Node.js

Q1. When a javaScript function is invoked (called) in Node, where is a new frame placed?

- [x] the call stack
- [ ] the event loop
- [ ] the poll phase
- [ ] the events queue

Explanation: From javascripttutorial: reference

Q2. Which of the following is a core module in Node?

- [ ] webpack
- [x] crypto
- [ ] request
- [ ] chalk

Explanation: From flaviocopes docs: reference

Q3. Which of the following Buffer class methods returns an uninitialized buffer?

- [x] allocUnsafe
- [ ] concat
- [ ] from
- [ ] alloc

Explanation: From official docs: reference

Q4. Which of the following modules is NOT a built-in module in Node?

- [x] ftp
- [ ] events
- [ ] dgram
- [ ] http2

Explanation: From flaviocopes docs: reference

Q5. Which fs module method can be used to read the content of a file without buffering it in memory?

- [ ] read
- [ ] readFile
- [x] createReadStream
- [ ] readFileSync

Explanation: *From official docs: reference To minimize memory costs, when possible prefer streaming via fs.createReadStream().*

Q6. Which of the following DNS module methods uses the underlying OS facilities and does not necessarily perform any network communication?

- [x] lookup
- [ ] resolve
- [ ] resolve4
- [ ] reverse

Explanation: From official docs: reference

Q7. How do you check that a value is a date object in Node?

- [x] util.types.isDate(value)
- [ ] assert.isDate(value)
- [ ] console.isDate(value)
- [ ] util.date(value)

Explanation: From official docs: reference

Q8. Can you create an https web server with Node.js?

- [ ] no, there are no modules supporting it yet
- [x] yes, with the https or http2 modules
- [ ] yes, through the path module
- [ ] yes, with the http module

Explanation: From official docs: reference

Q9. What is the Api that is designed to insulate Addons from changes in the underlying JavaScript engine?

- [ ] A-API
- [ ] Z-API
- [x] N-API
- [ ] X-API

Explanation: From official docs: reference

Q10. Which CLI option can you use to debug a node script in Chrome DevTools?

- [ ] --dev-tools
- [x] --inspect
- [ ] --chrome
- [ ] --debug

Explanation: From official docs: reference

Q11. What command would you use to count the number of logical CPUs on the machine that is running Node?

- [ ] node -p "process.cpus"
- [ ] node -p "util.cpus().size"
- [ ] node -p "process.os.cpus"
- [x] node -p "os.cpus().length"

Explanation: From coderrocketfuel docs: reference

Q12. Which of the following is a method on the console object?

- [ ] exit
- [ ] test
- [x] time
- [ ] print

Explanation: From official docs: reference

Q13. Which object is used to manage the cache of required modules?

- [ ] global.cache
- [ ] module.cache
- [ ] process.cache
- [x] require.cache

Explanation: From official docs: reference

Q14. What is the command to silence all process warnings?

- [ ] node index.js --trace-warnings
- [x] node --no-warnings
- [ ] node -trace-warnings
- [ ] node index.js --no-warnings

Explanation: From official docs: reference

Q15. How can you use the promise API with a callback-based function such as child_process.exec?

- [ ] new Promise(child_process.exec())
- [ ] util.promisify(child_process.exec())
- [x] util.promisify(child_process.exec)
- [ ] new Promise(child_process.exec)

Explanation: From official docs: reference

Q16. Which of the following is NOT a Node repl command?

- [ ] .break
- [x] .history
- [ ] .editor
- [ ] .save

Explanation: From official docs: reference

Q17. Which statement is true when you run the code shown below?

```
require('child_process').fork('script.js');
```

- [ ] The forked process shares the event loop with the parent process
- [ ] A new VM instance is created and the two VM instances will be shared between the forked process and the parent process.
- [x] The forked process will have its own VM instance.
- [ ] The forked process shares the same VM thread with the parent process.

Explanation: From official docs: reference

Q18. If EventEmitter is in scope, which of the following lines of code will have an event emitter emitting a change event?

