1. Javascript
2. prototype chain and inheritance
3. Functie composition
4. DOM and BOM
5. event propagation
6. event. preventDefault.
7. event.stopPropagation),
8. toggle checkbox by button click
9. regular function vs arrow function
10. array vs set,
11. object freeze vs seal.
12. create a method to remove last property from object
13. Set vs WeakSet
14. IIFE purpose
15. webapis
16. strict mode
17. for in vs for of
18. static keyword
19. polymorphism
20. follow JS coding conventions
21. constructor syntax, methods
22. learn classes better
23. destructuring
24. deep copy an object
25. loop an object
26. remove object key
27. .reduce(): find longest string
28. remove nth element from array
29. variadic function to return sum and average of args
30. switch case, fall-through
31. generator function to yield even numbers endlessly
32. Hoisting
33. Template literal
34. default parameter
35. rest
36. spread operator
37. event loop
38. temporal dead zone
39. foreach vs map
40. currying
41. closure
42. pure function
43. higher order function
44. memoization
45. Promise, states
46. unary operator
47. JS Coding conventions (formatting)
48. notice argument types and return types of builtin functions

49. classes, constructor, methods (using properties in methods)
50. instance properties, this, static keyword
51. rest operator
52. spread operator
53. increment loop iterator by any number
54. difference between (e) => {e} and (e) => (e)
55. const val = ""apple""; const x = {val} - what is inside x?
56. destructuring syntax
57. closure
58. Promises
59. async/await
60. array/objects
61. filter/map/reduce
62. map vs forEach
63. optional chaining
64. generators
65. single thread vs multi thread
66. 1.Running JavaScript
67. a- Script tag
68. b- Link External file
69. c- Browser Console
70. d- With Nodejs
71. 2.Variables
72. 1.Declarations
73. var
74. let
75. const
76. 2.Scope
77. Global
78. Function
79. Block
80. 3.Hoisting
81. 3.Data Types and Data Structures
82. 1.Primitive Types
83. a-undefined
84. b-Boolean
85. c-Number
86. d-BigInt
87. e-String
88. f-symbol
89. 2.
90. null
91. Object
92. Function
93. 3.Data Structures
94. a-Array
95. b-Map/ Weak Map
96. c-Set/ Weak Set

97. d-Date
98. 4.Type Conversion
99. 1-Explicit Conversion (we can convert Data types of variable using Type Conversion)
100. 2-Implicit Conversion
101. 5.Equality
102. == VS === (Equality and Strict Equality Operators)
103. 6.Loops
104. 1- While
105. 2- do..while
106. 3- for
107. 4- Break/Continue
108. 5- for..in
109. 6- for..of
110. 7.Control Flow
111. 1- if..else
112. 2- switch
113. 3- try/catch/throw
114. 9.Functions
115. 1.Function Declarations
116. 2.Function Expression
117. 3.Calling Functions
118. 4.Parameters and Arguments
119. 5.Scope
120. 6.Arrow Functions
121. 1.Nested Functions
122. 2.Lexical Scoping
123. 3.IIFE
124. 4.Revealing Module Pattern
125. 2.Closure
126. 3.Currying
127. 4.this
128. Implicit Binding
129. Explicit Binding
130. New Binding
131. Lexical Binding
132. Default Binding
133. 5.Prototype
134. 6.Prototypal Inheritance
135. 7.Class
136. 8.Iterators
137. 9.Generators
138. 10.Event Loop
139. 11.Asynchronous JavaScript
140. 1.setTimeout
141. 2.setInterval
142. 3.callbacks
143. 4.promises

144. 5.async await
145. Callback,
146. promise
147. API
148. dynamic typing ,
149. TDZ,
150. states of a Promise,
151. currying use cases,
152. Object.seal vs Object.freeze,
153. JS coding conventions,
154. class constructor,
155. subclassing,
156. static,
157. destructuring , .
158. reduce(): largest element in array, count occurrences of a value in array,
159. find non-repeating elements in array, function that returns the sum of all arguments passed to it,
160. generator function to yield multiples of a given number
161. Escape sequence,
162. user defined datatypes,
163. Variable shadowing,
164. type coercion,
165. event delegation,
166. Weakset,
167. Weakmap,
168. flatMap,
169. proxy object,
170. nullish operator,
171. generator function to generate odd numbers,
172. memoization,
173. Promise methods,

NODEJS

174. express,
175. streams,
176. duplex & transform stream,
177. Buffer class,
178. process.nextTick,
179. event driven programming,
180. MVC,
181. child process,
182. spawn,
183. fork,
184. exec,
185. app.locals,
186. partials ,
187. app.all,
188. Req vs Resp (Request vs Response)
189. session management ,

