Study Guide for Software Engineers

All of the problems and questions you see below were given to me by engineering managers at FANNG and Startup companies. The purpose of this study guide is to help you feel more confident going through your job search! I want to make sure we're putting you in the best possible position to land your next role. This guide will give you problems that you could encounter on technical interviews and behavioral interviews. I also added external resources you can use to enhance your skills as an engineer that are trusted by other engineers. 🙂

Coding Problem #1 - Any Language:

You are tasked with creating a simple Pokemon battle simulation game.
The game should have two Pokemon characters with the following attributes:

- Name (string)
- Type (string): can be one of the following - Fire, Water, Grass, Electric
- Level (integer): can be any value between 1 and 100
- Health Points (HP) (integer): can be any value between 1 and 100
- Attack (integer): can be any value between 1 and 100
- Defense (integer): can be any value between 1 and 100
- Speed (integer): can be any value between 1 and 100

Each Pokemon should have four moves, with the following attributes:

- Name (string)
- Type (string): can be one of the following - Fire, Water, Grass, Electric
- Power (integer): can be any value between 1 and 100
- Accuracy (integer): can be any value between 1 and 100

The game should have a function called battle that takes in two Pokemon objects and simulates a battle between them. The function should use a simple algorithm to determine which Pokemon attacks first:

- The Pokemon with the higher speed stat attacks first.

If the speed stats are tied, then a coin flip is used to determine which Pokemon attacks first.
During the battle, each Pokemon will take turns attacking with one of its moves. The move that each Pokemon uses should be chosen randomly. The accuracy of the move should be factored in, with a random number generator used to determine if the move hits or misses.

The battle should continue until one of the Pokemon's HP drops to 0. The function should return the name of the winning Pokemon.

You can assume that the input Pokemon objects will have the required attributes, and that the moves will be stored in a list in the Pokemon object. You can also assume that the random module is already imported.

———————————————

Coding Problem #2 - Any language

Suppose you are given a list of strings representing a list of tasks to be executed, along with

their dependencies. Each task is represented by a unique string ID, and the dependencies are represented as a list of IDs that must be completed before the task can be executed.

Your task is to write a function that takes in the list of tasks and their dependencies and returns a valid order in which the tasks can be executed. If there are multiple valid orders, the function should return any one of them.

———————————

Coding Problem #3 - Any Language

Write a function that simulates a game of Blackjack between a player and a dealer.

The function should take no inputs, but should have the following output:

- If the player wins, the function should return "Player wins!"
- If the dealer wins, the function should return "Dealer wins!"
- If the game is a tie, the function should return "Tie game!" The rules of the game are as follows:
- The player and the dealer are each dealt two cards.
- The player's cards are dealt face up, while the dealer's first card is dealt face down (the "hole" card).
- The goal of the game is to get as close to 21 points as possible without going over. Face cards (kings, queens, and jacks) are worth 10 points, and aces can be worth either 1 or 11 points, whichever is more advantageous to the player.
- If the player's first two cards add up to 21 (a "blackjack"), the player wins immediately.
- If the dealer's first card is an ace, the player has the option to take insurance, which pays 2:1 if the dealer has a blackjack.
- If neither the player nor the dealer has a blackjack, the player has the option to "hit" (take another card) or "stand" (keep their current hand).
- If the player's hand exceeds 21 points, they "bust" and lose the game immediately.
- Once the player has either stood or busted, the dealer reveals their hole card and must hit until their hand exceeds 16 points or they bust.
- If the dealer busts, the player wins. Otherwise, the player and dealer hands are compared, and the one with the higher score wins. In case of a tie, the game is declared a tie.

———————————

Mind Teaser Problem

Question: How many windows are there in New York City?

One possible approach to solving this question is to use a statistical estimation method called "sampling" to gather data about the number of windows in a representative subset of buildings in New York City, and then use that data to estimate the total number of windows in the entire city. Here's an example of how you could approach this problem:

- Define the problem: The goal is to estimate the total number of windows in New York City.