- [x] EventEmitter.emit('change');
- [ ] EventEmitter.new().emit('change');
- [ ] (new EventEmitter()).emit('change');
- [ ] new EventEmitter('change');

Explanation: *Because the EventEmitter is already in scope. No need to create new one.*

Q19. Which of the following objects is a stream

- [ ] process.uptime
- [x] process.stdout
- [ ] process
- [ ] Buffer

Explanation: *process.stdout is Buffer type.*

Q20. Which module variable holds the resolved absolute path of the current module file?

- [ ] `__pathname`

- [ ] `__location`
- [ ] `__flder`
- [x] `__filename`

Q21. If the child_process module methods are in scope, what is a current way to execute the command ps -ef using a child process?

- [ ] spawn("ps -ef")
- [x] exec("ps -ef")
- [ ] exec("ps", "-ef")
- [ ] fork("ps -ef")

Reference: From official docs: reference

Q22. Which console method can be used to print the stack trace to the point of its execution?

- [ ] stack
- [x] trace
- [ ] debug
- [ ] print

Q23. When you run JavaScript in a Node.js application, which of the following elements in a Node.js stack actually executes that JavaScript?

- [ ] the libuv library
- [ ] the c-ares library
- [x] the VM (like V8 or Chakra)
- [ ] the repl module

Q24. Looking at the code below, what does the console show?

```
const http = require('http');
 const hostname = '127.0.0.1'; const port = 3000;
 const server = http.createServer((req, res) => {
  res.statusCode = 200;  res.setHeader("Content-Type", "text/plain");  res.end("H
});
server.listen(port, hostname, () => { console.log(`server running at http://${hos
```

- [ ] server running at http://localhost:3000/
- [ ] server running at port 3000
- [ ] server running at http://localhost:4000/
- [x] server running at http://127.0.0.1:3000/

Explanation: From official docs: reference

Q25. What is the purpose of the path module?

- [x] to provide utilities to play with file and directory paths
- [ ] to provide utilities to add and remove files
- [ ] It is a retiring module.
- [ ] to provide utilities to test files

Explanation: From official docs: reference

Q26. How do you make an HTTP server object active and listen to requests on certain ports?

- [ ] server. start
- [ ] server.activate
- [x] server.listen
- [ ] server. run

Q27. What does the code shown below do?

```
const fs = require('fs'); const os = require('os');
const system = os.platform(); const user = os.userInfo().username;
fs.appendFile('hello.txt', `Hello ${user} on ${system}`, (err) => { if (err) thro
);
```

- [x] creates a text file hello.txt and appends customized text
- [ ] creates an image file
- [ ] console logs system information
- [ ] creates a file named data and append numbers

Q28. How do you start a Node application, if the entry file is indexjs?

- [ ] nodemon start
- [ ] start index.js
- [x] node index.js
- [ ] node start

Q29. What is the purpose of the file system (fs) module?

- [ ] to provide methods to work with requests and responses
- [x] to provide methods to work with files
- [ ] to provide methods to work with databases
- [ ] to find new file systems

Explanation: From official docs: reference

Q30. What is the Node LTS version?

- [ ] It is the current unstable version and is to be avoided.
- [ ] It is the version that will be retired soon.
- [ ] It is the version with the latest features.
- [x] It is the safest version for long-term support.

Q31. Which of the following is NOT a valid stream in Node?

- [x] process. stdinfo
- [ ] process. stdin
- [ ] process. stdout
- [ ] process. stderr

Q32. You have a script.js file with the single line of code shown here. What will be the output of executing script.js with the node command?

```
console.log(arguments);
```

- [x] ReferenceError: arguments is not defined
- [ ] an empty string
- [ ] undefined
- [ ] an object representing an array that has five elements

Explanation:: Reference Article The output of executing console.log(arguments); in Node.js will be a ReferenceError: arguments is not defined. The arguments object is not available in Node.js, as it is a specific feature of the browser JavaScript environment. In browsers, the arguments object is an array-like object that contains all of the arguments that were passed to a function. However, in Node.js, there is no arguments object, and the only way to access the arguments that were passed to a function is to explicitly declare them in the function's parameter list.

