1}^{m}[\gcd(i,j)=x]} = \prod_{x=1}^{\min(n,m)} \mathrm{d}(x)^{\sum\limits_{T\mid x,\,T\le\min(n,m)} \lfloor\frac{n}{T}\rfloor\lfloor\frac{m}{T}\rfloor}$$

$$= \prod_{x=1}^{\min(n,m)} \prod_{\substack{T\mid x\\ T\le\min(n,m)}} \mathrm{d}(x)^{\mu(\frac{T}{x})\lfloor\frac{n}{T}\rfloor\lfloor\frac{m}{T}\rfloor}$$

$$= \prod_{\substack{T\mid x\\ T\le\min(n,m)}} \left(\prod_{x=1}^{\min(n,m)} \mathrm{d}(x)^{\mu(\frac{T}{x})}\right)^{\lfloor\frac{n}{T}\rfloor\lfloor\frac{m}{T}\rfloor}.$$

F. Boki-chan Who Dislikes Mathematics

That is:

$$\prod_{i=1}^{n}\prod_{j=1}^{m}\mathrm{d}(\gcd(i,j)) = \prod_{\substack{T|x\\ T\le\min(n,m)}}\left(\prod_{x=1}^{\min(n,m)}\mathrm{d}(x)^{\mu(\frac{T}{x})}\right)^{\lfloor\frac{n}{T}\rfloor\lfloor\frac{m}{T}\rfloor}$$

F. Boki-chan Who Dislikes Mathematics

- Let $F(T) = \prod_{x=1}^{\min(n,m)} \mathrm{d}(x)^{\mu(\frac{T}{x})} = \prod_{x|T} \mathrm{d}(x)^{\mu(\frac{T}{x})}$.
- For any x, $\mathrm{d}(x)$ can only contribute to $F(T)$ if $\mathrm{d}(x) \leq v$.
- Consider offline processing of q queries and sorting by v in ascending order,
- Let the q queries be $v_1, v_2, \ldots, v_q$. Consider the effect of different v values on $F(T)$, and let the corresponding $F(T)$ values for these q queries be $F_1(T), F_2(T), \ldots, F_q(T)$.

F. Boki-chan Who Dislikes Mathematics

- Before all queries, $F_i(T) = 1(1 \le i \le q)$.
- After the i-th query, for all positive integers x, if $v_i < \mathrm{d}(x) \le v_{i+1}$, then for all $i < j \le q$, we have $F_j(T) \leftarrow F_j(T) \times \mathrm{d}(x)^{\mu(\frac{T}{x})}$, which can be maintained using a binary indexed tree.
- After updating $F(T)$, use number-theoretic block decomposition to compute the answer.
- The time complexity is $O(q\sqrt{n}(\log(V) + \log(n)) + n\log^2 n)$, where V is the modulus size.

M. Dye Your Color

Problem Description

Given a tree, maintain the coloring and query operations on the tree.

- Since $\dfrac{i \times j}{\operatorname{lcm}(i,j)} = \gcd(i,j)$, we have:

$$\operatorname{lcm}^2(i,j) - (i+j) \times \operatorname{lcm}(i,j) = i \times j$$

This is equivalent to:

$$\operatorname{lcm}(i,j) = i + j + \gcd(i,j)$$

M. Dye Your Color

- Let us assume $i \leq j$. When $i = j$, the equation clearly does not hold.
- When $i < j$, we have $1 \leq \gcd(i, j) \leq i < j$, so:

$$j < i + j + \gcd(i, j) < 3 \times j$$

- Since $i + j + \gcd(i, j) = \mathrm{lcm}(i, j)$, we must have:

$$j \mid i + j + \gcd(i, j)$$

- Therefore, $i + j + \gcd(i, j) = 2 \times j$.

M. Dye Your Color

- Therefore:

$$\frac{i \times j}{\gcd(i,j)} = \operatorname{lcm}(i,j) = i + j + \gcd(i,j) = 2 \times j$$

Thus:

$$i = 2 \times \gcd(i,j)$$

$$j = 3 \times \gcd(i,j)$$

So, $\frac{i}{j} = \frac{2}{3}$.

M. Dye Your Color

Definition: Maximum Connected Set

Define a non-empty set S as a maximum connected set if it satisfies the following conditions:

1. For any two distinct elements $i, j (i < j)$ in S, there exists a positive integer p such that $i \times 3^p = j \times 2^p$.
2. For any positive integer x not greater than n, if there exists an element y in S not equal to x and a positive integer p such that $x \times 3^p = y \times 2^p$, then x must be in S.

