



"Huge timesaver. Worth the money"



"It's an excellent tool"



"Fantastic catalogue of questions"

your next tech interview with

Explore our carefully curated catalog of interview essentials covering full-stack, data structures and algorithms, system design, data science, and machine learning interview questions

Start preparing now

Web & Mobile Dev

Data Structures & Algorithms

System Design

Machine Learning & Data Science

ADO.NET

Agile & Scrum

Android

Angular

AngularJS

ASP.NET

ASP.NET MVC

ASP.NET Web API

AWS

Azure

C++

C#

Cosmos DB

CSS

Dependency Injection

DevOps

Django

Entity Framework

Express.js

Flutter

GIT

Golang

GraphQL

HTML5

Ionic

Java

JavaScript

jQuery

Kotlin

Laravel

LINQ

MongoDB

.NET Core

Next.js

Node.js

Objective-C

OOP

PHP

PWA

Python

React

React Native

Reactive Programming

Reactive Systems

Redis

Redux



Ruby



Ruby on Rails



Rust



Spring



SQL



Swift



T-SQL



Testing



TypeScript



UX Design



Vue.js



WCF



Web Security



WebSockets



WPF



Xamarin



52 OOP interview questions

Only coding challenges ☐

Reset progress

Topic progress: 0%

OOP Fundamentals

1. What is **Object-Oriented Programming (OOP)**?
2. What is the difference between **procedural** and **Object-Oriented** programming?
3. What is **encapsulation**?
4. What is **polymorphism**? Explain **overriding** and **overloading**.
5. What is **inheritance**? Name some **types of inheritance**.
6. What is an **abstraction**? Name some **abstraction techniques**.
7. What is a **class** in OOP?
8. What is an **object** in OOP?
9. How do **access specifiers** work and what are they typically?
10. Name some ways to **overload** a **method**.
11. What is **cohesion** in OOP?
12. What is **coupling** in OOP?

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Question

☐

Class Relationships and Design

13. What is a **constructor** and how is it used?
14. Describe the concept of **destructor** or **finalizer** in OOP.

Question

☐

Question

☐

15. Compare **inheritance** vs. **mixin** vs. **composition**.

Question

☐

16. Explain the concept of an **interface** and how it differs from an **abstract class**.



Question

☐

17. Can a **class** have **multiple parents** in a **single-inheritance** system?



Question

☐

18. How would you design a **class** to prevent it from being **subclassed**?



Question

☐

19. Explain the '**is-a**' vs '**has-a**' relationship in OOP.



Question

☐

20. Explain how **aggregation** relationship is represented in OOP.



Question

☐

21. What is **method overriding**, and what rules apply to it?



Question

☐

22. Describe the use of **static methods** and when they are appropriate.



Question

☐

Advanced OOP Concepts

23. What is **multiple inheritance** and what are some of its disadvantages?



Question

☐

24. Can you explain the '**diamond problem**' in **multiple inheritance**?



Question

☐

25. How does OOP languages support **polymorphism** under the hood?



Question

☐

26. What are **generics**, and how can they be useful in **OOP**?



Question

☐

27. Explain the concept of **object composition** and its benefits.



Question

☐

28. What is **Liskov Substitution Principle** (LSP)? Provide some examples of **violation** and **adherence**.



Question

☐

29. What is the **dependency inversion principle**?



Question

☐

30. How can the **open/closed principle** guide object-oriented design?



Question

☐

31. Describe how the **Interface Segregation Principle** affects system design.



Question

☐

32. What is a **mixin**, and how does it differ from traditional **inheritance**?



Question

☐

Practical Use and Patterns

33. How would you refactor a **class** that has too many **responsibilities**?



Question

☐

34. Describe a **singleton pattern** and discuss its **pros** and **cons**.



Question

☐

35. What is a **factory method**, and when should it be used?



Question

☐

36. Explain the **builder pattern** and where you might apply it.



Question

☐

37. What is the **prototype pattern**, and how does it relate to **OOP**?



Question

☐

38. When would you use the **Adapter pattern**?



Question

☐

39. Can you explain the use of the **Decorator pattern**?



Question

☐

40. Describe the **Observer pattern** and a scenario in which you might use it.



Question

☐

41. What are the advantages of using the **Command pattern**?



Question

☐

42. How does the **Strategy pattern** provide flexibility in objects?



Question

☐

OOP Best Practices

43. What are some common OOP design **anti-patterns**?



Question

☐

44. How do you ensure that your objects are properly **encapsulated**?



Question

☐

45. Name some techniques for reducing **coupling** between **classes**.



Question

☐

46. How does **immutability** help in **object-oriented design**, and how can it be implemented?



Question

☐

47. What tools or techniques would you use to **document** an object-oriented design?



Question

☐

48. How do you address **circular dependencies** in an OOP system?



Question

☐

49. Explain how to apply **unit testing** to object-oriented code.



Question

☐

50. What strategies can be used to safely refactor **legacy object-oriented code**?



Question

☐

51. How can the principles of OOP help in achieving a **modular** and maintainable **codebase**?



Question

☐

52. How do you balance the use of OOP principles with **performance considerations** in a system design?



Question

☐

Unlock interview insights

Get the inside track on what to expect in your next interview. Access a collection

Track progress

Simple interface helps to track your learning progress. Easily navigate through the wide range of questions and

Save time

Save countless hours searching for information on hundreds of low-quality sites designed to drive traffic and

of high quality technical interview questions with detailed answers to help you prepare for your next coding interview.

focus on key topics you need for your interview success.

make money from advertising.

Land a six-figure job at one of the top tech companies



Ready to nail your next interview?

your
and get

Kickstart your prep