Q33. Which choice is not a valid method on event emitters?

- [x] start
- [ ] on
- [ ] once
- [ ] off

Q34. Which special object is an instance of EventEmitter?

- [x] process
- [ ] Buffer
- [ ] root
- [ ] require

Reference

Q35. What is the command to get a list of available commands for Node.js?What is the command to get a list of available commands for Node.js?

- [ ] node index.js -x
- [ ] node -v
- [x] node -h
- [ ] node index.js -h

Q36. When a request event is received in the HTTP module, what is the type of the first argument passed to that event, usually named req?

- [x] http.IncomingMessage
- [ ] http.ServerRequest
- [ ] http.ClientRequest
- [ ] http.ServerResponse

Q37. What are the arguments passed to the module wrapper function?

- [ ] `exports, __filename, __dirname`
- [ ] `exports, process, require, module, __filename, __dirname`
- [ ] `exports, module, __filename, __dirname`
- [x] `exports, require, module, __filename, __dirname`

Q38. Which library provides Node.js with the event loop?

- [ ] V8
- [ ] c-ares
- [x] libuv
- [ ] events

Q39. What does the .node file extension represent?

- [ ] a C++ file that can have a .node extension and that Node will be able to execute directly.
- [x] a C++ Addon file that is built with node-gyp
- [ ] a JSON file that can have a .node extension as well as the .json extension
- [ ] a JavaScript file that can have a .node extension as well as the .js extension

Q40. What can you export with module.exports?

- [ ] only objects.
- [ ] only functions
- [ ] only variables and arrays
- [x] functions, objects, arrays, or anything you assign to the module

Q41. Which core module in Node can you use to take advantage of multicore systems?

- [ ] os
- [ ] util
- [x] cluster
- [ ] net

Q42. Which core Node module has wrappers for OpenSSL methods?

- [ ] SSL
- [ ] hash
- [x] crypto
- [ ] TLS

Q43. Which line imports a promise-based version of the readFile method?

- [x] const { readFile } = require(fs).promises
- [ ] const { readFile } = require(fs)
- [ ] const { readFilePromises: readFile } = require(fs)
- [ ] const { readFile } = require(promises)

Q44. According to the rules of semantic versioning, what does a release incrementing the third number in an npm version string communicate to users about the release changes?

- [ ] Changes are not backwards compatible.
- [ ] Changes might not be backward compatible and might break existing code.
- [x] Changes are just bug fixes and no new features were added.
- [ ] Changes will add new functionality but will not break any existing code.

Q45. What does REPL stand for?

- [ ] run, examine, put, loop
- [x] read, eval, print, loop
- [ ] run, edit, print, loop
- [ ] read, extend, print, loop

Q46. Which file does node-gyp use to read the build configuration of a module?

- [ ] .gyprc
- [x] binding.gyp
- [ ] gyp.json
- [ ] package.gyp

Q47. Which core module in Node can you use for testing?

- [ ] chai
- [ ] jest
- [x] assert
- [ ] mocha

Q48. Which core module in Node provides an API to register callbacks to track asynchronous resources created inside a Node.js application?

- [ ] cluster
- [x] async_hooks
- [ ] dgram
- [ ] inspector

Explanation: From official docs: reference

Q49. Which Node.js module should you use when you need to decode raw data into strings?

- [ ] buffer
- [ ] util
- [x] string_decoder
- [ ] string_buffer

Refrence

Q50. Which global object acts like a bridge between a Node script and the host operating system?

- [ ] v8
- [ ] env
- [x] process
- [ ] child_process

Explanation: _process is an global object and act like a bridge, the others aren't

1. source
2. source

Q51. Which statement is true about Node.js and threads?

- [ ] Every Node process runs in a single thread, and all the I/O work is run in that same thread.
- [ ] Every Node process gets four threads that it can share between its JavaScript VM and the event loop.
- [x] The event loop is single-threaded, but a JavaScript VM can use multiple threads.
- [ ] JavaScript execution in Node.js is single-threaded, but I/O operations are executed using multiple threads.

