◆ Core Angular Concepts

1. Explain Angular's component lifecycle hooks. Which ones do you use most often and why?
2. How does Angular's change detection work? Can you explain the difference between **Default** and **OnPush** strategies?
3. What are Angular Zones and how does NgZone work?
4. Explain the difference between **Template-driven forms** and **Reactive forms**. Which one do you prefer and why?
5. What is **ViewEncapsulation** in Angular? Explain the differences between Emulated, ShadowDom, and None.
6. Can you explain how **dependency injection** works in Angular? How do providedIn: 'root' vs feature module vs component-level differ?
7. What's the difference between @Input(), @Output(), and @ViewChild() decorators?

◆ Advanced Topics

8. How do you optimize Angular applications for performance? (e.g., lazy loading, trackBy, pure pipes, OnPush)
9. How would you structure a large-scale Angular application?
10. How do you handle **state management** in Angular? Compare NgRx, Akita, NGXS, and simple services with RxJS.
11. What are **resolvers** in Angular routing, and when would you use them?
12. Explain **route guards** (CanActivate, CanDeactivate, etc.).
13. What's the role of **interceptors** in Angular's HTTPClient? Can you give a real-world use case?
14. How do you handle **multi-language (i18n)** in Angular applications?
15. How do you debug memory leaks in Angular applications?

◆ RxJS & Async Programming

16. Explain the difference between **Subject**, **BehaviorSubject**, **ReplaySubject**, and **AsyncSubject**.
17. How do you cancel HTTP requests in Angular using RxJS?

18. What’s the difference between mergeMap, switchMap, concatMap, and exhaustMap?
19. Can you explain the difference between Observable and Promise? Why would you prefer one over the other?
20. How do you implement error handling with RxJS operators?

◆ Testing

21. How do you unit test Angular components that have @Input() and @Output()?
22. How do you mock HTTP calls in Angular unit tests?
23. Explain the difference between **shallow testing** and **integration testing** in Angular.
24. How do you test asynchronous code in Angular with fakeAsync and tick()?
25. Have you used E2E testing tools like Cypress or Playwright with Angular? How do they compare with Protractor?

◆ Best Practices & Architecture

26. How do you decide when to use standalone components (Angular 15+) vs traditional NgModules?
27. How do you prevent cyclic dependencies in large Angular projects?
28. How do you manage shared modules vs core modules in Angular?
29. How do you ensure accessibility (a11y) in Angular apps?
30. What strategies do you use for secure Angular applications? (e.g., XSS, CSRF, JWT handling)

◆ Real-World Scenarios

31. Your Angular app is loading very slowly — how would you identify the root cause?
32. How would you implement a role-based access control system in Angular?
33. How would you handle offline support in Angular (e.g., service workers, PWA)?
34. How do you migrate a legacy AngularJS (1.x) app to modern Angular?
35. How do you handle version upgrades in a large Angular codebase?

✅ For a **senior developer interview**, expect follow-up questions like:

- *“Can you show me with code?”*
- *“What trade-offs do you consider when choosing NgRx vs services with RxJS?”*
- *“How would you debug production issues without impacting users?”*