Java Full Stack Review Questions

Java

Basics

1. What is Java? / Explain some features of Java
2. What is JRE / JDK / JVM?
3. What is the difference between an object and a class?
4. What is the root class from which every class extends?
5. What are the primitive data types in Java?
6. Where are Strings stored?
7. Explain stack vs heap
    a. Are variable references stored on the stack or heap? What about the objects they refer to?
    b. What is a stack frame? When are these created?
8. What are annotations?
9. What is a POJO vs a bean?
10. Can you force garbage collection in Java? When is an object eligible for GC?
11. Why are strings immutable in java? How would you make your own objects immutable?
12. What is the difference between String, StringBuilder, and StringBuffer?
13. What are the different variable scopes in Java?
14. What are the access modifiers in Java? Explain them.
15. What are the non-access modifiers in Java?
16. ‘What is the difference between static and final variables?
17. What are the default values for all data types in Java?
18. What is a wrapper class? List them.
19. What is autoboxing / unboxing?
20. Is Java pass-by-value or pass-by-reference?
21. What makes a class immutable?
22. If two objects are equal, do they have the same hashcode? If not equal?
23. What data types are supported in switch statements?
24. List all non-access modifiers
25. How to pass multiple values with a single parameter into a method?
26. What is a static block?
27. What are static imports?
28. What methods are available in the Object class?
29. What is the difference between == and .equals()?
30. What is an enhanced for loop and what is a forEach loop?
31. What are the 3 usages of “super” keyword?
32. What is the first line of any constructor?

33. How would you perform constructor chaining?
34. What happens if you don't define a constructor for a class? Can you still instantiate it?

OOP

1. What are the 4 pillars of OOP? Explain each and give examples of you implement them in Java code
2. hoExplain the principles of the SOLID acronym
3. What is the difference between an abstract class and an interface?
4. Can abstract methods have concrete methods? Can concrete (non-abstract) classes have abstract methods?
5. Can static methods access instance variables? Can non-static methods access static variables?
6. What are the implicit modifiers for interface variables? methods?
7. What is the difference between method overloading and overriding? What are the rules for changing the method signature of overloaded methods?
8. Can you overload / override a main method? static method? a private method? a default method? a protected method?
9. What are covariant return types? What rules apply to return types for overridden methods?
10. When do you use extends or implements keywords?
11. What are enumerations (enums)?
12. What are the implicit modifiers for interface variables / methods?

Collections / Generics

1. What are collections in Java?
2. What are the interfaces in the Collections API?
3. What is the difference between a Set and a List?
4. What is the difference between an Array and an ArrayList?
5. What is the difference between ArrayList and Vector?
6. What is the difference between TreeSet and HashSet?
7. What is the difference between HashTable and HashMap?
8. Are Maps in the Collections API? What makes Map different from other interfaces?
9. List several ways to iterate over a Collection. How would you iterate over a Map?
10. What is the purpose of the Iterable interface? What about Iterator?
11. What is the difference between the Comparable and Comparator interfaces?
12. What are generics? What is the diamond operator (<>)?
13. Create and instantiate a generic class. Create and use a generic method.

Threads

1. What is multi-threading?
2. In what ways can you create a thread?

3. List the methods in the Thread class and Runnable interface
4. Explain the lifecycle of a thread
5. What is deadlock?
6. What is the synchronized keyword?

IO / Serialization

1. How do you serialize / deserialize an object in Java?
2. What is a Marker interface? What does Serializable interface do?
3. What are transient variables?
4. Difference between FileReader and BufferedReader?
5. Explain the try-with-resources syntax
6. List some methods in the Scanner class

Exceptions

1. What is the difference between final, .finalize(), and finally?
2. Explain throw vs throws vs Throwable
3. Do you need a catch block? Can you have more than 1? Is there an order to follow?
4. What is base class of all exceptions? What interface do they all implement?
5. List some checked and unchecked exceptions?
6. Multi-catch block - can you catch more than one exception in a single catch block?

Reflections API

1. What is Reflection API?
2. What can you do with the Reflections API that you can't do in normal code?
3. List some classes and methods of the Reflections API

Design patterns

1. What are Singleton / Factory design patterns?
2. How would you create a Singleton?
3. Explain the DAO design pattern

JDBC

1. What is JDBC?
2. What are the core interfaces / classes in JDBC?
3. What is a stored procedure and how would you call it in Java?
4. What is the difference between Statement and PreparedStatement?
5. Steps to executing an SQL query using JDBC?
6. How to execute stored procedures using JDBC?
7. Which interface is responsible for transaction management?