Explanation: *https://www.geeksforgeeks.org/why-node-js-is-a-single-threaded-language/*

Q52. Which statement about event emitters is false?

- [x] Event names must be camelCase strings.
- [ ] The emit method allows a arbitrary set of arguments to be passed to the listener functions.
- [ ] Any values returned by the listeners for an emitted events are ignored.

- [ ] When an event emitter object emits an event, all of the functions attached to that specific event are called synchronously.

Q53. Which core module in Node can you use to compile and run JavaScript code in a sandbox environment?

- [ ] sandbox
- [ ] buffer
- [x] vm
- [ ] v8

Q54. How would you determine the number of cluster instances to start when using the cluster module?

- [ ] const numInstances = cluster.instances().length;
- [ ] const numInstances = cluster.instances();
- [x] const numInstances = require('os').cpus().length;
- [ ] const numInstances = process.cpus().length;

Explanation: *From official docs: https://nodejs.org/api/cluster.html#cluster_cluster*

Q55. You have to read a large text file, replace some words in it, and write it back to a new file. You know that the memory on your target system is limited. What should you do?

- [ ] Use regular expressions directly on the file.
- [ ] Use Promises and async/await to offload the task to libuv.
- [ ] Copy the file into a database and perform the operations there.
- [x] Use readline together with streams to read and transform and write the file contents line by line.

Explanation: *From official docs: https://nodejs.org/api/readline.html#readline_example_read_file_stream_line_by_line*

Q56. Which choice is not a Node global object?

- [ ] process
- [x] exports
- [ ] setTimeout
- [ ] Buffer

Explanation: `exports` may appear to be global but is not. Refrence

Q57. What is the correct way to pipe a readable stream and a writable stream?

- [x] readableStream.pipe(writableStream)
- [ ] readableStream.on(pipe, writableStream)
- [ ] writableStream.pipe(readableStream)
- [ ] writableStream.on(pipe, readableStream)

Q58. How can you convert path segments into a string using the platform-specific separator as a delimiter?

- [ ] path.concat
- [x] path.join
- [ ] path.format
- [ ] path.parse

Explanation: From official docs: reference

Q59. What is the purpose of N-API?

- [ ] to allow users to make requests to the server
- [x] to insulate Addons from changes in the underlying JavaScript engine
- [ ] to execute multi-threaded code in the Node environment
- [ ] to provide a quick way for users to create REST APIs

Q60. What is a process object and its role?

- [ ] a locally scoped object that provides information about the current node process
- [ ] a global object that provides information about files
- [ ] a global object that provides information about the database
- [x] a global object that provides information about the current node process

Q61. What will this code log to the console?

```
// File: person.js
exports.name = "Jane";

// File: index.js
const person = require('./person.js');
console.log(person);
```

- [ ] `{'Jane'}`
- [x] `{ name: 'Jane' }`
- [ ] `{}`
- [ ] `Jane`

Q62. What will this code log to the console?

```
// File: person.js
exports = "John";

// File: index.js
const person = require('./person.js');
console.log(person);
```

- [ ] `John`
- [ ] `Undefined`
- [ ] `{'John'}`
- [x] `{}`

Q63. Is it possible to write tests in Node.js without an external library?

- [x] yes, through the assert module
- [ ] yes, through the debugger module
- [ ] yes, through the console module
- [ ] no

From the article: Making a Testing Framework in Node.js (Without any External Libraries)

Q64. Which assert module method is usually used to test the error-first argument in callbacks?

- [ ] fail
- [ ] doesNotThrow

- [ ] deepStrictEqual
- [x] ifError

Q65. Which choice is not a method on the util module?

- [ ] promisify
- [x] asyncify
- [ ] types
- [ ] callbackify

Q66. Which choice is not a subclass of the Error class?

- [x] GlobalError
- [ ] TypeError
- [ ] RangeError
- [ ] AssertionError

Q67. What is Node built on?

- [ ] Python
- [x] V8 JavaScript engine
- [ ] PHP
- [ ] c

Refrence

Q68. How does it affect the performance of a web application when an execution path contains a CPU-heavy operation, such as calculating a long Fibonacci sequence?

