120 MCQs Questions

1. `console.log(1 + "1" - 1);`

   A. "10"

   B. 10

   C. "11"

   D. 11

   Answer: B

2. `typeof NaN` returns:

   A. "number"

   B. "undefined"

   C. "string"

   D. "NaN"

   Answer: A

3. Removes the last array element:

   A. `shift()`

   B. `pop()`

   C. `splice()`

   D. `slice()`

   Answer: B

4. `let a = [1,2]; let b = [1,2]; console.log(a == b);`

   A. `true`

   B. `false`

   C. Error

   D. `undefined`

   Answer: B

5. `setTimeout(() => {}, 0)` executes:

   A. Immediately

   B. After the call stack is empty

   C. Never

   D. After 1 second

   Answer: B

6. A closure is:

   A. A function + its lexical environment

   B. An array method

   C. A data type

   D. A module import

   Answer: A

7. `console.log(!!0);`

   A. `true`

   B. `false`

   C. `0`

   D. `undefined`

   Answer: B

8. Correct `null` check:

   A. `typeof variable === "null"`

   B. `variable === null`

   C. `variable == null`

   D. Both B and C

   Answer: D

9. `"use strict";` prevents:

   A. Undeclared variables

   B. Type coercion

C. Async functions

D. Hoisting

Answer: A

10. Creates a new modified array:

    A. `forEach()`

    B. `map()`

    C. `filter()`

    D. `reduce()`

    Answer: B

11. `console.log(typeof []);`

    A. "array"

    B. "object"

    C. "undefined"

    D. "null"

    Answer: B

12. `let x = 10; function foo() { console.log(x); let x = 20; } foo();`

    A. 10

    B. 20

    C. ReferenceError

    D. undefined

    Answer: C

13. Immediately Invoked Function Expression (IIFE) is:

    A. A function that runs instantly

    B. A callback function

    C. An arrow function

    D. A generator

    Answer: A

14. `console.log("5" - 3);`

    A. 2

    B. "2"

    C. 53

    D. "53"

    Answer: A

15. `const obj = { a:1 }; Object.freeze(obj); obj.a=2; console.log(obj.a);`

    A. 1

    B. 2

    C. Error

    D. undefined

    Answer: A

16. `console.log(0.1 + 0.2 === 0.3);`

    A. true

    B. false

    C. TypeError

    D. undefined

    Answer: B

17. `const arr = [10, 12, 15, 21]; for (var i = 0; i < arr.length; i++) { setTimeout(() => console.log(i), 0); }`

    A. 0,1,2,3

    B. 4,4,4,4

    C. undefined,undefined,undefined,undefined

    D. 10,12,15,21

    Answer: B

18. `String.prototype` is used for:

    A. Adding methods to strings

    B. Converting numbers to strings

C. String comparison

D. Encoding strings

Answer: A

19. `console.log(3 > 2 > 1);`

A. true

B. false

C. undefined

D. Error

Answer: B

20. `const { a } = { a: 1, b: 2 }; console.log(a);`

A. 1

B. { a:1 }

C. undefined

D. Error

Answer: A

21. `Promise.reject().catch(() => 1).then(console.log);`

A. 1

B. undefined

C. Error

D. null

Answer: A

22. `const sym = Symbol('key'); const obj = { [sym]: 'value' }; console.log(obj[sym]);`