JUnit

1. What is JUnit?
2. What is TDD?
3. What are the annotations in JUnit? Order of execution?
4. Give an example of a test case
5. How would you prevent a test from being run without commenting it out?

Log4j

1. What are the advantages to using a logging library?
2. What is log4j?
3. What are the logging levels of log4j?

Maven

1. What is Maven?
2. What is the default Maven build lifecycle?
3. Where / when does Maven retrieve dependencies from? Where are they stored locally?
4. What is the POM and what is the pom.xml?
5. What defines Maven project coordinates?

Advanced

1. What are functional interfaces? List some that come with the JRE for Java 8
2. What are lambdas?
3. What is try-with-resources? What interface must the resource implement to use this feature?
4. How to make numbers in your code more readable?
5. Which collections cannot hold null values?
6. If 2 interfaces have default methods and you implement both, what happens?
7. If 2 interfaces have the same variable names and you implement both, what happens?
8. Why does HashTable not take null key?
9. What new syntax for creating variables was introduced with Java 10?
10. Is there an interactive REPL tool for Java like there is for languages like Python?
11. What are collection factory methods?

SQL

1. Explain what SQL is. What are some SQL databases?
2. Draw a simple ERD for modelling Students and Classes
3. What are the 5 sublanguages of SQL? Which commands correspond to them?
4. What is the difference between DELETE, DROP, and TRUNCATE commands?
5. What are some SQL clauses you can use with SELECT statements?
6. What is the difference between WHERE and HAVING?
7. Explain what the ORDER BY and GROUP BY clauses do
8. Explain the concept of relational integrity
9. List the integrity constraints
10. Define the word “schema”
11. What is a candidate key? What about a surrogate key?
12. What conditions lead to orphan records?
13. What are some SQL data types?
14. What is normalization? What are the levels?
15. What are the properties a transaction must follow?
16. Explain the different isolation levels. What read phenomena do each prevent?
17. What is the difference between joins and set operators?
18. What are the types of joins? Explain the differences.
19. Explain the difference between UNION, UNION ALL, and INTERSECT
20. What is a cascade delete?
21. What is the purpose of a view? What about an index?
22. What’s the difference between a clustered and non-clustered index?
23. What is a trigger? Give the syntax for creating a trigger.
24. How would you setup a primary key that automatically increments with every INSERT statement?
25. What is the difference between scalar and aggregate functions? Give examples of each
26. What’s the difference between implicit and explicit cursors?

HTML/CSS

HTML

1. What is HTML?
2. What is the HTML5 doctype declaration?
3. List some tags. What is <head> used for? <body>?
4. What are the required tags for an HTML document?
5. What is the tag for ordered list? unordered list? Change bullet styling?
6. What features came with HTML5? Are HTML 5 tags different from other tags?
7. Do all tags come in a pair? List a self-closing tag.
8. What's the difference between an element and an attribute? List some global attributes.
9. What is the syntax for a comment in HTML?
10. What tags would you use to create a table? A navbar? What about a form?
11. What's the difference between a class and id?
12. How would you include CSS into an HTML document? What about JS?
13. What is a semantic tag? What about formatting tags/elements?
14. What's the difference between a block and an inline element?

CSS

15. What is CSS? what are the different ways of styling an html file?
16. Which way has highest priority when styles cascade: inline, internal, and external stylesheets. Which is best practice? Why?
17. What are the different CSS selectors? Write the syntax for each.
18. What is a psuedo-class? What is the syntax for selecting that?
19. Write a CSS selector for styling all spans inside of a div. What about only targeting spans that are direct descendents of divs?
20. Can I select multiple elements at once with CSS? If so, what is the syntax?
21. Explain the concept of specificity and how it relates to styling conflicts
22. Explain the CSS box model
23. What features did CSS3 introduce?
24. What is Bootstrap? What are some bootstrap classes you can use?
25. How many columns make up the Bootstrap grid system?
26. What is a CDN? what are the benefits? When would you choose to use a CDN vs downloading and using the Bootstrap source code in your project?

