# Tech Interviews are still broken

I was at a friend's place on the 4th of July and they asked if I wanted to watch them complete a coding challenge for a prestigious AI bootcamp. Surprised by their definition of "hanging out", I nonetheless said yes. I was curious for a few reasons:

1. It had been a while since I had even looked at a coding challenge. With all the new AI coding agents available, I was curious if the style of coding challenges had changed accordingly.
2. I wanted to see how my friends went about solving coding challenges, because I was terrible at them.

I asked my friend if my presence would be distracting for the test. They said it would, but it didn't matter. The test was going to be on "general python skills", so it probably wouldn't be particularly difficult.

Anyway, they jumped onto their computer, pulled up a seat for me and tippy-tapped and clicky-clacked to the email containing the test link.

A 1-hour timer started off in the top right corner and 5 coding challenges popped up on screen.

They were, in order:

1. <a LeetCode Easy problem>
2. <a LeetCode Medium problem>
3. <a LeetCode Medium problem>
4. <a LeetCode Medium problem>
5. <a LeetCode Hard problem>

My friend was off to the races. They breezed through the first problem, verbally exploring potential solutions, fingers zipping across the keyboard and generally making great time. They started off on the second... and that's when it all went to shit.

They struggled hard with the remaining problems. They found solutions that didn't cover every test case. They scribbled on their notepad. They attempted to divine a Dynamic Programming solution. With only 15 minutes left, my friend decided to ask an AI chatbot for help.

When the timer finally expired, we vented about how terrible that experience was.

---

Later on, a few thoughts (and intense feelings) bubbled to the surface, and I decided to write them down:

1. Success in interviews continues to be defined as: "Grind as many of these coding problems as you can until you start seeing the patterns in your sleep." It often feels like you either know the optimal solution in advance, or you'll never figure it out by yourself.
2. These coding challenges still have nothing to do with the actual bootcamp/job that the applicant is applying for.
3. Experience in the industry is irrelevant to doing well in these interviews. Friends of mine who left their jobs or were let go have said they needed to "get back into LeetCode"; the implication being that our jobs did not use any skills that we learnt for interviews.

4. AI can solve these coding challenges with graceful ease. But instead of embracing AI and changing these pointless tests, the gatekeepers have doubled down on their strategy: keeping the same tests, but trying to detect AI-assisted cheating.
5. I'm grateful and lucky to still have a well-paying job in this industry, cause I'd be in big trouble if I had to grind LeetCode to get in anywhere. I don't do well in these kinds of tests.

---

I think that if there were ever a time for algorithm-heavy tech interviews to end, it is now, in the age of AI and for a few reasons:

## Most software engineers don't solve these kinds of problems

We don't.

We really don't.

At least I don't and the people I know don't.

Yes, I'm generalizing. If I spend more time, I can collect data and make a better case. For now, this is just my opinion.

I believe that as engineers, we're asked to solve all kinds of problems, not just algorithmic ones. In fact, we often have to solve problems along constraints where an O(log n) algorithm does nothing for us. For example, if your boss wants your feature to land by tomorrow morning, a 10 line brute-force algorithm can often be a decent tradeoff to make that happen.

## AI coding tools should be allowed in interviews

Using AI coding tools is an acceptable thing to do now as a software engineer. And while I'm not a fan of how carelessly AI is used by people, I can't deny that it is adept at solving these kinds of coding problems. These problems usually fit an existing pattern, making it easy for AI to connect the dots.

The tech industry as a whole is moving towards a model where we exporting some of our thinking to AI using tools like Claude Code and Gemini CLI. So why would you restrict the use of these tools in the interview?

Assuming that we've all studied the principles and done the hard work "without calculators" at our various universities and colleges, a job interview seems like the point at which you should be allowed to use every advantage you have at your disposal?

## AI tools can make better interview questions

Since there are so many people vying for the top jobs and so few people to screen them properly, an easy way out is to make a really hard test that very few applicants can pass.

I think AI can do a better job here!

If you recall, the goal of my friend's bootcamp test was to determine "general python ability". AI chatbots can actually be customized to create and host such interviews, with minimal supervision.

This idea is fairly easy to prove. On a whim, I asked Gemini to “Design a technical interview to test general python skills”. I told it that the interview will be used to determine if the candidate will be hired for a SWE position.

Gemini did a fantastic job of designing a test with that intention in mind! It had a conversational section where I’d ask questions about python concepts and libraries. It had a section for a practical coding challenge. It even had a part where I’d ask the candidate to explain their code and do an informal code review on it together.

I like that! It’s not perfect obviously, but it’s much closer to an ideal interview, if you ask me.

## False-positives are common in this system

There’s plenty of engineers who can grind LeetCode and pass the technical interviews, but not all of them are good engineers to work with.

They can be the absolute best at algorithmic problem solving, but because the tests put less focus on the other skills required of our job (documentation, testing, code health, foresight, initiative, independence, scheduling, prioritization, socialization), those aspects go untested.

The engineers who are bad at those skills create havoc in the teams they join.

## Even without AI, there are better options

Thankfully, a few companies are already finding good alternatives to this interview process, but those approaches have yet to gain traction.

One alternative is to do a “work trial”, where you start working for the company for about a week or so and at the end of that week, the engineers you collaborated with give feedback about whether you should be hired or not.

Another approach is to simply give more practical problems in the interview that are closer to the work that the engineer would actually be doing.

A common criticism of the alternatives is the amount of time that now needs to be invested in the interview process, which matters when there’s thousands of applicants; a problem made worse by the barrage of recent layoffs at big tech companies.

---

I hope we can move to an interview model that is better than what most tech companies use today. I’m personally invested in such a change, because it would mean that whenever I want to change my job, I don’t have to mindlessly practice a bunch of algorithmic coding questions that I will never use once I pass the test.