Node.js Interview Questions

1. **What is Node.js?**
   Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code outside a web browser.

2. **How does Node.js work?**
   Node.js uses an event-driven, non-blocking I/O model and the V8 JavaScript engine, making it efficient and lightweight.

3. **What is the V8 engine?**
   V8 is Google's open-source JavaScript engine that compiles JavaScript directly into native machine code for improved performance.

4. **Why is Node.js single-threaded?**
   Node.js is single-threaded to efficiently handle concurrent requests using asynchronous I/O and an event loop.

5. **What is the Event Loop in Node.js?**
   The event loop allows Node.js to handle multiple tasks concurrently by executing callbacks in the background and managing asynchronous operations.

6. **What is the difference between asynchronous and synchronous in Node.js?**
   - **Asynchronous**: Non-blocking operations, allowing other tasks to continue executing.
   - **Synchronous**: Blocking operations that pause execution until the task is complete.

7. **What are callbacks in Node.js?**
   Functions passed as arguments to other functions, executed after the main function finishes.

8. **What is the difference between process.nextTick() and setImmediate()?**
   - process.nextTick(): Executes after the current operation, before the event loop continues.
   - setImmediate(): Executes in the next iteration of the event loop.

9. **What are streams in Node.js?**
   Streams are data-handling objects used for reading or writing data continuously. Types include readable, writable, duplex, and transform streams.

10. **What is a Buffer in Node.js?**
    Buffers are used to handle binary data in Node.js, often when reading from or writing to streams.

11. **Explain the concept of middleware in Node.js.**
    Middleware functions are executed sequentially during request-response cycles, performing tasks like logging, authentication, and data parsing.

Complete CoDInG

12. **What is the package.json file?**
A metadata file in Node.js that holds information about the project, dependencies, and scripts.

13. **What is NPM?**
NPM (Node Package Manager) is a tool to install, manage, and share Node.js packages and dependencies.

14. **What is the difference between npm install and npm install --save?**
--save ensures the package is added to dependencies in package.json. (In NPM 5 and above, --save is the default behavior.)

15. **What is the node_modules folder?**
A directory where installed dependencies are stored.

16. **Explain require and import in Node.js.**
    - **require**: CommonJS syntax for importing modules.
    - **import**: ES6 module syntax (used with type: module in package.json).

17. **What are the core modules in Node.js?**
Examples include fs (File System), http, path, os, events, util, etc.

18. **How does Node.js handle concurrency?**
Node.js handles concurrency through the event loop and asynchronous I/O operations.

19. **What is the global object in Node.js?**
An object similar to window in the browser, providing global variables and functions like __dirname and setTimeout.

20. **What are child processes in Node.js?**
Child processes allow you to run system commands and spawn new Node.js processes using the child_process module.

21. **What is the cluster module in Node.js?**
The cluster module helps create multiple worker processes to take advantage of multi-core systems.

22. **What are the benefits of using Node.js?**
    - Fast execution
    - Non-blocking I/O
    - Scalability
    - Unified language for front-end and back-end (JavaScript)

23. **What is the difference between fs.readFile() and fs.createReadStream()?**
    - **fs.readFile()** reads the entire file into memory.
    - **fs.createReadStream()** reads data in chunks, suitable for large files.

24. **What is the purpose of the os module in Node.js?**
It provides operating system-related utility methods and properties.

Complete CODING

25. **Explain the util module in Node.js.**
The util module provides utility functions like promisify and inherits.

Express.js Interview Questions

26. **What is Express.js?**
Express.js is a minimal, flexible Node.js web application framework for building APIs and web applications.

27. **Why use Express.js?**
It simplifies routing, middleware integration, and API creation, speeding up development.

28. **What is routing in Express.js?**
Routing defines how the server responds to various HTTP requests (GET, POST, etc.) for specific URLs.

29. **What is the role of middleware in Express.js?**
Middleware functions execute before, during, or after request handling, used for logging, authentication, and parsing.

30. **What is app.use() in Express?**
app.use() mounts middleware functions in the application.

31. **What is the difference between app.get() and app.post()?**
    - app.get() handles GET requests.
    - app.post() handles POST requests.

32. **What are the different HTTP methods supported by Express?**
GET, POST, PUT, PATCH, DELETE, OPTIONS, and HEAD.

33. **How do you handle errors in Express.js?**
Using error-handling middleware with four parameters: (err, req, res, next).

34. **What is req.params in Express?**
It contains route parameters. Example: /user/:id -> req.params.id.

35. **What is req.query in Express?**
It contains query parameters. Example: /search?name=abc -> req.query.name.

36. **How can you parse incoming JSON data in Express?**
Using express.json() middleware.

37. **What is next() in Express middleware?**
next() passes control to the next middleware function.

38. **What are templating engines in Express?**
Tools like EJS, Pug, or Handlebars that render dynamic HTML pages.

39. **What is res.send() in Express?**
Sends a response of various types (string, object, etc.).

Complete CODING

40. **What is res.json() in Express?**
    Sends a JSON response.

41. **What is the difference between res.send() and res.json()?**
    res.send() can send any type, while res.json() specifically sends JSON.

42. **What is the purpose of the cookie-parser middleware?**
    To parse cookies from HTTP requests.

43. **How do you secure an Express app?**
    - Use HTTPS
    - Sanitize inputs
    - Use Helmet for security headers

44. **What is res.redirect() in Express?**
    Redirects a request to a different URL.

45. **How do you handle file uploads in Express?**
    Use libraries like multer for multipart/form-data handling.

46. **What is the purpose of express.Router()?**
    To create modular and mountable route handlers.

{Complete CODING}