TypeScript Function Interview Questions (Basic to Advanced)

■ Basic Level
1. What is a function in TypeScript?
2. How do you define a function in TypeScript?
3. What is the difference between function declaration and function expression?
4. What is the return type of a function? How do you define it in TypeScript?
5. What happens if you don’t specify a return type?
6. What are optional parameters in TypeScript?
7. What are default parameters?
8. Can you explain rest parameters in TypeScript?
9. How is type inference applied in function return types?
10. What does the void return type mean?

■ Intermediate Level
11. What is the difference between function type and function interface?
12. How do you define a function type in TypeScript?
13. What are arrow functions and how are they different from normal functions?
14. How does `this` behave differently in arrow functions vs. regular functions?
15. What is the purpose of never as a return type?
16. What is function overloading in TypeScript?
17. How do you implement function overloading? Give an example.
18. What is the difference between optional and rest parameters?
19. How can you create a function that accepts any number of arguments?
20. Can a function in TypeScript return another function?

■ Advanced Level
21. What are generic functions in TypeScript?
22. How do you define a generic function with constraints?
23. What’s the purpose of in a function definition?
24. Explain how type inference works in generic functions.
25. How do you restrict generic types to a specific shape (like { id: number })?
26. What is the difference between T extends U and keyof T in function generics?
27. Can you create a higher-order function in TypeScript?
28. What are callback functions? How do you type them?
29. How do you type a function that returns a Promise?
30. How do you handle async/await in TypeScript with proper types?
31. What are function signatures in interfaces?
32. How can you use type aliases for function definitions?
33. What is the difference between using interface and type for a function type?
34. How do you type a function that takes another function as a parameter?
35. How to define a curried function type in TypeScript?

■ Real-World / Scenario-Based Questions
36. How do you ensure a function parameter must be an object with specific keys?
37. How would you type a function that takes both a string or a number?
38. How do you type a function that can return different types based on conditions?
39. How can you type a function that accepts both sync and async callbacks?
40. How can you ensure that a function only accepts a subset of an interface as parameter?
41. How do you type a function that modifies an array of generic type elements?
42. How can you use utility types (like Partial, Pick, Omit) inside function types?
43. What’s the difference between Function, () => void, and (arg: any) => any?

44. How do you type a function that throws an error?
45. How do you define a constructor function in TypeScript?

■ Tricky / Expert-Level Questions

46. What is type narrowing and how does it affect function logic?
47. How can you use discriminated unions in function parameters?
48. How do you type a recursive function?
49. What is declaration merging and how can it affect function types?
50. How do you type a function that returns a tuple?
51. How can you type a function overload that depends on generic type conditions?
52. What is ReturnType<>, and how can you use it for function types?
53. What is Parameters<>, and how can it help in reusing function types?
54. How do you extract argument types from an existing function type?
55. How do you define a call signature with optional generics?

■ Bonus Practical Examples

- Write a function that swaps the values of two generic variables.
- Write a function that merges two objects using generics.
- Write a function that filters an array of any type using a callback.
- Create a function that accepts another function and logs its execution time.
- Write a function type that describes a CRUD handler interface.