ACM ICPC Dhaka Site 2013

Problem Analysis

Md Mahbubul Hasan

Software Engineer
Google Zurich

Switzerland, 2013

Outline

Problem A: Falling Ants

Author:	Shahriar Manzoor
Special Thanks:	Monirul Hasan, Md Mahbubul Hasan
Type:	Adhoc, Giveaway

Solution

The main problem is to reading the problem with patience :) The required formula is given at the end of the statement. You just need to find out the least H value and in case of tie higher volume. No precision trick, no long long trick.

Fun Fact

Author made this problem from the picture, not the other way round :)

Problem B: Game of MJ

Author:	Hanain Heickal
Special Thanks:	Shiplu Hawlader
Type:	Game Theorey

Solution

We can consider ith position of B base number as B number of $B^i mod 3$ (0 indexed number). So in this way we get how many 0s, 1s and 2s are there. Now, a player can not reach $0 mod 3$ so it is obvious that if current mod value is 1 then we must pick another 1, if it is 2 then we must pick 2 also we can pick 0 at any time. So finally it is some case analysis depending on the parity of the counts of $0, 1$ and 2. If you are not careful with covering all the cases you are going to get **Wrong Answer**!

Problem C: Game of Throne

Author:	Shiplu Hawlader
Special Thanks:	Rujia Liu (Online version), Md Mahbubul Hasan
Type:	Dynamic Programming (Onsite version), Graph Theory

Solution

The problem reduces to matching problem. The city in A should have odd degrees and cities of B should have even degrees, that means, there is a path from every city in A that goes through any number of cities and ends up at another A. So it is some what like matching from vertices A to vertices of A. Every node of A should be part of the matching. So the solution is to find all possible shortest paths among the vertices of A and then match pairs of nodes in minimum cost. Since $K \leq 21$ we can do bitmask DP here. Note that, for odd K there is no solution.

Fun Fact

Go and try out the harder online version! Judge solution is only 1248 line!

Problem D: Pattern Locker

Author:	Monirul Hasan
Special Thanks:	Shahriar Manzoor, Md Mahbubul Hasan
Type:	Combinatorics, Formula

Solution

It is just sum of permutation of k numbers from n numbers for some consecutive values of k. We know, $nPk = n \times (n-1) \times \ldots (n-k+1)$. Probably you will get Time Limit Exceeded if you are trying to evaluate all the terms individually. Just try to get nPk from $nP(k-1)$ thats the trick.

Problem E: Pearl Chains

Author:	Abdullah Al Mahmud
Special Thanks:	Tasnim Imran Sunny
Type:	Combinatorics, Formula, Stopper

Solution

It is a special number something like *Stirling number* or *Euler number*. Go and find out the formula for it, implement it and curse the setter!

Problem F: Two Points Revisited

Author:	Md Mahbubul Hasan
Special Thanks:	Mir Wasi Ahmed, Hasnain Heickal
Type:	Simple Geometry

Solution

Only one line solution but I hope some people banged their heads off! The solution is consider the minimum bounding square and then just rotate it 90 degree in any side! That's it! The code should be something like: scanf... print... Thats it, no intermediate line.

Fun Fact

This problem was supposed to be given in Preliminary but considering the rush on server you can put with this problem we removed it. I hope you were not able to keep down onsite server :)

Problem G: Watching Kangaroo

Author:	Md Mahbubul Hasan
Special Thanks:	Towhidul Islam Talukder, Shiplu Hawlader, Hasnain Heick
Type:	Data Structure, Greedy

Solution 1

The problem has many different approach. One approach can be using two BITs. Scan from left to right, when you cross a left end of segment push it in BIT1, when crossing middle remove from BIT1 and insert into BIT2 and finally after passing right end remove from BIT2. Now try to figure out what can be query at the query points.

Problem G: Watching Kangaroo

Solution 2

Second approach is much simpler but requires some insight. First remove those segments that are completely inside another segment. It can be done by sorting and sweeping. Then again scan from left to right. Maintain an **answer segment pointer** at each query node ask if the current **answer segment** is good or the next one, if the next one then update this pointer. Also take care if the query point went outside the answer segment.

Problem H: GCD XOR

Author:	Tasnim Imran Sunny
Special Thanks:	Shiplu Hawlader, Hasnain Heickal
Type:	Pattern, Math, Number Theory

Solution

Generate some pairs for small N and try to figure out the pattern.

Problem I: Fiasco

Author:	Mir Wasi Ahmed
Special Thanks:	Derek Kisman, Md Mahbubul Hasan
Type:	Graph Theory

Solution

There can be several approaches to this problem as well. The easiest approach may be if you remember that, BFS is correct in a weighted graph if the edge weights are increasing from root. So find out the BFS tree of the given graph then assign the weights increasingly from root to leaves. That is first put smaller weights on the edges between level 0 and level 1, then level 1 and level 2 and so on.

Problem J: Dromicpalin Substring

Author:	Sohel Hafiz
Special Thanks:	Md Mahbubul Hasan
Type:	Simple string

Solution

You can consider it like sliding window. You will refresh your parity array for each update of left end. Left end should be updated once the right end reaches end. At each step you will update right end! (So complex!!) Now chance the parity of the right end character. If at time, you find no odd character or only one odd character then you found a dromicpalin. For the count of odd characters keep a counter variable which is to be updated after each parity change.

Problem K: Fill the Cuboid

Author:	Rujia Liu
Special Thanks:	Md Mahbubul Hasan
Type:	Backtrack

Solution

Start trying with a larger cube. To check whether we can put a cube of size s at (x, y, z) we can keep fill/non-fill information in 3D array. However, to make this check faster we can keep this information in 2D array as bitmask! That is at [x][y] we keep a bitmask number like: 010010 to denote which cells in z-axis at (x, y) are filled up. We can query if there is any blocked cell in consecutive s cells in this column with just one bitwise operation.

Fun Fact

S. Manzoor: Rujia told me to use this problem in Dhaka **Preliminary** if I wish, since this is one of the **easiest** problem he has created!

Thank you and Congratulation to the Winning team!