190. explore node.js architecture,
191. core modules,
192. package-lock.json,
193. explore res methods,
194. Express.static,
195. types of middlewares
196. content negotiation,
197. BSON document,
198. profiler,
199. Try catch finally syntax
200. ,Type casting,
201. instanceOf,
202. Pure functions,
203. Generator function,
204. Arrow function,
205. IIFE,
206. HOF example,
207. Filter function,
208. Reduce function syntax,
209. Bind function syntax,
210. States of, promise,
211. Async await,
212. Event loop,
213. setImmediate,
214. IIFE purpose
215. eval
216. Promise.allSettled
217. Symbol
218. debouncing
219. currying use case
220. remove object key
221. .reduce(): sum of positive numbers
222. remove nth element from array
223. capitalise first letter
224. digits missing in an array
225. looping an object
226. switch fall-through
227. Function borrowing-call,apply,bind
228. New vs extends keyword
229. Process.nexttick
230. setImmediate
231. retain search query in input after searching
232. ES6 difference
233. Array methods
234. Map filter reduce flat
235. use strict
236. spread operator
237. rest operator

238. object deep copy
239. json
240. json methods
241. callback
242. promise
243. promise methods
244. async
245. await
246. sync, async
247. first class functions
248. higher order function
249. hoisting
250. optional chaining
251. undefined
252. not defined
253. scope chaining
254. lexical envt
255. block scope
256. closure
257. set timeout set interval
258. callback hell
259. event loop
260. prototypes
261. callstack
262. callback queue
263. microtask queue
264. loops
265. dom
266. dom manipulation
267. conditions
268. object and array
269. defer vs async
270. IIFE
271. currying
272. event listeners
273. event bubbling
274. event capturing
275. event propagation
276. event delegation
277. this
278. window
279. object methods (key)
280. styling console log
281. jquery
282. map and set
283. map vs foreach
284. execution context
285. indexof , typeof

286. instance of
287. nullish coalescing
288. object shorthand
289. diff b/w == & ===
290. js engine
291. pure functions
292. side effects
293. generators
294. call, apply, bind
295. class and constructor
296. class constructors
297. Var & Let
298. Validation
299. Data types
300. Events
301. Scop
302. Json
303. Dom
304. Query selector
305. Grid system
306. Break points
307. Container
308. Container- fluid
309. Scripting language
310. Dom manipulation methods
311. Typeof
312. Higher order fn
313. First order fn
314. Offset
315. Event listener
316. Settimeout
317. Array methods
318. InnerHTML
319. console.log(3+ +'3')
320. length of an object
321. const myValue = obj.myKey ?? null;
322. map and forEach
323. how to initialise a variable
324. Array.from
325. new Set()
326. not able to do a even number sum finding program
327. how to write an anonymous function
328. IIFE
329. Object.keys()
330. Object.values()
331. Object.entries()
332. Symbol
333. replace

334. concat
335. The differences between CONST and Object.Freeze
336. Deep freeze
337. Stages of promise
338. Implementation of promise
339. Using reduce find second smallest number from an array
340. Debouncing
341. Throttling
342. Generator function
343. closure
344. call
345. apply
346. bind
347. first class function
348. IIFE
349. Currying
350. indexOf
351. fetch
352. JS Coding conventions
353. memory management in IS (deallocation)
354. converting function to arrow function
355. understand asynchronous better
356. reduce to find largest element in a number array
357. syntax of the callback function in reduce
358. accessing object property in template literal
359. destructuring object and array
360. day is a number, if day is 1-5, print "weekday", or if 6 and 7, print "weekend", else print "invalid day"
361. switch syntax, fall-through
362. if-else if-else for the same problem
363. difference between =, == and === (use them properly in conditions)
364. generator that generates even numbers infinitely
365. arrow function vs regular function
366. call apply bind
367. Object freeze
368. Object.seal
369. Weakset
370. weakmap
371. event delegation
372. event propagation
373. e stoppropagation
374. e preventdefault
375. promise
376. allsettled
377. Error
378. first callback
379. http vs https
380. REPL