JavaScript

Language Fundamentals

1. What is JavaScript? What do we use it for?
2. Can we run JavaScript in a web browser, on a server, or both?
3. What programming paradigm(s) does JS support?
4. What are the data types in JS?
   a. What is the type of NaN? What is the isNaN function?
   b. What is the data type of a function?
   c. What about an array?
   d. What is the difference between undefined and null?
5. What are JS objects? what is the syntax?
6. What is JSON? Is it different from JS objects?
7. What are some ways you can use functions in JS?
8. What are the different scopes of variables in JS?
   a. What are the different ways to declare global variables?
   b. Is it a best practice to use global variables? Why or why not?
9. What is function and variable hoisting?
10. What is the global object in client-side JavaScript?
    a. What are some built-in functions (methods on the global object)?
    b. How would you set some code to run at a specified time in the future? What about repeatedly?
11. Explain how the guard and default operators work
12. What are callback functions? What about self-invoking functions?
13. What is closure and when should you use it?
14. Use the object literal syntax to create an object with some properties
15. What is a truthy or falsy value? List the falsy values.
16. What is the difference between == and ===? Which one allows for type coercion?
17. Explain the template literal syntax
18. What is a Promise?
19. What are arrays in JS? can you change their size?
20. List some array methods and explain how they work
21. Explain what “strict mode” does
22. Explain how inheritance works in JS
23. What does the "this" keyword refer to?
24. Explain the concept of lexical scope
25. What are some methods on the function prototype?
26. What will happen when I try to run this code: console.log(0.1+0.2==0.3) ?

ES6+

27. What new features did ES6 introduce?
28. What is the difference between var, let, and const keywords?
29. Give the syntax for template literals / string interpolation
30. What's the difference between a normal function declaration and an arrow function?
31. Does JS have classes? If so, when were they introduced?
32. What is object and array destructuring? What is the rest/spread operator?
33. How would you set default values for parameters to a function?
34. What is the difference between for-of and for-in loops?
35. What's the benefit of computed property names?
36. Explain the async/await keywords. Why is it preferred to use this instead of .then() methods?
37. What is a generator function? What's the point of the yield keyword?
38. What built-in data structures does JavaScript provide?

Events and DOM

39. What is the DOM? How is it represented as a data structure? What object in a browser environment allows us to interact with the DOM?
40. List some ways of querying the DOM for elements
41. How would you insert a new element into the DOM?
42. What are event listeners? What are some events we can listen for? What are some different ways of setting event listeners?
43. What is bubbling and capturing and what is the difference?
44. What are some methods on the event object and what do they do?
45. How would you submit a form using JS?

AJAX

46. What is AJAX? why do we use it?
    a. What are the benefits of using AJAX?
    b. Are there any downsides of using AJAX?
47. Explain why it is important that AJAX is asynchronous
48. List the steps to sending an AJAX request
49. What are steps to sending an AJAX request?
50. List the different ready states of the XmlHttpRequest object
51. How does the fetch API differ from the XHR object?

Servlets

1. What is a servlet? What about a servlet container? Which servlet container have you worked with?
2. Describe the servlet class inheritance hierarchy. What methods are declared in each class or interface?
3. How would you create your own servlet?
4. What is the deployment descriptor? What file configures your servlet container?
5. Explain the lifecycle of a servlet - what methods are called and when are they called?
6. Is eager or lazy loading of servlets the default? How would you change this?
7. What are some tags you would find in the web.xml file?
8. What is the difference between the ServletConfig and ServletContext objects? How do you retrieve these in your servlet?
9. What is the purpose of the RequestDispatcher?
10. Explain the difference between RequestDispatcher.forward() and HttpServletResponse.sendRedirect()
11. What are some different ways of tracking a session with servlets?
12. What is object mapping? Any Java libraries for this?
13. How would you send text or objects back in the body of the HTTP response from a servlet?
14. What is the difference between getParameter() and getAttribute() methods?
15. Explain the front controller design pattern

Angular

1. What makes a “single page application” (SPA) different from a normal web page?
2. Explain the difference between server-side and client-side rendering
3. What are some features of the Angular framework?
4. How does TypeScript relate to JavaScript? What are the major benefits of using it over JavaScript?
5. List the data types of TypeScript
6. How would you create a new Angular project?
7. What is a component? How would you create one? List some other commands using the Angular CLI
8. What files make up a component? What is the “spec” file used for?
9. Explain the relevance of npm to Angular projects. Which file does npm use to track dependencies?
10. List some decorators for Angular apps
11. What is the lifecycle of a component? List some lifecycle hooks
12. What is a directive and what are the different types? How to tell these directives apart with syntax?
13. What is the benefit of using a directive like NgClass over the class attribute, or even property binding to the class attribute?
14. What is a pipe? A service?
15. How would you create a custom pipe? What about a service?
16. How does dependency injection work in Angular?
17. What is an Angular module? What properties should you set inside it?
18. What’s the difference between a JavaScript module and Angular module? What are some common Angular modules?
19. How would you lazy load a module?
20. How have you used the HttpClient? What methods does it have and what do they return?
21. What is an Observable? What’s the difference between it and a Promise?
22. What forms of data binding does Angular support? Explain the syntax for each
23. What does Webpack do for your ng project?
24. How would you implement routing in your project?
25. What is an EventEmitter and when would you use one?
26. What’s the difference between using reactive and template-driven forms? How would you setup each?
27. How would you run your unit tests for an Angular project?