A. 'value'

B. undefined

C. Error

D. Symbol('key')

Answer: A

23. `console.log(new Date(2023, 1, 30).getMonth());`

A. 1

B. 2

C. 30

D. 0

Answer: A

24. `console.log(parseInt("7em"));`

A. 7

B. "7em"

C. NaN

D. 0

Answer: A

25. `const fn = () => arguments.length; console.log(fn(1,2));`

A. 2

B. 0

C. ReferenceError

D. undefined

Answer: C

26. `console.log("hello".padStart(10, "123"));`

A. "123123hello"

B. "12312hello"

C. "hello12312"

D. "hello"

Answer: B

27. `console.log(typeof function* () {});`

A. "function"

B. "generator"

C. "object"

D. "undefined"

Answer: A

28. `const arr = [1,2]; arr.length = 0; console.log(arr[0]);`

A. 1

B. undefined

C. 0

D. Error

Answer: B

29. `console.log(+"Infinity");`

A. Infinity

B. "Infinity"

C. NaN

D. TypeError

Answer: A

30. `const obj = { a:1 }; console.log(JSON.stringify(obj, ['a']));`

A. `{"a":1}`

B. `{}`

C. `"a"`

D. Error

Answer: A

31. **What is the output?**

javascript

```
console.log(1 + "1" - 1);
```

A. "10"

B. 10

C. "11"

D. 11

Answer: B

32. **What does `typeof NaN` return?**

A. "number"

B. "undefined"

C. "string"

D. "NaN"

Answer: A

33. **Which method removes the last element of an array?**

A. `shift()`

B. `pop()`

C. `splice()`

D. `slice()`

Answer: B

34. **What is the output?**

javascript

```
let a = [1, 2];
let b = [1, 2];
console.log(a == b);
```

A. `true`

B. `false`

C. Error

D. `undefined`

Answer: B

35. **What does `setTimeout(() => {}, 0)` do?**

A. Executes immediately

B. Executes after the call stack is empty

C. Never executes

D. Executes after 1 second

Answer: B

36. **What is `closure` in JavaScript?**

A. A function bundled with its lexical environment

B. A built-in array method

C. A data type

D. A way to import modules

Answer: A

37. **What is the output?**

javascript

```
console.log(!!0);
```

A. `true`

B. `false`

C. `0`

D. `undefined`

Answer: B

38. **How do you check for `null`?**

A. `typeof variable === "null"`

B. `variable === null`

C. `variable == null`

D. Both B and C

Answer: D

39. **What does `"use strict";` do?**

A. Enforces stricter type checking

B. Enables modern JavaScript features

C. Prevents undeclared variables

D. Optimizes performance

Answer: C

40. **Which method creates a new array with modified elements?**

A. `forEach()`

B. `map()`

C. `filter()`

D. `reduce()`

Answer: B

41. **What is Node.js primarily used for?**

A. Frontend development

B. Server-side scripting

C. Mobile apps

D. Database management

Answer: B

42. **Which module handles file operations?**

A. `http`

B. `fs`

C. `path`

D. `url`

Answer: B

43. **What does `require('http')` do?**

A. Imports an HTTP server module

B. Fetches data from a URL

C. Creates a frontend HTTP request

D. Encrypts HTTP traffic

Answer: A

44. **What is `npm`?**

    A. Node.js Package Manager

    B. New Project Manager

    C. Network Protocol Module

    D. Native Process Monitor

    Answer: A

45. **How do you start a Node.js server?**

    A. `node start server.js`

    B. `npm run server.js`

    C. `node server.js`

    D. `npm start server.js`

    Answer: C

46. Node.js is primarily used for:

    A. Frontend development

    B. Server-side scripting

    C. Mobile apps

    D. Database management

    Answer: B

47. File operations module:

    A. `http`

    B. `fs`

    C. `path`

    D. `url`

    Answer: B

48. `require('http')` imports:

A. HTTP server module

B. URL fetcher

C. Frontend request tool

D. Encryption module

Answer: A

49. `npm` stands for:

A. Node Package Manager

B. New Project Manager

C. Network Protocol Module

D. Native Process Monitor

Answer: A

50. Starting a Node.js server:

A. `node start server.js`

B. `npm run server.js`

C. `node server.js`

D. `npm start server.js`

Answer: C

51. `__dirname` returns:

A. Current filename

B. Current directory path

C. Node.js version

D. Environment variables

Answer: B

52. The `events` module uses:

A. Observer pattern

B. Singleton pattern

C. Factory pattern

D. MVC pattern

Answer: A

53. `process.env` accesses:

A. Command-line arguments

B. Environment variables

C. File system

D. Network interfaces

Answer: B

54. Middleware in Express.js:

A. Handles HTTP requests

B. Manages databases

C. Renders frontend views

D. Validates user input

Answer: A

55. `Buffer` class handles:

A. Binary data

B. JSON data

C. Environment variables

D. File paths

Answer: A

56. `require()` caches modules:

