2019 ECNU Campus Invitational Contest
Solution Sketches

Fangxin Lin, Xuliang Zhu, Yiran Zhou, Yuge Zhang

April 27, 2019

Judges' Anticipation

- Very easy: A H
- Easy: F L
- Medium easy: B G I J
- Medium hard: C K
- Hard: D E

Problem A

- First solved: Jiadong Xie, 00:03 (+).
- Enumerate all the answers and check,
- or write a bunch of `if`'s.

Problem H

- First solved: Yanru Guan, 00:15 (+1).
- The total minutes of huashui is the sum of an arithmetic sequence.
- Need to write a gcd (or use gcd provided in your language).

Problem F

- First solved: Jiadong Xie, 00:10 (+).
- Convert everything into abbreviations.
- For each distinct abbreviations, if there are n of them available, add $\binom{n}{2}$ to answer.
- Use sorting or STL map for implementation.

Problem L

- First solved: Zhiqi Zhang, 00:52 (+).
- Greedy approach.
- Eliminate all the 6, match 3 in pairs, 2 in triples, 1 in sextuples.
- There are at most one 3, two 2, and five 1 left.
- Match 1, 2 with 4 and 5.
- Do brute force for the rest of 1, 2 and 3, or notice that 1, 2, 3 can be easily fit in 6, the answer can be calculated with a simple division.

Problem B

- First solved: Zhiqi Zhang, 00:24 (+).
- Let's assume that n pairs are separated in the hands of two players, and the first player win with probability $f(n)$ if he has $n+1$ card (with old maid), and probability $g(n)$ if the old maid is in the hand of his opponent.
- It's intuitive that the answer converges. (think of $g(1)$)
- $f(0) = 0$, $g(0) = 1$; $f(n) = 1 - g(n)$.
- $g(n) = \frac{n}{n+1}(1 - f(n-1)) + \frac{1}{n+1}(1 - g(n))$, thus $g(n) = \frac{\lfloor \frac{n}{2} \rfloor + 2}{n+2}$
- You can also try some small n's and discover the patterns.

Problem G

- First solved: Jiadong Xie, 02:25 (+).
- Easy to see that all we need is to solve $n = 5$.
- Try some brute force or construction.

Problem I

- First solved: Yicheng Gu, 00:59 (+).
- It seems that setting -1 as ∞ and running FloydWarshall will work.
- Check everything:
  - 0 on diagonal.
  - Symmetric properties.
  - ∞ should be set not too large, and not too small.
  - Whether Floyd changes existing edges.
- Proof is not trivial, so have fun with that.

Problem J

- First solved: Jiadong Xie, 03:56 (+1).
- For $k \leq 8$, make the last one count $(k-1)9999999$.
- For $k = 9$, try 8888888888.
- Do a easy math to find the initial counting (as this will be the first round).
- After the survivor, everyone will be executed immediately.

Problem C

- First solved: Xiaoran Chen, 03:51 (+3).
- Obviously, no prime larger than $n/2$ is useful, and at most 2 primes larger than $n/3$ can be added. The answer is clear; the rest of the problem is the construction part.
- You can simply do it like this:
  $P_0\ 2P_0\ 6\ 2p_0\ 4p_0\ 5p_0\ \ \ldots\ 3p_0\ 3p_1\ 4p_1\ 5p_1\ldots\ 2p_1\ \ \ldots 12\ 2P_1\ P_1$
- You can solve the cases where $n \leq 12$ with another construction or some brute force.

Problem K

- The first observation will be: between $1999\cdots9$ and $2222\cdots2$, the answer is 3. So subtract $222\cdots3$ to convert it to regular 2 case.
- Greedily construct the solution: note that carry digits are almost always helpful as it provides more choices.
- Cases with highest digit 2 might require different approaches and more insights.
- The problem is not designed to be easily-implemented, but it's possible that there exist other neat solutions.

Problem D

- First solved by Junchi Zuo, 03:30 (+).
- Think of the problem that does not require “distinct”.
  - Enumerate the split point,
  - do a dynamic programming to count the number of common subsequences.
- Think of the problem that counts the distinct number of subsequences of a sequence (no extra constraints).
  - Do standard dynamic programming (calculate f[i] as the sum of a bunch of f[j]);
  - make a_j the last appearance of a_j before a_i.
- This problem is a combination of the two problems. Left as an exercise.
- $O(n^4)$ may not be sufficient enough to pass. Optimization to $O(n^3)$ might require some insights on what is changing and what is not (no D.S. needed).

Problem E

- First solved: Dongyu Liu, 01:37 (+4).
- Nasty implementation and painful debugging.
- The most boring task in this problemset.
- This is the problem where judges have overestimated the difficulty. Actually, contestants shows remarkable abilities to deal with long and tedious statements.