DevOps + Linux + AWS

DevOps

1. What is DevOps? What is the goal of various DevOps processes?
2. Explain CI/CD. What is the difference between Continuous Deployment and Continuous Delivery?
3. What tools have you used to achieve CI/CD? Explain how you've setup and used them
4. What is a DevOps pipeline? Explain the steps to setting one up
5. What is SonarQube / SonarCloud? Explain some of the features of it
6. What is a "build"? What is the end result of a build? What is the build tool you've used for Java projects?
7. What are the Maven lifecycles? List the steps in the build lifecycle

Linux

8. What is Unix? Linux? How are Linux OS's different from other OS's?
9. List some Linux distributions
10. Explain the terms terminal, shell, console, command-line?
11. What is the bash shell? How would you write a bash script?
12. What is the shebang syntax and purpose?
13. What is an environment variable and how would you set one?
14. What is the difference between relative and absolute paths?
15. Where are the root and home directories located? How to get to each?
16. What Linux command would you use to:
    a. Search for text in a text file
    b. Navigate your file hierarchy on the command line?
    c. List files? What about hidden files? See permissions for the files?
    d. Change the permissions of a file
    e. Edit a file from the terminal
17. How do file permissions work on a Linux system?
18. What is a package manager? What are some common Linux package managers?
19. What does "I/O redirection" mean?
20. Explain the following bash commands:
    a. cat actors.txt | grep Tom Cruise
    b. echo hello world > myfile.txt
21. You run a command but get an error message saying you don't have enough permissions. What could you try to resolve this?
22. You spin up a web server that listens for web traffic on port 80, but you forget the name of the process. How would you look this up and then terminate the process on a Linux command line?

AWS

23. How would you describe AWS? What is “the cloud” or “cloud computing” and why is it so popular now?
24. Define Infrastructure, Platform, and Software as a Service
25. What’s the difference between a Region and an Availability Zone (AZ)?
26. How are you charged for using AWS services?
27. Explain the following AWS services:
    a. EC2
    b. EBS
    c. RDS
    d. Route 53
    e. VPN
    f. S3 / Glacier
    g. Lambda
    h. AMI
    i. IAM
    j. ELB
28. What steps would you take to create an EC2 and connect to it via your shell?
29. What configuration options are available for an EC2?
30. What are Security Groups? When defining a rule for a security group, what 3 things do you need to specify?
31. What’s the difference between scalability, elasticity, and resiliency? What is autoscaling?

Hibernate

1. What is Hibernate? What is JPA?
2. What is the benefit of using Hibernate over JDBC?
3. Tell me about some of the JPA annotations you have worked with? What do they do? How do you specify multiplicity relationships with JPA annotations?
4. What are the interfaces of Hibernate?
5. Tell me how you set up hibernate? What files are you editing, what goes in them, etc.
6. What ways are available to you to map an object to database entities in Hibernate?
7. In the session interface, what is the difference between save and persist methods? get and load methods? Update vs merge methods?
8. What are the different session methods?
9. What is the difference between Eager and Lazy fetching and how to setup either?
10. Under what circumstances would your program throw a LazyInitializationException?
11. What are the 4 ways to make a query using Hibernate?
12. What is HQL? What makes it different from SQL?
13. What is the Criteria API? Can you perform all DDL and DML commands with it? How do Restrictions and Projections work within this API?
14. What is caching? What is the difference between L1 and L2 cache?
15. How do you enable second level caching?
16. Tell me about NamedQueries.
17. Can you write native SQL with Hibernate? Is this a good idea?
18. What are the configuration options for Hibernate?
    a. How to specify the SQL dialect?
    b. What data must be specified for the SessionFactory?
    c. What is hbm2ddl?
    d. How would you configure Hibernate to print to the console all SQL statements run?
19. What are the different object states in Hibernate? What methods move objects to different states?
20. What is a proxy? When does the proxy resolve to the real object?
21. What is the difference between Dynamic Insert and Dynamic Update?
22. What is automatic dirty checking?
23. What is Transactional Write Behind?
24. Explain how transaction propagation works

