APPROACHES

1. Think DP/recursion.

2. Think greedy.

3. Think binary search.

4. Think sliding window/two pointer.

5. Think bit masking.

6. Think bit manipulation.

7. Think prefix.

8. Think difference array.

9. Think mapping.

10. Think multiset/ordered set.

11. Think gcd/lcm/modulo arithmetic.

12. Think in variables/expressions.

13. Think dfs/bfs.

14. Think bipartite.

15. Think in complement like all - answer.

16. Think segment tree.

17. Think brute force.

LEARNINGS

1. To find whether a number x divides all the numbers in a range l to r, x should be a factor of hcf of all the numbers in l to r, x can be the hcf itself.
2. To find the frequency of a element in a given range offline, store the indices in a map<ll, vector<ll>> for each element, do upperbound(r) – lowerbound(l) for map[a[i]]. If its online maybe try with segment tree or map<ll, multiset<ll>> but fear of TLE of MLE.
3. Can think of segment tree if associative property over a range.
4. Consider coordinate compression.
5. Max subarray sum can be solved using divide and conquer in nlogn.
6. When bit related problems, apply operations bit by bit and make truth table.
7. If always applying replace operation again and again with that value, see that it stablises or not.
8. <https://cses.fi/problemset/task/1734/>
9. When counting subarrays based problems, count the contribution of each index (contribution technique).
10. Consider truth table always when true false conditions.
11. Consider converting numbers to string if helpful.
12. Solve in reverse order in dsu if rollbacks needed using stack.
13. Offline queries + static data -> Mo’s Algorithm