Problem I: Secret Chamber at Mount Rushmore

Shortest judge solution: 405 bytes. Shortest team solution (during contest): 498 bytes.
Python solutions by the judges: both Pypy and CPython

This was one of the two easiest problems in the set. The set of translations forms a directed graph on the 26 letters of the alphabet. Given two query words $s_1 s_2 s_3 \ldots s_L$ and $t_1 t_2 t_3 \ldots t_L$ of equal length L (if the lengths are different, the answer is clearly `no` and we just answer that), we need to check whether for each $1 \le i \le L$, there is a path from s_i to t_i in the graph of letter translations.

One natural way of doing this is to precompute the transitive closure of the graph using the Floyd-Warshall algorithm, which allows you to check each pair (s_i, t_i) in constant time. But the bounds are quite small, so you can use pretty much any polynomial time algorithm for this (e.g. doing a DFS for every pair (s_i, t_i)).

Problem J: Son of Pipe Stream

Shortest judge solution: 1649 bytes. Shortest team solution (during contest): 5492 bytes.
Python solutions by the judges: only Pypy

Even though the problem does its best to avoid using the word "flow", it should be pretty clear that this is in fact a maximum flow problem, though it has a few twists to resolve. First, the viscosity parameter v is a red herring that is pretty much irrelevant to the problem and we will ignore it here.

A first small observation is that the desired flow will be *a* maximum flow from the water and Flubber sources to the destination. Let Z be that total maximum flow. Assume for the moment that it was possible to distribute this arbitrarily between Flubber and water, so that it was possible to route any amount F of Flubber between 0 and Z and $W = Z - F$ water. Then a bit of calculus (left as an exercise) shows that the maximum flow would pick $F = a \cdot Z$.

There are two different reasons why the assumption made above might not hold. One is a "trivial" one: the maximum amount of Flubber routable from Flubber source to destination might be less than $a \cdot Z$, or the amount of water routable might be less than $(1 - a) \cdot Z$. If this is the case, we should simply set the desired amount of Flubber F to value nearest to $a \cdot Z$ in the interval $[Z - W_{\max}, F_{\max}]$. Let us refer to the resulting potentially optimal values of F and W as F^* and $W^* = Z - F^*$.

The second reason is a bit more subtle – it is not clear whether it is the case that we can simultaneously achieve Flubber F^* and water W^*. Let $\vec{f_1}$ be a flow which routes $F_{\max}$ Flubber and $Z - F_{\max}$ water, and let $\vec{f_2}$ be a flow which routes $Z - W_{\max}$ Flubber and $W_{\max}$ water. Then for $\alpha \in [0, 1]$, $\alpha \vec{f_1} + (1 - \alpha)\vec{f_2}$ is a flow of fluids in which $\alpha F_{\max} + (1 - \alpha)(Z - W_{\max})$ of the fluid originates at the Flubber source, and we can set α appropriately to a constant so that this equals F^*. However, there is a snag with this: there might be pipes where $\vec{f_1}$ and $\vec{f_2}$ route fluids in opposite directions, so it is not clear that we can achieve this flow while satisfying the "water and Flubber must not go in opposite directions" constraint. Phrased a bit more abstractly, the "no opposing flows" constraint is not a convex constraint.