- Determine the sampling methodology: We will use a simple random sampling method to select a representative subset of buildings in New York City. This involves randomly selecting a sample of buildings from a list of all buildings in the city, using a computer-generated random number sequence.
- Determine the sample size: The sample size should be large enough to provide a reliable estimate, but not so large that it becomes impractical to collect the data. Based on statistical theory, a sample size of at least 30 is typically required for accurate estimation.
- Collect data: Once the sample buildings have been selected, we will send survey teams to each building to count the number of windows in each unit. This data will be recorded in a spreadsheet or database.
- Analyze the data: Once the data has been collected, we will use statistical methods to analyze the data and

estimate the total number of windows in New York City. This may involve calculating the mean, median, or mode of the data, or fitting a statistical distribution to the data and using that to estimate the total number of windows.

- Validate the results: To ensure the accuracy of our estimate, we may conduct a second survey on a different sample of buildings, or we may compare our estimate to existing data sources, such as building permits or tax records, to ensure that our estimate is within a reasonable range.

Overall, this approach involves a combination of statistical theory, data collection, and data analysis to estimate the total number of windows in New York City. While the exact number of windows may never be known with absolute certainty, this method provides a reliable and accurate estimate based on a representative sample of buildings.

————————————

Golang Challenge:

Write a function that takes in two slices of integers, a and b, and returns a new slice that contains the elements that are present in both a and b, but in reverse order. If there are no common elements, the function should return an empty slice.

————————————

NodeJS Challenge:

Write a function that takes in an array of numbers, and returns the second smallest number in the array. If the array is empty or has only one element, return null.

————————

Python Coding Challenge:

Write a function that takes a list of integers as input and returns the two integers that add up to a specific target.

For example, given the input list '[2, 7, 11, 15]' and target value '9', the function should return the tuple '(2, 7)', as '2 + 7 = 9'.

If no two integers in the list add up to the target, the function should return 'None'.

————————————

Ruby on Rails Challenge:

You have been asked to implement a simple blog application that allows users to create, read, update, and delete blog posts.

Your task is to create a Ruby on Rails application that includes the following features:

- User authentication: Users should be able to sign up, sign in, and sign out. Only signed-in users should be able to create, edit, and delete blog posts.
- Blog post CRUD: Signed-in users should be able to create, read, update, and delete blog posts. Each blog post should have a title, body, and author.
- Homepage: The homepage should display a list of all blog posts, sorted by the most recent first. Each blog post title should be clickable and lead to the full post page.
- Full post page: The full post page should display the blog post's title, body, author, and date of creation. If the current user is the author of the post, they should be able to edit or delete the post from this page.
- Styling: The application should be styled using CSS and/or Bootstrap to make it visually appealing.
- Bonus feature: Implement a comment system where signed-in users can leave comments on blog posts. Each comment should have a body and an author.

Your application should use Rails 6 and SQLite as the database. You can use any gems or libraries that you find helpful, such as Devise for user authentication or Bootstrap for styling. Once you've completed the challenge, you can deploy your application to a hosting service like Heroku.

———————————————-

React Coding Challenge

Brief

You need to create a simple todo list application in React. The application should allow users to add new tasks, mark tasks as completed, and delete tasks. Users should also be able to filter tasks by completed status.

Requirements

- The user should be able to add new tasks by entering text and pressing a button.
- The user should be able to mark a task as completed by clicking a checkbox next to the task.
- The user should be able to delete a task by clicking a button next to the task.
- The user should be able to filter tasks by completed status using a dropdown menu.
- The application should be responsive and work on mobile devices.
- The application should use React and any additional libraries or tools you choose.

Bonus

If you want to go above and beyond, here are some bonus features you could add:

- Allow users to edit tasks.
- Add due dates to tasks.
- Allow users to sort tasks by due date or completion status.
- Use local storage to save the user's tasks between sessions.

Tips

- Break the problem down into smaller parts and tackle them one at a time.

———————————————————

NextJS Coding Challenge:

Design and build a Mortal Kombat character profile page using Next.js that displays information about a specific Mortal Kombat character. The page should have the following requirements:

- Character information: The page should display basic information about the character, including their name, photo, and background story.
- Moves list: The page should display a list of the character's special moves and fatalities, along with a brief description of each move.
- Video clips: The page should display video clips of the character performing their special moves and fatalities.
- Navigation: The website should have a navigation menu that allows users to navigate to different character profile pages.
- Responsive design: The website should be designed to be responsive, with a layout that adjusts to different screen sizes.