381. library vs framework
382. switch fall-through
383. type inference
384. ternary: converting conditional return to ternary
385. static keyword
386. Object.seal
387. subclassing, inheritance
388. looping object and printing keys and values
389. type coercion vs type casting
390. JIT
391. execution order when timers and callbacks are involved
392. counting occurrence using reduce
393. Template literals
394. Event delegation
395. Currying advantages
396. Array methods practicals
397. Object methods practicals
398. Options method
399. process.next Tick
400. setImmediate
401. fs.link fs.stat
402. app.locals app.all
403. Partials
404. Prototype pollution
405. Weak Set
406. 5. Proxy object
407. 6. Null vs undefined
408. .7 Memory allocation for undefined
409. Symbol
410. 9. Class: adding methods,
411. child class constructor
412. 10. Remove key from object
413. 1. Loop Through An Object
414. 12. for...in vs for...of
415. 13. Iterator, enumerator
416. 14. Execution order when timers and callbacks are involved
417. Ternary syntax
418. , ternary with return reduce:
419. longest string in array
420. Generator: yield multiples of a given number
421. Currying
422. Closure
423. Temporal Dead Zone
424. Object.seal
425. 22. Debouncing
426. 23. Class syntax
427. 24. Object syntax
428. 25. Generate array containing multiples of 7 between 1 and 100

26. Function to return average of two args
27. Swap values of two variables
28. TDZ (Temporal Dead Zone)
29. Pass by reference
30. Increment loop iterator by any number
31. Difference between (e) >={e} and (e) >= (e)
32. const val = "apple"; const x = {val} - what is inside x?
33. Destructuring syntax
34. Execution order when timers and callbacks are involved (referenced link)
35. Pass by reference
36. Execution context
37. Primitive Data Type vs Non-Primitive
38. String Coercion
39. Examples of higher-order functions
40. Factory Functions
41. Check a function whether palindrome or not
42. IIFE (Immediately Invoked Function Expression)
43. Benefits of IIFE
44. Prototypical inheritance
45. DOM manipulation
46. Object empty
47. Array length
48. Remove nth element from array
49. Variadic function to return sum and average of args
Primitive vs non-primitive data types - understand better
ClearTimeout
IFE purpose
remove key corresponding to the highest
countdown timer to count from 10 to 0
callback concept of event-driven programming
error properties
understand scopes better (const, var)
this in global scope
making object immutable
remove object keys corresponding to odd numbers
looping objects
filter strings from array
(e) = (e) vse => (e)
ternary return
generator function to endlessly yield multiples of a given number (partially done)
capitalise first letter
Node js express js
middleware to log all parameter names
path param vs query param use cases,
advantage of dynamic routing.
router chaining,
web api vs rest ap1,
how to check whether the given file exists or not?,

477. path module vs url module,
478. split controllers
479. CommonJS
480. env without dotenv
481. thread vs process
482. understand spawn()
483. HTTP OPTIONS,
484. CORS
485. using query params and path params in middleware
486. User-Agent
487. app.set
488. app.locals
489. express.urlencoded
490. CORS
491. nodejs workflow
492. middlewares
493. app.use
494. JSON.parse, stringify
495. fs
496. buffer class
497. streams, types of streams
498. cookies
499. view engines
500. child process
501. " Thread (it's not a line of code)
502. Process
503. cache miss
504. what's stored in cache
505. viewing cookies on a browser
506. nesting routers
507. where is cookies in HTTP request
508. parts of HTTP request
509. HTTP status code 403
510. common HTTP status code (including success)
511. GET vs POST
512. query params in url
513. components of a url
514. router params (variables)
515. await
516. promise.all vs promise.race - understand better
517. localstorage
518. set
519. get
520. authentication vs authorisation
521. event loop
522. streams - types
523. process.nextTick
524. process objects

525. cluster module
526. fork()
527. spawn()
528. Http status code
529. Promises v/s callback
530. generators
531. Node JS
532. process.nextTick
533. fork()
534. spawn()
535. What is Node.js?
536. Runtime environment
537. Advantages of Node
538. Features of Node
539. Node.js Architecture
540. Core modules in Node.js
541. NPM (Node Package Manager)
542. Npm init
543. Npm vs Npx
544. What is package.json? And package.lock.json
545. Dev dependencies vs dependencies
546. Alternatives to Express.js
547. Other Node.js frameworks
548. What is Express.js
549. Express.js features
550. Routing in Express.js
551. Router.all
552. HTTP vs HTTPS
553. HTTP methods
554. Request and response headers
555. Server communication
556. Static files
557. CORS (Cross-Origin Resource Sharing)
558. Middleware in Node.js
559. Example of all middlewares
560. Error handling middleware and its working
561. Streams and piping
562. Event-driven programming
563. Event-driven architecture
564. Event emitter
565. Process.nextTick
566. Console.error vs console.warning
567. DNS module
568. Process.env
569. Event loop
570. Libuv
571. Threadpool
572. Why single thread