- [ ] As Node.js is asynchronous, this is handled by a libuv and a threadpool. The performance will not notably degrade.
- [ ] As the application code runs asynchronously within a single thread, the execution will block, accepting no more requests until the operation is completed.
- [ ] As Node.js is asynchronous, this is handled by a threadpool and the performance will not notably degrade.
- [x] The current thread will block until the execution is completed and the operating system will spawn new threads to handle incoming requests. This can exhaust the number of allowed threads (255) and degrade performance over time.

Q69. What is used for parsing and running Javascript in Node.js?

- [ ] EventLoop
- [ ] Libuv
- [x] Google V8
- [ ] Express.js

Refrence

Q70. What is the importance of having good practices around status code in your response?

- [x] It indicates success or failure to the client and helps with testing.
- [ ] It is not important to have good practices regarding status codes
- [ ] Response codes are the only way you can tell what is happening on the server.
- [ ] It contains information about the current performance of the server.

Q71. How can ECMAScript modules be used natively in Node?

- [ ] ECMAScript modules cannot be used natively in Node.
- [x] ECMAScript modules can be used natively in Node with the .mjs file extension
- [ ] ECMAScript modules can be used natively in Node only by using a compiler like Babel.
- [ ] ECMAScript modules can be used natively in Node only by using a bundle like webpack.

Reference

Q72. When exploring the Node documentation's features, what are the stability ratings?

- [x] They are an indication of the stability of Nodejs modules and usage recommendations.
- [ ] They tell if a feature is ES6 compliant.
- [ ] They are a Node command to validate stability of your code.
- [ ] They tell if a feature is LTS (Long Term Supported).

Q73. Which DNS module method uses the underlying OS facilities and does not necessarily perform any network communication?

- [ ] resolve
- [ ] reverse
- [x] lookup
- [ ] resolve4

Q74. When you `require(something)` , where will Node.js attempt to `resolve(something)` ?

- [ ] the local .modules folder, then the parents' node_modules folder
- [x] the local node_modules folder, then the parents' node_modules folder
- [ ] the .modules folder under the home directory
- [ ] a "something.js" file or a "something" folder, which exist on the same level as the requiring file

Q75. An external library has its own codebase and license. It is not managed by the Node.js core team. Which choice is an external library that Node.js uses?

- [ ] net
- [x] openssl
- [ ] cluster
- [ ] events

Reference

Q76. What is the main purpose of the package-lock.json file?

- [ ] to be a system file
- [x] to provide an exact, single representation of the dependency tree
- [ ] to serve as a module to export dependencies
- [ ] to be a log for the application