To complete this challenge, you would need to have a solid understanding of React and Next.js. You would need to use dynamic routing to create a unique URL for each character profile page. Additionally, you would need to implement a video player using a video player library like Video.js or Plyr. Finally, you would need to use CSS or a CSS library like Bootstrap or Material UI to design the website and make it responsive.

———————————————————-

Typescript Coding Challenge

Brief

You are tasked with creating an employee management system in TypeScript. The system should allow managers to add new employees, update employee information, delete employees, and view a list of employees. Each employee should have a name, job title, and salary.

Requirements

- The user should be able to add new employees by entering their name, job title, and salary into a form and clicking a button.
- The user should be able to update an employee's information by selecting the employee from a list and editing their name, job title, or salary in a form.
- The user should be able to delete an employee by selecting the employee from a list and clicking a button.
- The user should be able to view a list of employees, including their name, job title, and salary.
- The application should use TypeScript and any additional libraries or tools you choose.

Bonus

If you want to go above and beyond, here are some bonus features you could add:

- Allow managers to sort employees by name or salary.
- Allow managers to search for employees by name or job title.
- Use local storage to save the list of employees between sessions.

Tips

- Use interfaces to define the shape of your employee objects.
- Break the problem down into smaller parts and tackle them one at a time.

———————————————————

Java Coding Challenge:

Brief

You are tasked with creating a library catalog system in Java. The system should allow librarians to add new books, update book information, delete books, and view a list of books. Each book should have a title, author, and ISBN.

Requirements

- The user should be able to add new books by entering their title, author, and ISBN into a form and clicking a button.
- The user should be able to update a book's information by selecting the book from a list and editing its title, author, or ISBN in a form.
- The user should be able to delete a book by selecting the book from a list and clicking a button.
- The user should be able to view a list of books, including their title, author, and ISBN.
- The application should use Java and any additional libraries or tools you choose.

Bonus

If you want to go above and beyond, here are some bonus features you could add:

- Allow librarians to sort books by title or author.
- Allow librarians to search for books by title or author.
- Use a database to store the list of books.

Tips

- Use classes to define the shape of your book objects.
- Break the problem down into smaller parts and tackle them one at a time.

———————————————

Architectural Design Questions:

Design a scalable architecture for a social media platform that allows users to share text, photos, and videos. The platform should be able to handle millions of users and billions of posts. Consider the following architectural considerations:

What to Consider:

- Data storage and retrieval: How will you store and retrieve user data, posts, comments, and media files? What kind of database system will you use? How will you handle the large volume of data and ensure efficient querying?
- Load balancing: How will you handle the large number of users and traffic on the platform? What kind of load balancing strategy will you use? Will you need to use a content delivery network (CDN) to improve performance?
- Security and privacy: How will you ensure the security and privacy of user data? What kind of encryption and authentication mechanisms will you use? How will you handle user permissions and access control?

- User experience: How will you design the user interface and user experience to ensure ease of use and engagement? What kind of features and functionalities will you offer to users?
- Monetization: How will you monetize the platform? Will you use ads, subscriptions, or a combination of both? How will you balance the need for revenue with the need to provide a positive user experience?
- Mobile compatibility: How will you ensure that the platform is compatible with mobile devices? Will you need to develop native mobile apps or use a responsive web design?
- Analytics and reporting: How will you track user behavior and engagement on the platform? What kind of analytics and reporting tools will you use to gain insights and improve the platform?

———————————————

Systems Design Question:

Design a system for a ride-sharing company like Uber or Lyft. The system should allow riders to request rides and drivers to accept and complete those rides.

Consider the following requirements:

- Rider app: The rider app should allow users to request rides, track the location of their driver, and pay for rides.
- Driver app: The driver app should allow drivers to accept and complete ride requests, track rider location,

and receive payment for completed rides.

- Matching algorithm: The system should include a matching algorithm that matches riders with nearby drivers based on factors like location, availability, and ride history.
- Payment processing: The system should handle payment processing, including rider payments, driver payouts, and fees to the ride-sharing company.
- Rating system: The system should include a rating system that allows riders to rate drivers and drivers to rate riders. The system should also use these ratings to improve the matching algorithm.
- Security: The system should ensure the security of user data, including personal information, payment information, and ride history.
- Scalability: The system should be designed to handle a large number of users and requests, with minimal downtime and performance issues.