573. Scaffolding
574. URL encoder
575. Body-parser
576. Session and cookies
577. MVC (Model-View-Controller)
578. PUT and PATCH HTTP methods
579. Scope chaining (let vs var)
580. Index of findings
581. Substring
582. Append
583. Create element
584. Addevent
585. Blocking code
586. JSON parse
587. JSON stringify
588. Conditional operator vs optional chaining
589. Clustering
590. Fork function ,spawn , exec and exec file
591. Router.all
592. Domain port flow
593. View engine
594. HTTP status codes
595. Headers
596. API development
597. HTTPS methods
598. Middleware types
599. Duplex stream
600. Garbage collection
601. Trace
602. Status code 400 range
603. BSON (Binary JSON)
604. Express use
605. Status code
606. Status code 400 range
607. Header
608. Asynchronous operations
609. Event handlers
610. REPL (Read-Eval-Print Loop)
611. hashmap,
612. floatmap,
613. json.stringify,
614. json.parse,
615. res.send
616. res.write
617. res.end
618. res.json,
619. query and params,
620. app.use,

621. app.set,
622. express-session,
623. maxAge vs Expires,
624. Cookie vs. sessionStorage vs. localStorage,
625. View Engine
626. child process
627. thread vs process
628. spawn() vs fork()
629. fs operations
630. CommonJS
631. environment variables (setting)
632. app.locals
633. static files
634. CSRF
635. query params, req.query
636. path params, req.params
637. CommonJS
638. env without dotenv
639. thread vs process
640. understand spawn()
641. HTTP OPTIONS, get a clearer idea of CORS
642. using query params and path params in middleware
643. User-Agent
644. app.set
645. app.locals
646. express.urlencoded
647. Logger Middleware: Logs details about incoming requests.
648. Body Parser Middleware: Parses JSON and URL-encoded request bodies.
649. Custom Middleware: Adds a custom header to the response.
650. Route-specific Middleware: Applies middleware only to specific routes.
651. Error-handling Middleware: Handles errors in the application.
652. Built-in Middleware: Serves static files from the 'public' directory.
653. Third-party Middleware: Uses the 'helmet' middleware for securing HTTP headers.
654. Cookie Parser Middleware: Parses cookie headers.
655. Session Middleware: Manages sessions.
656. Passport Middleware: Handles authentication.
657. Compression Middleware: Compresses responses.
658. CORS Middleware: Enables Cross-Origin Resource Sharing.
659. npm start
660. nodemon
661. package.json
662. package-lock.json
663. express.json vs express.urlencoded
664. body-parser
665. res.send vs res.write
666. Parts of HTTP request and response
667. HTTP status Codes

668. Cookie vs cache
669. reaFile vs readFileSync
670. query vs params
671. CORS
672. OPTIONS
673. types of middlewares
674. states in promise
675. methods of promise
676. app.use() vs app.set()
677. res.write() vs res.send()
678. express in detail
679. put vs post(idempotency)
680. Control flow
681. primary strategy for state management in node js
682. using code prove process.nexttick having high priority than setimmediate
683. process, threads and forking
684. nodejs threads
685. setting environment variables
686. browser doesn't create cookies
687. Parts of HTTP request and response
688. syntax (signature) of middleware
689. accessing query params
690. dynamic route
691. path param
692. app.set, settings
693. CSRF
694. fork: vs spawn
695. cluster module
696. how to set env without dotenv
697. CommonS
698. User-Agent
699. Eventloop
700. 3. Streams-types
701. 4. process.nextTick
702. 5. Buffer Class
703. 6. process objects
704. 7. Cluster Module
705. 8. Fork(
706. 9. Spawn)
707. 10. HTTPstatuscodes
708. 11. Middleware
709. 12. app.use
710. 13. JSON.parse, stringify
711. 14. fs
712. 16. Streams,types of streams
713. 17. Cookies
714. 18. Viewengines
715. 19. Childprocess

20. Environment variables
21. HTTP OPTIONS,
CORS
22. Queryparams,pathparams
23. Error Handling Middleware Args
24. User-Agent
25. Parts of HTTP request and response
26. CommonJSNode.js Express
28. Eventloop
29. process.nextTic
30. Cluster Module
31. process thread
32. Fork
33. Spawn
34. Error handling middleware args
35. Changing HTTP status code of response
36. Queryparams,req.query
37. User-Agent
38. Parts Of URL
39. Localhost
40. HTTPmethods
42. Router p a r a m s (variables)
options method
preflight request
thread pool
child process
dynamic routing
encryption
hashing
CSRF
write head
set header
use proper middlewares add proper auth middlewares
use env
encrypt a string in node JS
write date time into file
put vs post(idempotency)
options (need clarity)
Event driven architecture
Event emitter
Reactor pattern
Worker thread buffer need clarity)
Middlewares - P
Middleware positioning
Dev dependencies
app.all
Commonly used response status codes
boxing

