Clare Pinder

Can apply to jobs

April 30, 2019 3:30pm

General feedback

This was a really excellent review. There wasn't one moment where I felt you were off-process: as a result, you were focussed, clear, and intentional. You identified the core problem rapidly, modelled it well, and designed and built a powerful but self-contained algorithm to solve the problem. Your debugging approaches seemed on point and your code was well-factored and thoughtful. It's hard to point out areas to improve within the context of Makers, but I have a hunch you might be interested in some aspects of functional programming – Haskell or Clojure being interesting routes into that.

I can TDD Anything Strong

Your first test was a simple smoke test to check your files were set up correctly. You then removed the test as it was unnecessary to your further process, which was really great to see. Your next test split the problem up intuitively - you chose the case where all the numbers were already divisible by the given step, which was an inspired way to simplify the problem. This allowed you to slime the solution while also delivering value rapidly.

I can program fluently Strong

You were moving rapidly around the command-line, getting RSpec set up. You seemed to have no problem with getting the RSpec test and basic OO structure in place. You selected a map to iterate over the array, which was a good choice and shows good understanding of more advanced array manipulations. You started work on the algorithm design using some research, then jumped to Pry to experiment with simplified versions of the problem, until you came to an algorithm that really worked: dividing, rounding, then multiplying.

I can debug anything Steady

You used printed output from the program to move from the REPL into the program environment, using this approach to drive your algorithm design. You were trying simple, logical approaches that validated hypotheses as you progressed.

I can model anything Steady

You came up with a great lightweight model of soundwaves as arrays of frequencies, a single class with a step filter method, sketched out on paper.

I can refactor anything Steady

When writing your map function, I noticed that you first "refactored" your hard code to the map. This was a really good way of ensuring that your small-step process was working well. Before committing, you removed commented code.

I have a methodical approach to solving problems Strong

You started by thinking really clearly from the user and client perspectives, translating their needs into a set of program requirements. You then jumped into testing. When designing the algorithm, you first researched, then played with bits of the problem in the REPL, then wrote a clean, simple solution which solved the problem.

I use an Agile product development process Strong

You asked some really great questions about what the user interactions would be, what edge cases might exist, and so on. This was a great set of high-level user questions which clarified all the important aspects of the program you wanted to create. You asked other great questions about the kinds of sound wave limits and the default step that should be provided.

I write code that is easy to change Steady

We didn't assess this during the review, but the portfolio is contains credible evidence for this goal.

I can justify the way I work Strong

You were extremely clear with your thought process, conveying a strong sense of intentionality.

I grow collaboratively Steady

We didn't assess this during the review, but the portfolio is contains credible evidence for this goal.

I manage my own wellbeing Steady

We didn't assess this during the review, but the portfolio is contains credible evidence for this goal.

I can learn anything by myself Steady

We didn't assess this during the review, but the portfolio is contains credible evidence for this goal.