To solve this problem, you would need to use a combination of software development skills, including front-end and back-end development, database design, and API integration. You would also need to consider factors like scalability, security, and user experience when designing the system.

———————————————————

List of Potential Technical Questions a hiring manager might ask a Senior Software Engineer

- What design patterns have you used in your projects, and when did you find them useful?
- Can you explain the difference between a process and a thread?
- What is the purpose of a load balancer, and how have you configured one in the past?
- How do you debug performance issues in a large distributed system?
- Can you explain how a database index works, and how it affects query performance?
- What is your experience with microservices architecture, and how have you handled inter-service communication?

- How do you ensure the reliability and availability of your applications, and what tools or processes have you used to achieve this?
- Can you explain the CAP theorem and how it applies to distributed systems?
- What is your experience with containers, and how have you used them in your projects?
- Can you explain the difference between SQL and NoSQL databases, and when would you choose one over the other?
- What is your experience with cloud computing platforms, and how have you leveraged them in your projects?
- How do you approach testing and quality assurance in your projects, and what tools or processes do you use to ensure code quality?
- Can you explain how a blockchain works, and what are its potential applications in the industry?
- What is your experience with DevOps practices, and how have you integrated them into your development workflow?
- Can you walk me through your approach to building scalable, fault-tolerant systems

———————————————————

List of potential Technical Questions a hiring manager might ask a mid-level engineer

- What experience do you have with building and maintaining web applications? Can you describe a project you worked on in detail?
- Explain how you would implement a caching system in a web application to improve performance. What factors would you consider when choosing a caching strategy?
- What is your experience with testing? How do you ensure that your code is well-tested and reliable? Have you used any specific testing frameworks or tools?
- Have you worked with databases before? How do you optimize database performance, and what techniques do you use to ensure data consistency and reliability?

- Can you describe a challenging problem you faced in a project, and how you solved it? What steps did you take to identify the problem, and what solution did you come up with?
- Describe your experience with DevOps tools and practices. Have you worked with continuous integration/continuous deployment (CI/CD) pipelines, containerization, or infrastructure as code (IaC)?
- Explain how you ensure that your code is maintainable and scalable. What techniques or design patterns do you use to make your code more modular and extensible?
- Have you worked with any front-end frameworks or libraries such as React, Vue, or Angular? What are some benefits and drawbacks of using a front-end framework?
- How do you ensure that your code is secure? What techniques do you use to prevent common security vulnerabilities such as cross-site scripting (XSS) and SQL injection?
- Can you describe a time when you had to lead a technical project or mentor a junior developer? What were some challenges you faced, and how did you overcome them?

———————————————

Potential Behavioral Interview Questions you can expect from a hiring team

- Can you tell me about a time when you faced a technical challenge that you didn't know how to solve? How did you approach the problem, and what steps did you take to find a solution?

- How do you approach collaborating with other team members? Can you describe a project where you worked closely with someone who had a different skillset or perspective?
- Can you tell me about a time when you had to explain a technical concept to a non-technical person? How did you make sure that they understood the concept?
- How do you stay up to date with industry developments and new technologies? Can you describe a recent learning experience or project that you worked on outside of work?
- Can you tell me about a time when you had to deal with a difficult team member or stakeholder? How did you handle the situation, and what steps did you take to resolve any conflicts?
- Can you describe a time when you had to make a difficult trade-off between technical quality and project deadlines? How did you make the decision, and what were the outcomes?
- Can you tell me about a time when you took initiative to improve a process or system at work? What was the problem you identified, and what steps did you take to improve the situation?
- Can you describe a project where you had to work with a tight budget or resource constraints? How did you manage the project, and what challenges did you face?
- Can you tell me about a time when you received feedback on your work? How did you incorporate the feedback, and what did you learn from the experience?
- Can you describe a project where you had to work with a team that was distributed across different locations or time zones? What challenges did you face, and how did you collaborate effectively with the team?

