Hey there,

Thanks for applying for a position at Strathcom. By now, you've had a conversation with me and understand a bit more about what we're looking for. The next step is for my team to start getting involved. They'll do that by getting a better sense of how you approach coding.

Attached is a coding problem. Homework, if you prefer. The goal is for you to take it away this weekend and spend a few hours building a solution to the problem. Please send your results back to me Monday morning. A public github/gitlab/bitbucket repo is an easy way to do that but I'll take a zip file if that's more convenient.

The coding problem was written by the entire development team to represent what they want to test in a candidate. Because the homework assignment will take you a few hours to perform, and takes each member of our team an hour or so to assess, we only give the assignment to candidates who we think have the opportunity to do well on it. We don't want to waste anyone's time, ours included.

After you submit your solution, the dev team will review it on our side. In the review, we're looking for::

- readability and maintainability,
- basic style standards (i.e. linting),
- how you name classes, methods, variables,
- how you structure and design your code.

Once we've reviewed your code, we'll make the decision to proceed to in-person interviews. That includes a technical interview that will be mostly based on the coding problem. Rather than getting you up at a whiteboard coding algorithms that you haven't used for years, we'll talk about your coding problem during that interview. You might still choose to use a whiteboard to walk us through some of your code, but it won't be a trivia test. We want to understand why you wrote your code the way you did. We've got opinions but we're more interested in yours.

Good luck and have fun,
Mike.

P.S. Here's a few tips:

- You might get stuck at certain points, needing data or decisions to move forward. In those cases, just call a magic function or class that encapsulates your assumptions. Don't get hung up on small things. Figure out how to move forward.
- Think of this like math class in high school. It's good if your solution actually works, but you'll get part marks for showing your work. Don't fret if your code doesn't compile or function. It sucks but this is all about showing ***how*** you code rather than delivering a polished product.

- Cheating is OK! Use libraries that help you, consult Google and Stack Overflow, maybe even phone a friend. Using the tools that are available to us makes us move faster. Take advantage of them. **But** if you're copy/pasting code, make sure you understand it and can explain it. Once you paste it, it's your code - own it.