- Example: For $n = 100$, $S = \{8, 12, 18, 27\}$ is a maximum connected set. However, $\{8, 12, 18, 27, 30\}$ and $\{12, 18, 27\}$ are not.

M. Dye Your Color

- According to the problem, when Boki-chan casts a dyeing magic on the area numbered i, the areas in the same maximum connected set as i will also be affected by the dyeing magic.
- Therefore, we can preprocess all the maximum connected sets.
- Clearly, the size of a single maximum connected set will be logarithmic. Next, we can treat operation 1 as modifying the tree node values multiple times.
- We only need to consider how to dynamically maintain the maximum independent set of the tree, which can be done using dynamic programming (dp).

L. Greedy World

Problem Description

Initially, there are $k-1$ x's and 1 y. You can perform at most one operation, taking two numbers, replacing them with their average, and putting them back. The goal is to minimize the final value of all numbers.

- When x, y satisfy $0 \leq y - x \leq 2^{k-1} - 1$, the answer is x.
- When x, y satisfy $2^{k-1} \leq y - x \leq 2^k - 1$, the answer is $x + 1$.

L. Greedy World

Proof

Let the initial sequence be A, where $A_1 = y, A_2 = A_3 = \cdots = A_k = x$, and let the final answer be s_0.

Now, consider a sequence of length k, a, where $a_1 = y - x, a_2 = a_3 = \cdots = a_k = 0$, and let the final answer after performing the operation several times on this sequence be s_1. We have the conclusion: $s_0 = s_1 + x$.

L. Greedy World

Proof

Lemma

1. Lemma 1: For any positive integers a, b, c, we have $\lfloor \frac{(a+c)+(b+c)}{2} \rfloor = \lfloor \frac{a+b}{2} \rfloor + c$
2. Lemma 2: For any positive integer t, we have $\lfloor \frac{\lfloor \frac{t}{2} \rfloor}{2} \rfloor = \lfloor \frac{t}{4} \rfloor$

The lemmas are obvious.

L. Greedy World

Proof of $s_0 = s_1 + x$

First, we prove the conclusion: $s_0 = s_1 + x$.
Let x, y be arbitrary integers.
For a sequence of length 2, $A = \{y, x\}$ and $B = \{y - x, 0\}$, after performing one operation on sequence A, the final answer is $s_1 = \lfloor \frac{y+x}{2} \rfloor = \lfloor \frac{(y-x+x)+(0+x)}{2} \rfloor$, and by Lemma 1, we get: $s_1 = \lfloor \frac{y-x}{2} \rfloor + x$.
For sequence B, after one operation, the final answer is $s_2 = \lfloor \frac{y-x}{2} \rfloor$. By comparing the two, we see that $s_1 = s_2 + x$, and the conclusion holds.
We assume that for a sequence of length k, $A = \{y, x, x, \ldots, x\}$ and $B = \{y - x, 0, 0, \ldots, 0\}$, after several operations, the final answers for A and B are s_1 and s_2, respectively, and we have $s_1 = s_2 + x$.

L. Greedy World

Proof of $s_0 = s_1 + x$

Now, consider a sequence of length $k+1$, $A = \{y, x, x, \ldots, x\}$ and $B = \{y - x, 0, 0, \ldots, 0\}$, according to the assumption, for the first k elements of sequences A and B, we can perform several operations to transform both sequences into $\{s_1, s_1, s_1, \ldots, s_1, x\}$ and $\{s_2, s_2, s_2, \ldots, s_2, 0\}$, where $s_1 = s_2 + x$. We will find that for any operation on sequence A (let's assume that the current operation involves indices i, j, that is, transforming A_i and A_j into $\lfloor \frac{A_i + A_j}{2} \rfloor$), we can perform the same operation on sequence B (that is, transforming B_i and B_j into $\lfloor \frac{B_i + B_j}{2} \rfloor$). It can be observed that there will always be $\lfloor \frac{A_i + A_j}{2} \rfloor = \lfloor \frac{B_i + B_j}{2} \rfloor + x$. Therefore, the final answers for sequences A and B, S_1 and S_2, satisfy $S_1 = S_2 + x$.

Hence, the proof for $s_0 = s_1 + x$ is complete.