Spring

Spring Core

1. What are Spring Projects and Spring Modules?
2. What is IOC and what does the IOC Container do?
3. What is dependency injection and what are some of the benefits of using dependency injection?
4. What types of dependency injection does Spring support?
5. What are some differences between BeanFactory and ApplicationContext? Which one eagerly instantiates your beans?
6. What is the Spring Bean lifecycle?
7. What is bean wiring? What about autowiring?
8. What are the different ways that Spring can wire beans?
9. What are the scopes of Spring beans? Which is default?
10. What is the concept of component scanning and how would you set it up?
11. What are the benefits and limitations of Java configuration?
12. What does the @Configuration and @Bean annotations do?
13. What is @Value used for?
14. What is Spring Expression Language? What can you reference using SpEL? What is the difference between $ and # in @value expressions?

Spring MVC

15. Explain the MVC architecture and how HTTP requests are processed in the architecture
16. What is the role of the DispatcherServlet? What about the ViewResolver?
17. List some stereotype annotations. What are the differences between these?
18. How would you declare which HTTP requests you'd like a controller to process?
19. What is the difference between @RequestMapping and @GetMapping?
20. How to declare the data format your controller expects from requests or will create in responses?
21. What annotation would you use to bypass the ViewResolver?
22. How would you extract query and path parameters from a request URL in your controller?
23. What concerns is the controller layer supposed to handle vs the service layer?
24. How would you specify HTTP status codes to return from your controller?
25. How do you handle exceptions thrown in your code from your controller? What happens if you don't set up any exception handling?
26. How would you consume an external web service using Spring?
27. What are the advantages of using RestTemplate?

Spring AOP

28. What is “aspect-oriented programming”? Define an aspect.
29. Give an example of a cross-cutting concern you could use AOP to address
30. Define the following:
    a. Join point
    b. Pointcut
    c. Advice
31. What are the different types of advice? What is special about the @Around advice? How is the ProceedingJoinPoint used?
32. Explain the pointcut expression syntax
33. What visibility must Spring bean methods have to be proxied using Spring AOP?

Spring Data

34. What is Spring ORM and Spring Data?
35. What is the Template design pattern and what is the JDBC template?
36. What does @Transactional do? What is the PlatformTransactionManager?
37. What is a PersistenceContext?
38. Explain how to integrate Spring and Hibernate using ContextualSession
39. What interfaces are available in Spring Data JPA?
40. What is the difference between JPARepository and CrudRepository?
41. What is the naming conventions for methods in Spring Data repositories?
42. How are Spring repositories implemented by Spring at runtime?
43. What is @Query used for?

Spring Boot

44. How is Spring Boot different from legacy Spring applications? What does it mean that it is “opinionated”?
45. What does “convention over configuration” mean?
46. What annotation would you use for Spring Boot apps? What does it do behind the scenes?
47. How does Boot’s autoconfiguration work?
48. What is the advantage of having an embedded Tomcat server?
49. What is the significance of the Spring Boot starter POM?
50. What is the Spring Boot actuator? What information can it give you?
51. What files would you use to configure Spring Boot applications?
52. What is the benefit of using Spring Boot profiles?

Web Services

HTTP

1. What is a “web service”? What’s the advantage of distributing software as a web service?
2. What’s the difference between REST and SOAP services?
3. What does HTTP stand for?
4. What are the components inside of an HTTP request? What about an HTTP response?
5. What are the important HTTP verbs / methods and what do they do?
    a. Which are idempotent?
    b. Which are safe?
6. List the levels of HTTP status codes and what they mean
7. What are some specific HTTP status codes that are commonly used?
8. What is service-oriented architecture (SOA)?
9. How can you achieve loose coupling with SOA?
10. What is an Enterprise Service Bus (ESB)?
11. What are some best practices when creating web services?

SOAP

12. What does the acronym SOAP stand for?
13. What protocols and data format do SOAP services use?
14. What is the “contract” for a SOAP service?
15. What’s the structure of a SOAP message?
16. What are the SOAP messaging modes? Messaging Exchange Patterns?
17. Are SOAP messages delivered with GET or POST requests?

