Tea Party

Elsa Browning, Kyle Hakala, Dan Woeste

December 11, 2016

The Problem

We are having a party with amounts of tea and candy. Return the int outcome of the party encoded as 0=bad, 1=good, or 2=great. A party is good (1) if both tea and candy are at least 5. However, if either tea or candy is at least double the amount of the other one, the party is great (2). However, in all cases, if either tea or candy is less than 5, the party is always bad (0).

Taken from CodingBat: http://codingbat.com/prob/p177181

Setup

We allowed our problem to use the boolean and integer stacks. We explicitly provided the exec-if atom, as well as the items in1, in2, 5, and 2 for trials one, two, and three.

For test cases (re, the input-set in our problem) for the first two trials, we used all of the listed cases from the original CodingBat problem. For our third and fourth trials, we added a collection of our own randomly generated input cases. These inputs were psuedo-randomly mixed up to avoid clumping/clustering (a.k.a., clumpterings). However, in our third trial, we added a case that had 1005 as an input which seemed to cause stability issues. So we changed this test case for our fourth trial.

The expected-output funciton was first written in Java and then translated to Clojure. All test cases returned the expected output using the expected-output function.

Results

Our first trial finished rather fast over only 53 generations. However, we did not save all of this data; we ran this on the command line and realized that we should probably output to a text file after the fact. So we only saved the last generation and solution.

With speed of such apparent grace and vigor, we attempted to run a second trial which took a wonderous 470 generations. We thought that the uptick in generations may have been attributed to being "lucky" in the first trial.

Our third attempt at a trial did not finish before 750 generations, so we cancelled this and reevaluated our lives. We did not save this information. It was after this trial that we changed our test case with the 1005 input hoping to find a solution.

Our final trial completed after 522 generations. We believe the increase is due to removing the extra guidelines, 5 and 2.

We tried to parse through the clojush instruction stack, but the resulting headache was too much for our crew to bear. So we hope that the resulting solutions are actually sound.

Fin