764. set proper status codes use correct HTTP methods
765. JWI vs session
766. |WT signature
767. Js-labels
768. Event loop app.locals
769. Thread vs process
770. Command-line arguments
771. framework vs library
772. error first
773. concurrency
774. accessing query params
775. dynamic routing
776. CSRF
777. HTTP 400
778. browser cache (cookies and session aren't stored here)
779. content type
780. user agent *
781. structure of HTTP response
782. url fragments
783. invoking error handling middleware
784. accessing ENV
785. dynamic routing
786. route/path params
787. CSRF
788. HTTP 403
789. localStorage

Mongodb

790. Advantages of mongodb format of mongodb document
791. addToSet
792. default port
793. how to make mongodb structured
794. $cond if else
795. bulk write
796. batch sizing
797. transactions
798. virtual collection
799. hashed index
800. fs.files
801. fs.chunks
802. double the price values
803. fruit names ending with "E" (case insensitive)
804. name and price of priciest fruit
805. CAP theorem
806. isolation
807. sharding
808. replica sets
809. journaling
810. Namespace

811. Components of Id
812. List all the indexes in a collection - P
813. Aggregation
814. Update query - P
815. Department wise average salary of employees - P
816. Covered query
817. 3. V i e w
818. 4. TTL index
819. 5. Sfacet
820. 6. SaddFields
821. .7 Increase price of all fruits by 20% (Smul)
822. Red coloured vitamin C fruits
823. 9. Count of fruits that have a certain field
824. 10. Using projection
825. 11. Priciest fruit
826. 12. Embedded document
827. 13. WiredTiger
828. 14. Decrease price of all fruits by 0.5
829. 15. Diff between SQL and NoSQL
830. 16. Collection and document
831. 17. Find by ID
832. 18. Relational and document DB
833. 19. JSON and BSON
834. 20. Extended JSON
835. 21. eq, ne, nin, regex, nor, not operators
836. 23. MongoDB utilities
837. 24. MongoDB export,
838. import,
839. dump,
840. restore
841. 25. Serialising and deserializing
842. 26. findbyid and update and delete
843. 27. Capped collection
844. 31. Distinct
845. 32. Upsert
846. 33. addToSet
847. 34. Sunwind
848. 35. Sexists
849. Geospatial index
850. Sexists (careful with query structure)
851. Sor/Sand (practice, know where to use these) write concern
852. types of index - understand better
853. deleting by objectid
854. fetching conditionally, showing just some fields
855. Sregex for basic string matching
856. NoSQL injection
857. bulkWrite()
858. Retryable Writes/Retryable reads

859. count
860. group
861. $avg
862. $addtoset vs $push
863. $lookup query
864. Projection
865. Voting in replicasets
866. No of food in each category query
867. dispatch
868. Mongo
869. normalisation
870. denormalisation
871. covered query
872. document validation
873. $exists: documents that are missing a field
874. priciest vitamin C fruit
875. unique colors (color is array in each doc)
876. Design patterns,
877. updateMany syntax,
878. $inc decrement query,
879. No of books by each publisher query, $in query,
880. Remove vs drop,
881. Regular expression,
882. Backup and restore commands,
883. Scaling in mongodb
884. GridFs,
885. TTL index,
886. Partition tolerance,
887. journaling
888. Expressions and Operators
889. 1.Assignment Operators
890. 2.Arithematic Operators
891. 3.Logical Operators
892. 4.Conditional Operators
893. 5.Comparison Operators
894. 6.Relational Operators
895. 7.Bitwise Operators
896. 8.String Operators
897. 9.Comma Operators
898. 10.Unary Operators
899. Compound indexing
900. creating an empty collection
901. fetching conditionally, projections
902. sorting
903. renaming a field
904. conditionally updating field (adding fields and removing fields as well)
905. using basic regex (substring matching)
906. replication (why)

907. what is redundancy
908. dropping a collection
909. drawbacks of indexing
910. covered query, Sall,
911. Spop vs Spull,
912. query to find the student name who got second largest mark in class 10.
913. Create a capped Collection
914. Learn and Practice aggregation
915. $group
916. $match
917. $max
918. $min
919. BSON Types
920. Clustered Collection
921. $or ,$and ,$in
922. $expr
923. covered query
924. namespace
925. View
926. delete out of stock fruits
927. names ending with "y" ($regex)
928. unique colors (where color is an array field)
929. views,
930. atomicity,
931. shard key,
932. replication
933. types of indexes,
934. $facet,
935. $addToSet