Q77. What response will you get when you send a get requests to the server with this code?

```
const http = require('http');
const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

- [ ] `server running at http://127.0.0.1:3000`
- [ ] `server running at port 3000`
- [ ] `server running at http://localhost:3000/`
- [ ] `server running at http://localhost:4000/`
- [✓] `Hello World`

Refrence

Q78. What is the primary function of the npm command in Node.js development?

- [ ] To install and manage Node.js versions.
- [✓] To manage packages and dependencies for Node.js projects.
- [ ] To run JavaScript code in the browser.
- [ ] To create and manage databases for Node.js applications.

Explanation: The npm command is the Node Package Manager and is primarily used to manage packages and dependencies for Node.js projects. It allows developers to install, update, and manage packages from the npm registry.

Q79. In Node.js, how can you handle asynchronous operations effectively to avoid callback hell?

- [ ] Use global variables to share data between asynchronous functions.
- [ ] Avoid using asynchronous functions altogether.
- [ ] Use synchronous functions to ensure order of execution.
- [✓] Use Promises, async/await, or libraries like async.js to manage asynchronous operations in a more structured way.

Explanation: To handle asynchronous operations effectively and avoid callback hell, it's recommended to use Promises, async/await, or libraries like async.js. These techniques provide a more structured and readable way to work with asynchronous code.

Q80. Which core module in Node.js can be used to create web servers?

- [ ] fs (File System)
- [ ] http (HTTP Client)
- [ ] url (URL Parsing)
- [✓] http (HTTP Server)

Explanation: The http core module in Node.js can be used to create web servers. It provides the necessary functionality to handle HTTP requests and responses, making it possible to create web applications and APIs.

Q81. What is the purpose of the os module in Node.js?

- [ ] To work with the file system and perform I/O operations.

- [ ] To create and manage child processes.
- [x] To provide information about the host operating system, such as CPU, memory, and network interfaces.
- [ ] To parse and manipulate URLs.

Explanation: The os module in Node.js is used to provide information about the host operating system. It offers functions to access details about the CPU, memory, and network interfaces, making it useful for system-related tasks.

Q82. How can you serve static files, such as HTML, CSS, and images, in a Node.js web application?

- [ ] Use the url module to serve static files.
- [ ] Embed the static files directly into JavaScript code.
- [x] Use middleware like express.static in combination with the Express.js framework to serve static files.
- [ ] Write custom JavaScript functions to serve each static file individually.

Explanation: To serve static files in a Node.js web application, you can use middleware like express.static in combination with the Express.js framework. This middleware simplifies the process of serving HTML, CSS, images, and other static assets.

Q84. How can you terminate a Node.js application programmatically?

- [ ] Using the process.terminate() method
- [ ] Sending a SIGSTOP signal
- [ ] Using the exit() method
- [x] Sending a SIGINT signal (e.g., with process.kill(process.pid, 'SIGINT')) or calling process.exit()

Explanation : To terminate a Node.js application, you can either send a SIGINT signal or call process.exit() programmatically.

Q85. What is the purpose of the child_process module in Node.js?

- [ ] To create and manage child processes in a separate thread for parallel execution.
- [ ] To provide access to child elements of JSON data.
- [ ] To handle file I/O operations in a separate thread.
- [x] To create and manage child processes for running external commands or scripts in a separate process.

Explanation : The child_process module is used for creating and managing child processes to run external commands or scripts independently.

Q86. What is the primary purpose of the cluster module in Node.js?

- [ ] To manage user authentication and authorization.
- [x] To take advantage of multicore systems by forking multiple Node.js processes and distributing the workload among them.
- [ ] To create clusters of data storage for efficient data handling.
- [ ] To provide clustering for network communication.

Explanation : The primary purpose of the cluster module in Node.js is to utilize multicore systems efficiently by forking multiple Node.js processes to distribute workloads.

Q87. Which method of the fs module in Node.js is used to check if a file exists asynchronously?

- ☐ fs.existsSync
- ✓ fs.access
- ☐ fs.existsSync
- ☐ fs.stat

Explanation: The fs.access method in Node.js is used to check if a file exists asynchronously. It is a recommended way to check for the existence of a file as it does not throw an error if the file doesn't exist, and it's more efficient than using fs.exists or fs.stat. If the file exists, the callback function will be called with no error. If the file doesn't exist, the callback will be called with an error.

Q88. Which core module in Node.js is used for network programming and creating network applications?

- ☐ http
- ☐ os
- ☐ fs
- ✓ net

Explanation: The net core module in Node.js is used for network programming and creating network applications. It provides the necessary functionality for creating both TCP and Unix socket servers and clients.

Q89. What is the purpose of the util.promisify method in Node.js?

- ☐ To create a new Promise object.
- ☐ To convert a function that returns a Promise into a callback-style function.
- ✓ To convert a callback-style function into a function that returns a Promise.
- ☐ To handle synchronous operations.

Explanation: The util.promisify method in Node.js is used to convert a callback-style function into a function that returns a Promise. It simplifies working with asynchronous functions by allowing you to use async/await syntax and Promise-based error handling.

Q90. Which Node.js module provides an interface for interacting with the file system, including reading and writing files?

- ☐ http
- ✓ fs
- ☐ net
- ☐ os

Explanation: The fs module in Node.js provides an interface for interacting with the file system. It allows you to perform various file-related operations, including reading and writing files, creating directories, and more.