Brief Analysis of World Finals Warmp-Up 2014

Problem A: Palinagram

Author Shahriar Manzoor
Special Thanks Md Mahbubul Hasan
Problem Type Adhoc

This is simple AdHoc problem. Count the number of appearance of each character in input. Suppose a character is called special if it appears odd number of time. If the number of special characters is less than two no extra characters are required to be appended. Otherwise print all the special characters except the last one. The proof is pretty intuitive.

Problem B: Meeting Point of Circles

Author Md Mahbubul Hasan
Special Thanks Derek Kisman, Towhidul Islam Talukder
Problem Type Geometry

This is a pretty straight forward geometry problem. Suppose we modify the problem as: given time t, after t time is it possible to move all the circles so that they go through a common point? If we can solve this problem then we can binary search over the time to get minimum time.

Now the question is where can a circle go or better to ask, where can the circumference of a circle be after t time? If you think a bit, it is some kind of ring. The outer radius of the ring is $t + r$ and inner radius is $min(r - t, 0)$. Now the problem is, is there any point in the plane that lies inside all the rings? You can solve it by line sweep if you want but here we can solve it more easily. Draw some pictures and look at the common region of the rings. In most cases there will be a corner in the border of common region. By corner we mean intersection of two boundary circles of rings. So what we can do is, take all the pairs of circles and find their intersection point. Check if any of them are inside/on the boundary of the rings. Now, there can be a special case. When the rings are nested inside another then no intersection point is on the boundary. But in that case a whole circumference of a circle will be inside all of the circles. To simplify we take a point from all the circles (say leftmost point of the circles) and check with them as well. Apart from binary search the complexity is: $O(n^3)$.

Problem C: Dynamic Accessible Pairs

Author Lijie Chen
Translated with minor modification Rujia Liu
Special Thanks Haoqiang Fan, Yifan Zhou, Md Mahbubul Hasan
Source Chinese Olympiad in Informatics WinterCamp 2014
Problem Type Data Structure

This is a very tough problem. Instead of writing the analysis I would like to paste the conversation mail between alternate writer (Md Mahbubul Hasan) and author (Rujia Liu):

So I first tried to relax the problem. I think I can solve for offline version.

1. *Binarize + centroid decompose the tree. As a result for every node, one of the two children has at least 1/3 of the nodes in subtree. Also the edges will be weighted 0/1 so that the distance remains correct.*
2. *Now bottom up style cost merging.*

Its order is $O(nlog^2n)$ *if i am not wrong. Now, to make it online, i think the entire structure is to be maintained in balanced tree making* $O(nlog^3n)$ *if i am not wrong*

This is indeed one of the correct solutions. Another solution pointed out by Rujia Liu is similar to one of the solutions of "GRE Words Revenge" problem from Chengdu 2013 (UVa 1676). After every K steps you have to rebuild the offline structure. For in-between queries you have to do something normal a bit costly queries. Find out optimal value of K.

Problem D: Gauss Reborn

Author Md Mahbubul Hasan
Special Thanks Tanaeem M Moosa, Hasnain Heickal
Problem Type Number Theory, Inclusion-Exclusion

The interesting fact is, if (a, b), (d, c) and (b, d) are co-primes then the numerator and the denominator of the sum of the fractions would be co-prime too. (The reverse is also true). [You can prove by contradiction]. So first for each d find count and sum of c such that (c, d) are co-prime. Then sum/count these values for all the numbers that are co-prime with d. And so on. You need to apply inclusion-exclusion while doing these count/sum.

Problem E: Game of Throne Season 2

Author Shiplu Hawlader
Special Thanks Md Mahbubul Hasan
Problem Type Flow, Shortest Path

Of course we will do Binary Search over the answer. Suppose our guess is not d. We construct a flow graph with vertices like: (v, d) which means it denotes a vertex for dth day of node v. Run a max-flow and check if the flow is less than K or not.