L. Greedy World

Proof

Returning to the original problem, we transform the problem to a sequence $a = \{y - x, 0, 0, \ldots, 0\}$, and let $t = y - x$.

Initially, we greedily perform operations between the smallest non-zero integer and any zero in the sequence (if exists), and it can be proven that this strategy will minimize the final answer.

L. Greedy World

Proof

Auxiliary Proof: Optimal Strategy

Before proceeding, we need to understand the role of 0 in the sequence. During any operation, let the current smallest positive integer be x_1, and if we choose any two positive integers to operate (assume we operate on a_i and a_j, where $i < j$), we get:

$$a_i = a_j = \left\lfloor \frac{a_i + a_j}{2} \right\rfloor$$

It can be observed that the new smallest positive integer x_2 satisfies $x_2 \geq x_1$.

L. Greedy World

Proof

Auxiliary Proof: Optimal Strategy

The reason is that now $a_i, a_j \geq x_1$, so:

$$\left\lfloor \frac{a_i + a_j}{2} \right\rfloor \geq \left\lfloor \frac{x_1 + x_1}{2} \right\rfloor = x_1$$

L. Greedy World

Proof

Auxiliary Proof: Optimal Strategy

However, if we choose $a_i = x_1$, $a_j = 0$, after the operation:

$$a_i = a_j = \left\lfloor \frac{x_1 + 0}{2} \right\rfloor < x_1$$

In other words, only when the current smallest positive integer is chosen to operate with 0, can the smallest positive integer in the sequence decrease.

L. Greedy World

Proof

Construction and Proof by Contradiction

Let the initial sequence be $a = \{t, 0, 0, \ldots, 0\}$, with k elements. If we perform m operations $(1 \leq m \leq k-1)$ according to the strategy, the sequence will look like:

$$a_1 = \left\lfloor \frac{t}{2} \right\rfloor, \quad a_2 = \left\lfloor \frac{t}{2^2} \right\rfloor, \quad \ldots, \quad a_m = a_{m+1} = \left\lfloor \frac{t}{2^m} \right\rfloor, \quad 0, 0, \ldots$$

L. Greedy World

Proof

Construction and Proof by Contradiction

We will use proof by contradiction:
If the $(m+1)$-th operation does not follow the strategy (i.e., we do not select a_{m+1}, a_{m+2}), but instead select a_{m-1} and a_{m+2}:

$$a_{m-1} = a_{m+2} = \left\lfloor \frac{\left\lfloor \frac{t}{2^{m-1}} \right\rfloor + 0}{2} \right\rfloor = \left\lfloor \frac{t}{2^m} \right\rfloor$$

L. Greedy World

Proof

Construction and Proof by Contradiction

At this point, we have:

$$a_{m-1} = a_m = a_{m+1} = a_{m+2}$$

That is, the current operation did not reduce the smallest positive integer.

L. Greedy World

Proof

Construction and Proof by Contradiction

If we still follow the original strategy and select a_{m+1} and a_{m+2}, we get:

$$a_{m+1} = a_{m+2} = \left\lfloor \frac{\left\lfloor \frac{t}{2^m} \right\rfloor + 0}{2} \right\rfloor = \left\lfloor \frac{t}{2^{m+1}} \right\rfloor$$

At this point:

$$2 \times \left\lfloor \frac{t}{2^{m+1}} \right\rfloor \leq \left\lfloor \frac{t}{2^m} \right\rfloor$$

That is, the smallest positive integer is reduced, and the original strategy is more optimal. If we select earlier positions, like a_k and a_{m+2}, the conclusion remains the same.

L. Greedy World

Proof

Construction and Proof by Contradiction

When there is no 0 in the sequence, the sequence should be of the form:

$$a_1 = \lfloor \frac{t}{2^{k-1}} \rfloor, a_2 = \lfloor \frac{t}{2^{k-1}} \rfloor, a_3 = \lfloor \frac{t}{2^{k-2}} \rfloor, \dots, a_{k-1} = \lfloor \frac{t}{2^2} \rfloor, a_k = \lfloor \frac{t}{2} \rfloor$$

According to the problem, we have: $0 \leq t \leq 2^k - 1$, we perform the following classification:

L. Greedy World

Proof

Case 1

Returning to the original problem, when t satisfies $0 \le t \le 2^{k-1}-1$:
Since $t \le 2^{k-1}-1 < 2^{k-1}$, we have $a_1 = a_2 = \lfloor \frac{t}{2^{k-1}} \rfloor = 0$ always.
Also, since $t < 2^{k-1}$, dividing both sides of the inequality by 2^{k-2} gives:

$$\lfloor \frac{t}{2^{k-2}} \rfloor < \lfloor \frac{2^{k-1}}{2^{k-2}} \rfloor = 2.$$

Since $\lfloor \frac{t}{2^{k-2}} \rfloor \in \mathbb{N}$, we have

$$\lfloor \frac{t}{2^{k-2}} \rfloor \le 1,$$

L. Greedy World

Proof

Case 1

Thus, $a_3 \leq 1$. From this, we can deduce that we can make $a_3 = 0$ with at most one operation.
After that, for $a_4 = \lfloor \frac{t}{2^{k-3}} \rfloor$, we can perform one operation to make:

$$a_3 = a_4 = \left\lfloor \frac{a_3 + a_4}{2} \right\rfloor = \left\lfloor \frac{0 + \lfloor \frac{t}{2^{k-3}} \rfloor}{2} \right\rfloor.$$

By Lemma 2, we get:

$$a_3 = a_4 = \lfloor \frac{t}{2^{k-2}} \rfloor \leq 1.$$

L. Greedy World

Proof

Case 1

We can then operate on $a_1 = 0$ and a_3, a_4 to make $a_3 = a_4 = 0$.
Continuing this process, we can eventually make all elements of the sequence a equal to 0.
From the above conclusion, the final answer for the original sequence A is x.

L. Greedy World

Proof

Case 2

When t satisfies $2^{k-1} \leq t \leq 2^k - 1$:
For expressions of the form $\lfloor \frac{t}{2^m} \rfloor$ (where $1 \leq m \leq k-1$), we have:

$$2^{k-m-1} \leq \left\lfloor \frac{t}{2^m} \right\rfloor \leq \left\lfloor \frac{2^k - 1}{2^m} \right\rfloor < 2^{k-m},$$

that is:

$$2^{k-m-1} \leq \left\lfloor \frac{t}{2^m} \right\rfloor \leq 2^{k-m} - 1.$$

L. Greedy World

Proof

Case 2

We consider two extreme cases:

Case 1: When all $\left\lfloor \frac{t}{2^m} \right\rfloor = 2^{k-m} - 1$, let the final answer of the current sequence be s_1, and construct a new sequence b, where $b_i = a_i + 1 = 2^{k-m}$. The final answer of b is s_2, and we have:

$$s_2 = s_1 + 1.$$

By construction, the sequence b looks like:

$$b_1 = 2, \quad b_2 = 2, \quad b_3 = 4, \quad \ldots, \quad b_k = 2^{k-1}.$$

L. Greedy World

Proof

Case 2

Operate on b_2, b_3:

$$b_2 = b_3 = \left\lfloor \frac{b_2 + b_3}{2} \right\rfloor = \left\lfloor \frac{2+4}{2} \right\rfloor = 3,$$

then perform operations on b_1 with b_2, b_3 to make $b_2 = b_3 = 2$. Continuing this process, we can make all elements of the sequence b equal to 2, i.e., $s_2 = 2$. Therefore, we have:

$$s_1 = 1, \quad s_0 = s_1 + x = x + 1.$$

L. Greedy World

Proof

Case 2

Case 2: When all $\left\lfloor \frac{t}{2^m} \right\rfloor = 2^{k-m-1}$, the sequence a looks like:

$$a_1 = 1, \quad a_2 = 1, \quad a_3 = 2, \quad \ldots, \quad a_k = 2^{k-2}.$$

Operate on a_2, a_3:

$$a_2 = a_3 = \left\lfloor \frac{1+2}{2} \right\rfloor = 1,$$

Continuing this process, we can eventually make all elements equal to 1, i.e., $s_1 = 1$, thus:

$$s_0 = s_1 + x = x + 1.$$

L. Greedy World

Proof

Case 2

General Case: Since the final answer for sequence a should be between these two extreme cases, and both cases yield $s_1 = 1$, we also have for the general case:

$$s_1 = 1, \quad s_0 = x + 1.$$

L. Greedy World

Proof

In conclusion, we have:

- When x, y satisfy $0 \leq y - x \leq 2^{k-1} - 1$, the answer is x.
- When x, y satisfy $2^{k-1} \leq y - x \leq 2^k - 1$, the answer is $x + 1$.