6. What is callback hell, and how can it be mitigated in Node.js?
Callback hell refers to deeply nested callbacks that can make code hard to read and maintain. It can be mitigated using techniques like named functions, Promises, or async/await.

7. What is the 'require' function, and how is it used in Node.js?
The 'require' function is used to import modules in Node.js. It loads and returns the exported object of the requested module.

8. Explain the difference between 'require' and 'import' for loading modules in Node.js.
'require' is the CommonJS module system used in Node.js, while 'import' is an ES6 feature used in modern JavaScript. They serve similar purposes but have different syntax.

9. What are the core modules in Node.js, and how do you use them?
Core modules are built-in modules provided by Node.js. You can use them by simply requiring the module name, e.g., 'fs' for file system operations or 'http' for creating web servers.

10. How can you handle errors in Node.js applications?
Errors in Node.js can be handled using try-catch blocks, error event listeners, or by passing errors as the first argument in callbacks.

11. What is the purpose of the 'util' module in Node.js?
The 'util' module provides utility functions for working with JavaScript objects, such as `util.inherits` and `util.promisify`.

12. How can you achieve file I/O operations in Node.js?
File I/O operations can be performed using the 'fs' (file system) module, which provides methods for reading and writing files.