Problem F: Gift Dilemma

Author Hasnain Heickal
Special Thanks Muhammed Hedayet Islam
Problem Type Math, Number Theory, Extended GCD

Obviously we are to solve: $Ax + By + Cz = P$ in positive integer (or non-negative or whatever). Now there is a weird limit in input section: $C/gcd(A, B, C) \geq 200$. Let us first divide the equation by: $gcd(A, B, C)$ then say the equation will become: $A'x + B'y + C'z = P'$ where $C' \geq 200$. Since $P \leq 10^8$ then $z \leq 10^8/200 = 5 * 10^7$. So for each valid values of z we get equation like: $A'x + B'y = Q$ and we are interested in its positive integer solution. And this is common application of extended gcd algorithm.

Problem G: Query for Divisor-free Numbers

Author Tasnim Imran Sunny
Special Thanks Md Mahbubul Hasan
Problem Type Math, Adhoc, Data-Structure

First, for each position i find maximum $left_i$ such that, $a[left_i]$ divides a_i and $left_i < i$. Similarly find $right_i$. (You can find them by binary searching on the list of indices for each of the divisors) Now the rest is line sweep over the array. We will process from left to right. If we encounter a i we increase 1 at i and decrease 1 at $left_i$. When we encounter a $right_i$ we cancel those increments or decrements. If when at i there is query with (L, i) we just take the segment sum of that range. The proof is left for you :)

Problem H: Palindromic Sums

Author Tasnim Imran Sunny
Special Thanks Hasnain Heickal, Kazi Rakibul Hossain
Problem Type Pattern, Adhoc

This is all about finding pattern of the numbers. Here only one type of pattern will be shown, you have to calculate how many such numbers you can make or in case, you may need to find some additional patterns. One possible pattern can be like: $(12 \mid 23 \mid 34 \mid 45)(null \mid 0 \mid 1 \mid 2 \mid 3 \mid 4)(21 \mid 32 \mid 43 \mid 54)$ and to get the other set swap the first and last pattern. Dont want to spoil the fun. Find out the appropriate pattern yourselves :)

Problem I: Minimum Sum

Author Kazi Rakibul Hossain
Special Thanks Muhammed Hedayet Islam, Jane Alam Jan
Problem Type Adhoc, Data-Structure

Consider the position i in the array of a. Let $left_i$ be the index such that it lies at left of i and that is the largest possible index such that $a[left_i] < a[i]$. Similarly define $right_i$ to be the largest index such that $right_i > i$ and $a[right_i] < a[i]$. Then $a[i]$ is minimum for any subsequence that has its left end at: $[left_i, i]$ and right end at $[i, right_i]$. You can find out $left_i$ or $right_i$ using Binary Indexed Tree or Segment Tree or Stack. However, there is subtle flaw in this solution. This solution will not work if a value appears multiple time in the array. However, you can deal with such case very easily. Try to consider $a[left_i] \leq a[i]$ and see if it works (or may be a bit more fix up is required).

Problem J: The Largest Circle

Author Shahriar Manzoor
Special Thanks Muhammed Hedayet Islam, Hasnain Heickal
Problem Type Geometry

It is obvious that the center of the circle will be at the intersection of the diagonals. The reason is that point is the center of symmetry of the parallelogram. Now find the distance to each of the sides from the center. The main difficulty of the problem is, you have to calculate everything in integer. Use two variables say p and q to denote fraction $\frac{p}{q}$. Now find the values of p and q.

Problem K: Kiano The Clause!

Author Muhammed Hedayet Islam, Md Mahbubul Hasan
Special Thanks Shiplu Hawlader
Problem Type Bitmask-DP

Suppose the pattern matrix is P and the matrix we are to build is T. You have to build T in such a way that if you place P at any place then, there must be a cell of P which contains 1 but the corresponding cell at T is 0. Since there are M rows in P a row of T can be covered by at most M different P. ith row of T can be covered by P that starts from i, $i-1 \ldots i-M+1$th rows in T. We need to make sure that none of the sub-matrices in T are covered by P. So just come from top to bottom and keep the status of last M places for starting of P in a bitmask.