———————————————

Potential Interview Questions you could be asked by a hiring manager at a startup:

- What is your experience working in a fast-paced, agile environment? Can you describe a time when you had to adapt quickly to changing requirements or priorities?
- How do you approach problem-solving when faced with limited resources or budget constraints? Can you describe a time when you had to make trade-offs between technical quality and project deadlines?
- How do you prioritize competing demands in a rapidly changing environment? Can you give an example of a project where you had to make decisions quickly and pivot the project direction?
- Can you describe a time when you had to work independently and take ownership of a project or initiative? How did you ensure that you were meeting project requirements and deadlines without direct supervision?
- Can you tell me about a project where you had to wear multiple hats, such as working on both the front-end and back-end development? How did you manage your time and balance the competing demands of different parts of the project?
- Can you describe a project where you had to work with a small team of developers? How did you collaborate with team members to ensure that you were meeting project requirements and delivering a high-quality product?

- How do you stay up to date with new technologies and developments in the industry? Can you give an example of a time when you used a new technology to solve a problem or improve a product?
- Can you tell me about a time when you had to make a difficult technical decision that impacted the project's outcome? How did you approach the decision, and what were the results?
- How do you approach testing and quality assurance in a startup environment? Can you describe a project where you had to balance quality assurance and testing with fast-paced development requirements?

— — — — — — — — — — — — — —

Additional Resources:

Where you can find more coding challenges:

- HackerRank - offers coding challenges in various programming languages and domains, as well as interview preparation materials and mock interviews.
- LeetCode - offers a large collection of coding challenges in various programming languages, as well as a discussion forum to ask questions and get help from the community.
- Codility - offers coding challenges in various domains, including algorithmic thinking, data structures, and databases, as well as customizable assessments for technical hiring.
- TopCoder - offers coding challenges and competitions in various domains, including algorithmic thinking, machine learning, and game development.
- Project Euler - offers a collection of challenging mathematical and computational problems, which can help to develop problem-solving skills and algorithmic thinking.
- CodeSignal - offers coding challenges and skills assessments in various programming languages, as well as interview preparation materials and practice interviews.
- Codecademy - offers coding courses and projects to help develop coding skills in various programming languages and domains.
- InterviewBit - offers coding challenges and interview preparation materials for software engineering interviews, including behavioral and technical questions.
- GeeksforGeeks - offers coding challenges, tutorials, and interview preparation materials for various programming languages and domains, including data structures and algorithms.
- Coderbyte - offers coding challenges and exercises in various programming languages and domains, as well as interview preparation materials and mock interviews.

Ways to keep up with trends in the industry through Forums, Blogs, and Podcasts:

Forums:

- Reddit r/programming: A popular subreddit where developers can share news, articles, and insights on programming and software engineering.
- Stack Overflow: A question-and-answer site where developers can get answers to technical questions, share knowledge, and learn from other developers.
- Hacker News: A community-driven news site where developers can share news and insights on software engineering, startups, and technology.

Podcasts:

- Software Engineering Daily: A daily podcast that covers a wide range of topics in software engineering, including machine learning, DevOps, and cybersecurity.
- The Changelog: A weekly podcast that covers a range of topics in open source software, including web development, databases, and DevOps.
- CodeNewbie: A podcast for beginner and intermediate programmers that covers topics like getting started with coding, finding mentors, and developing coding skills.
- JavaScript Jabber: A podcast that covers JavaScript and related technologies, including Node.js, React, and Angular.
- SE Radio: A podcast that covers software engineering and related topics, including design patterns, software architecture, and programming languages.

Blogs and News Sources:

- TechCrunch: A leading source of news and analysis on technology, startups, and venture capital.
- Wired: A popular news source that covers technology, science, and culture, with a focus on the latest trends in technology.
- The Verge: A technology news and media network that covers a wide range of topics, including software engineering, hardware, and consumer electronics.
- InfoWorld: A website that covers news, reviews, and analysis of software engineering and technology trends.
- GitHub Blog: The official blog of GitHub, which covers a range of topics related to software engineering, including open-source projects, developer tools, and best practices.
- Codecademy Blog: A blog that offers tutorials and resources for beginner and intermediate programmers, covering a range of programming languages and frameworks.
- Smashing Magazine: A website that covers web development and design, with a focus on the latest trends, tools, and techniques.
- O'Reilly Ideas: A blog that covers topics related to software engineering, technology, and business, with a focus on emerging trends and innovations.