REST

18. What does the acronym REST stand for? What makes a service “RESTful”?
19. What protocols and data format do REST services use?
20. What are the architectural constraints of REST?
21. Explain the levels of the Richardson Maturity Model
22. Explain the HATEOAS concept
23. What is a “resource” in a REST service?
24. What does the “uniform interface” constraint mean? Give an example of some RESTful endpoints you would create for an API. Should the URLs contain nouns, verbs, or adjectives?
25. How would you implement authentication/authorization in a RESTful web service while maintaining statelessness?

Microservices

1. Compare the microservice and monolithic architectures. What are the advantages / disadvantages of each?
2. Can a Java microservice communicate with a Node.js microservice? Why or why not?
3. What's the difference between horizontal and vertical scalability? Which way do monoliths and microservices typically scale?
4. What are some best practices for building microservices?
5. What is a messaging queue and how is using one different from calling service APIs directly?
6. What is a "replica"?
7. Explain the Netflix OSS stack for microservices (Eureka, Zuul, Hystrix)
8. How would you setup and configure Eureka? Zuul? Hystrix?
9. What is the purpose of an API gateway and how does Zuul perform this?
10. What is service discovery and how does Eureka do this?
11. What is the circuit breaker pattern and how does Hystrix implement it? What are the different circuit states?
12. Is there any order in which you need to spin up these services?
13. One of my microservices is throwing a TransportException! What might be the problem?

Docker

14. What is containerization?
15. How are containers different from virtual machines?
16. What is a Docker image? container?
17. List the steps to create a Docker image and spin up a container
18. What is the relevance of the Dockerfile to this process? List some keywords in the Dockerfile
19. What is the benefit to an image being built in Layers?
20. What are some other Docker commands?
21. What is a container registry? How would you retrieve and upload images to DockerHub?
22. What is Docker compose and why is it useful?
23. If you want to store state for a container, how does Docker recommend doing that?

Kubernetes

1. What is container orchestration? How does Kubernetes help with this?
2. What is a pod? A service? A deployment? A replicaSet?
3. Explain the pod lifecycle
4. What configuration file would you write for a Kubernetes deployment and what properties would you set?
5. What is the difference between a pod and a node?

6. What containers would be grouped together in a pod?
7. What is the purpose of the master and worker nodes on Kubernetes?
8. List some kubectl commands you know and what they do
9. How do you handle secrets in Kubernetes?

Misc/Other

Git

1. What is version control? What makes git different from other version control software?
2. List the git commands you know and what they do
3. How would you prevent a file from being tracked by git?
4. What is a branch? What are some common branching strategies?
5. What is a merge conflict? How do you prevent these and resolve if it happens?
6. What is a GitHub pull request?
7. What is the git workflow for editing code and saving changes?
8. What is a commit?
9. How would you go back in your commit history if you make a mistake?

SDLC

1. What are the steps in the software development lifecycle?
2. What is the difference between Waterfall and Agile methodologies? Explain the benefits and drawbacks of each
3. List some of the principles declared in the Agile manifesto
4. What specific Agile frameworks exist? What are the main features of each?
5. What is the Scrum process? Explain each of the Scrum ceremonies
6. How long is a typical sprint?
7. What is a “standup” and what should you report about your work?
8. What is the role of a “Scrum master” in a project? What about the “Product owner”?
9. Explain the following metrics/charts: sprint velocity, burndown chart
10. What is a Scrum board? Have you used any software tools for your team’s Scrum board?

React

1. What is React? Is it a library or framework? What’s the difference between those?
2. What does it mean to be component-based? What does a component represent?
3. Tell me how you would start up a new React project?
4. How would you create a component?
5. What are “props”? What is state? What data should be put in which?
6. What is the lifecycle of a component?
7. What is the difference between a functional and a class component?
8. What is the purpose of a container component? What about a higher-order component?
9. What is routing and how would you do routing in React?
10. What is JSX? What does it compile into? How to include JS code in JSX?
11. How do you handle events in React?
12. Does React have 1-way or 2-way data binding and data flow?
13. If a parent component has data that a child component needs to access, what should you do?
14. If you have state in two child components that a parent component needs access to, what is a good solution for that?
15. How do you do conditional rendering?
16. What should you remember to include as a prop for lists of elements?
17. What is the Flux design pattern?
18. What are actions? Reducers? Store?
19. What are some pros/cons of using TypeScript in a React application?
20. Why is immutable state important?
21. How would you handle forms and submitting forms with React?