Prerequisites and how to get the most out of this course | Coursera

We recognize that you're all coming to this course with different backgrounds, expectations and goals. So here we want to provide you with some information and suggestions for getting the most out of this course.

Even if you're not looking for a job or just want to get to the capstone, this course is still more important than you might think

We also know that not all of you are looking for software engineering positions right now, and you might not even want to take this course but you have to because you want to get to the capstone. Our goal is NOT to waste your time, we promise. This is a specialization on software engineering, which includes more than simply programming. Communication and other "soft" skills are equally important to any software engineer, and in fact in most aspects of life. Nearly all major software projects are done collaboratively and for you to be part of those projects, you need to know how to communicate about your work. You might think that this content has no place in a software engineering course, but in fact, we include assignments that focus on the same communication skills as the ones in this course in many of our in-person computer science courses as well.

"No really, I just want to get to the capstone. Do I really HAVE to pass this course?" The answer is yes. But if you *really REALLY* don't want to do the assessments, we've put in mechanisms where you can essentially skip them by passing with a very low grade. You will have to submit all the assignments (including the peer review), but we've made it so that you'll be able to pass, we all-but-guarantee it. And finally, although we've made it easier to pass, we still encourage you to take it seriously.

This course assumes a solid background in Java programming, data structures, and basic algorithms

Technical interviews will always expect you to be familiar with core data structures, from the basic to the more advanced. You need to have knowledge of these data structures, including how to build them and how to analyze their running times, in your "back pocket". In this course we will help you practice solving problems with data structures, but we will not be teaching them explicitly. If you are not fairly confident with Java or if you are unfamiliar with any of the data structures or algorithms below, you will probably want to learn a bit more before taking this course:

- Strings
- Arrays and Linked Lists

- Trees, including: binary and non-binary, binary search trees
- Heaps and priority queues

- Graphs and graph search algorithms
- Hashtables/Hashmaps
- Sorting algorithms (insertion sort, selection sort, merge sort, quick sort)
- Big-O running time analysis
- Intermediate Java programming knowledge including: Object Oriented design and programming (polymorphism, inheritance), Input/Output (from files and System.in), Exception handling and throwing, all basic programming structures (loops, variables, conditionals, etc).

If you're looking to learn or brush up on your Java or data structures knowledge, we recommend you consider taking the other courses in our specialization first. Of course, this is not an exhaustive list of the data structures you might encounter in your interviews, so throughout this course we'll try to remind you of other structures you should be sure to brush up on.

You'll learn the most if you do the assessments

There are countless YouTube videos, books and web pages out there on interviewing for software engineering jobs and internships. So what does this course add? We think that the biggest thing this course provides is the opportunity for you to ***do carefully constructed assignments*** *designed to help you practice and refine the skills that you will need in an interview*. Yes, you might learn something by just watching the videos, but unless you do the work you're not really going to get much better. So we hope you will do the assignments!

Your peers are relying on you (and you on them) to stay on schedule

If you are interested in and commit to doing the assignments, then you'll become part of a great community that is working together to improve their interviewing (and engineering) skills. The assignments use a combination of self-assessment and peer assessment--the skills you're developing in this course simply cannot be effectively graded automatically. This means that it is important that you provide quality feedback in a timely fashion to your peers. In return they will do the same for you. We provide you structure and guidance for providing this feedback, but your peers are relying on you to keep up and do the work. When everyone makes this commitment to help each other, you'll get valuable feedback in a timely fashion.

If you're just in it to watch, that's OK too

We know not all of you want to take the full course, and that's OK too. You're welcome to just browse around and watch what you like, of course! We hope you'll find the videos helpful (and feel free to send us

feedback too).

Choose your own path through the course

Finally, like the previous three courses in this specialization, this course is not designed to be completed in a linear fashion from start to finish. If you think you're ready to do an assignment, go ahead and jump right in. If you want a little more instruction, you can go back for more support. We've also got our "In the Real World" and "When I Struggled" series with stories from Google Engineers and UC San Diego students to inspire you in this challenging process of mastering the software engineering interview!

So no matter what your background, goals, etc, welcome again. We're glad to have you!