[Technical interviews](https://www.themuse.com/advice/how-to-ace-your-technical-interview) are known for their puzzles. Whether you’re asked to write a binary tree out on a whiteboard, explain a sorting algorithm, or implement FizzBuzz in 10 lines or less, you should prepare for technical programming questions in the average interview.

But that’s not the only thing you should be prepared for.

The reality is that many developers will be able to solve common programming interview questions with little variance between answers. So, what makes you different from the next candidate? That’s exactly what we’ll be focusing on in these five tips.

**1. Talk Out Your Reasoning and Problem Solving Process**

The worst thing you can do when asked [a tough question](https://www.themuse.com/advice/4-ways-to-handle-interview-questions-you-dont-know-how-to-answer) is to go totally speechless. Unfortunately, it’s very easy to do this on accident. When you encounter a problem that requires significant mental energy and focus, it’s likely that your first inclination is to retreat into your mind. While this is natural, it is also not very helpful for the interviewer.

The point of technical questions, in part, is to uncover how you think about solving problems. What is your process? How do you break down the different components? How do you arrive at a solution? How do you react when trying something that doesn’t work? Are you better at experimenting in code or at sketching something out on a whiteboard?

Explain what’s happening in your head as you solve the problem. Act as if you are recording your voice to publish online to teach others how to solve that problem. Even if your interviewers are giving you the space to think quietly, you may benefit from explaining your thought process without them prompting you to do so. Not only does this help them understand your skills and critical thinking more thoroughly, but it also makes you more memorable.

**2. What’s Better Than Solving a Problem in a Technical Interview? Solving it Twice**

Very few problems have only one solution, and all problems have infinite incorrect solutions. So if you focus on only one way to solve a problem, you’re missing a major opportunity to prove your flexibility and skill set.

Instead of simply going with a well known solution or working in a single language, open the discussion up about that particular problem and solve it for different scenarios. For example, if you are asked to program FizzBuzz, you might offer to do so in two languages, or by employing two different paradigms, or perhaps by taking some performance constraints for one solution and aesthetic constraints for another.

By validating that the problem may have multiple solutions, you’re showing your adaptability, flexibility, and awareness, all of which will instill trust in your interviewers that you will be able to choose the right solution among many possibilities.

Of course, don’t go overboard—there is an art to reading when answering a question with multiple solutions is overkill or happily welcomed. When in doubt, don’t be afraid to ask your interviewers if they mind if you take some time to expand on your solution with a secondary option.

**3. Don’t Be Afraid to Share Your Opinions, When Applicable**

Sometimes, as you work through problems, you will make decisions that are entirely based on your own taste and opinion. And that’s OK—employers are interested in your opinion! The way you think and react to situations makes a big difference to the culture of a company. Having an opinion is also a sign of leadership and technical maturity as a developer. To have a discussion about your opinions requires that you’ve evaluated other positions on a given subject.

Remember, however, that opinions can be held very closely. If you disagree with your interviewer on a given subject, tread lightly when sharing that information. While it’s good to have a point of view, it’s also important to note that sharing it is not always necessary and know how to choose your battles. A good rule of thumb: Don’t share your opinion unless you are asked.

## 4. Never End an Answer With “I Don’t Know”

Unless you are facing a “Kobayashi Maru” scenario, never end an interview question with “I don’t know.” That’s not an option on the job, so it shouldn’t be an option in the interview.

Of course, I’m not saying that you should know everything. That’s impossible! But you should show that you have a strategy for learning what you need to know to get the job done. Try responding with “I don’t know how to do that, but here’s how I would go about figuring it out.” This answer should go further than just saying “I’d Google it,” too. You’re better off explaining the most likely direction you would investigate.

Most of all, don’t be ashamed! Learning is largely the process of figuring out the things you don’t know. Your employer doesn’t expect you to be perfect, but they do expect you to be diligent and intelligent, and to never quit on a problem.

## 5. Always Play for the Team

Unless you’re a freelance developer, your job will always be set within the context of a team, and the team’s success is always paramount to your own. So your interview should reflect that you aren’t just concerned with solving your problems in a given day, but rather that you are focused on doing whatever is necessary for the team to succeed.

So, how can you communicate this in an interview?

### Never Discuss Problems as if They Are in a Vacuum

Almost any project would practically have resource requirements and limits, so show your awareness of the context of the problem. A problem that is solved well but has gone over budget is not truly the greatest solution.

### Show Your Awareness of the Expertise on the Existing Team

Sometimes the best answer to a question is to ask others on your team to collaborate with you, and you may reference those people in the technical questions: “If I was presented this problem in the context of our team, I would probably ask [team member’s name] to review my solution as well.” This shows that you’re willing to rely on the expertise of others and that your goal is to arrive at the best solution.

### Communicate Your Team Driven Values Explicitly

This is an important enough issue that you should come right out and say it. You want to make sure your employers are fully aware that your goal is to help the company succeed, not just to sit at your desk and code all day.

Ultimately, your job as a developer isn’t only to code. It is to be a team player, a leader, and someone who is never willing to quit on a problem. The opportunities in a technical interview are not only to show that you can meet objectives or write algorithms, but also to put yourself above the competition and show the value you add to an employer.

This is a newly added part to the document