A. Per process

B. Per file

C. Per function

D. Never

Answer: A

57. `setImmediate()` vs `setTimeout(() => {}, 0)`:

A. `setImmediate()` runs first

B. `setTimeout()` runs first

C. Same execution order

D. Random order

Answer: D (Order is environment-dependent)

58. `cluster` module enables:

A. Multi-threading

B. Child processes

C. HTTP/2 support

D. Database sharding

Answer: A

59. `util.promisify()` converts:

A. Callbacks → Promises

B. Promises → Callbacks

C. Sync → Async

D. Streams → Buffers

Answer: A

60. `res.sendFile()` in Express.js:

A. Sends a file as response

B. Uploads a file

C. Reads a file

D. Deletes a file

Answer: A

61. **MongoDB stores data as:**

A. Tables

B. JSON documents

C. CSV files

D. XML

Answer: B

62. **Which command inserts a document?**

A. `db.collection.insertOne()`

B. `db.collection.add()`

C. `db.collection.create()`

D. `db.collection.save()`

Answer: A

63. **What is `_id` in MongoDB?**

A. A required unique identifier

B. An optional field

C. A foreign key

D. A temporary placeholder

Answer: A

64. **Which operator updates fields?**

A. `$push`

B. `$set`

C. `$update`

D. `$change`

Answer: B

65. **Which method finds documents?**

A. `db.collection.search()`

B. `db.collection.get()`

C. `db.collection.find()`

D. `db.collection.fetch()`

Answer: C

66. MongoDB stores data as:

A. Tables

B. JSON documents

C. CSV files

D. XML

Answer: B

67. Inserts a document:

A. `db.collection.insertOne()`

B. `db.collection.add()`

C. `db.collection.create()`

D. `db.collection.save()`

Answer: A

68. `_id` field is:

A. Required and unique

B. Optional

C. Foreign key

D. Temporary

Answer: A

69. Updates fields:

A. `$push`

B. `$set`

C. `$update`

D. `$change`

Answer: B

70. Finds documents:

A. `db.collection.search()`

B. `db.collection.get()`

C. `db.collection.find()`

D. `db.collection.fetch()`

Answer: C

71. `$inc` operator:

A. Increments a value

B. Decrements a value

C. Inserts a field

D. Deletes a field

Answer: A

72. Indexes improve:

A. Query performance

B. Data storage size

C. Network latency

D. Security

Answer: A

73. Sharding:

A. Splits data across servers

B. Encrypts data

C. Compresses data

D. Backs up data

Answer: A

74. Aggregation `$group` stage:

A. Groups documents by expression

B. Filters documents

C. Sorts documents

D. Limits documents

Answer: A

75. `$lookup` performs:

A. Left outer join

B. Inner join

C. Cross join

D. Union

Answer: A

76. TTL index:

A. Auto-deletes documents after time

B. Times query execution

C. Tracks login sessions

D. Limits collection size

Answer: A

77. `{ name: { $regex: /^J/ } }` finds:

A. Names starting with "J"

B. Names ending with "J"

C. Names containing "J"

D. Case-insensitive "J"

Answer: A

78. Atomic operations:

A. Guarantee no partial updates

B. Run in parallel

C. Are slow

D. Require sharding

Answer: A

79. MongoDB is a:

A. Relational DB

B. NoSQL DB

C. Graph DB

D. Columnar DB

Answer: B

30. `db.collection.deleteMany({})` deletes:

    A. All documents

    B. First document

    C. Indexes

    D. Collection

    Answer: A

31. Creates a state variable:

    A. `let state = useState(0);`

    B. `const [count, setCount] = useState(0);`

    C. `state = createState(0);`

    D. `this.state = { count: 0 };`

    Answer: B

32. JSX is:

    A. A JavaScript testing framework

    B. Syntax extension for JavaScript

    C. State management library

    D. Build tool

    Answer: B

33. Passes data to child components:

    A. `state`

    B. `props`

    C. `context`

    D. `refs`

    Answer: B

34. `useEffect` purpose:

A. Manage state

B. Perform side effects

C. Create reusable logic

D. Handle routing

Answer: B

35. Replaces `componentDidMount`:

A. `useState`

B. `useEffect` with `[]`

C. `useContext`

D. `useReducer`

Answer: B

36. `key` prop is used for:

A. Identifying elements uniquely

B. Styling components

C. Passing data

D. Handling events

Answer: A

37. Controlled component:

A. Manages state via React

B. Manages its own state

C. Cannot receive props

D. Is a class component

Answer: A

38. `useMemo` memoizes:

A. Computed values

B. Components

C. API calls

D. Event handlers

Answer: A

39. React Router uses:

A. `BrowserRouter`

B. `RouteController`

C. `Navigation`

D. `LinkRedirect`

Answer: A

40. `useRef` primarily accesses:

A. DOM elements

B. State variables

C. Props

D. Context

Answer: A

41. Higher-Order Component (HOC):

A. Function that takes a component and returns a new component

B. Base class for all components

C. Hook for state management

D. Router component

Answer: A

42. `useContext` requires:

A. `React.createContext()`

B. `React.Provider`

C. Both A and B

D. `useReducer`

Answer: C

43. Server-Side Rendering (SSR) improves:

A. SEO and initial load time

B. Runtime performance

C. Bundle size

D. State management

Answer: A

94. `React.lazy()` enables:

A. Code-splitting

B. State updates

C. Error boundaries

D. Prop validation

Answer: A

95. `children` prop contains:

A. Content between component tags

B. Child components only

C. State values

D. Event handlers

Answer: A

96. **How do you create a state variable?**

A. `let state = useState(0);`

B. `const [count, setCount] = useState(0);`

C. `state = createState(0);`

D. `this.state = { count: 0 };`

Answer: B

97. **What is JSX?**

A. A JavaScript testing framework

B. A syntax extension for JavaScript

C. A state management library

D. A build tool

Answer: B

98. **How do you pass data to a child component?**

A. Using `state`

B. Using `props`

C. Using `context`

D. Using `refs`

Answer: B

99. **What is the purpose of `useEffect`?**

A. To manage state

B. To perform side effects

C. To create reusable logic

D. To handle routing

Answer: B

100. **Which hook replaces `componentDidMount`?**

A. `useState`

B. `useEffect` with empty dependency array

C. `useContext`

D. `useReducer`

Answer: B

101. **Which tag defines a hyperlink?**

A. `<link>`

B. `<a>`

C. `<href>`

D. `<url>`

Answer: B

102. **What is the correct structure?**

A. `<html><head></head><body></body></html>`

B. `<head><html></html><body></body></head>`

C. `<body><head></head><html></html></body>`

D. `<html><body></body><head></head></html>`

Answer: A

03. **Which attribute makes an input field required?**

A. `mandatory="true"`

B. `validate="required"`

C. `required`

D. `mustfill`

Answer: C

04. Defines a hyperlink:

A. `<link>`

B. `<a>`

C. `<href>`

D. `<url>`

Answer: B

05. Correct HTML structure:

A. `<html><head></head><body></body></html>`

B. `<head><html></html><body></body></head>`

C. `<body><head></head><html></html></body>`

D. `<html><body></body><head></head></html>`

Answer: A

06. Required input field attribute:

A. `mandatory="true"`

B. `validate="required"`

C. `required`

D. `mustfill`

Answer: C

07. `alt` attribute in `<img>` is for:

A. Alternative text

B. Image alignment

C. Source URL

D. Image dimensions

Answer: A

08. Semantic element for navigation:

A. `<div>`

B. `<span>`

C. `<nav>`

D. `<header>`

Answer: C

09. Self-closing tag:

A. `<div></div>`

B. `<br>`

C. `<p></p>`

D. `<a></a>`

Answer: B

10. `defer` attribute in `<script>`:

A. Delays script execution

B. Runs script after HTML parsing

C. Runs script immediately

D. Disables script

Answer: B

11. `datalist` element creates:

A. A dropdown input menu

B. A table

C. A list of links

D. A form

Answer: A

12. `contenteditable` attribute:

A. Makes element editable

B. Hides content

C. Validates input

D. Encodes content

Answer: A

13. Selects `id="header"`:

A. `#header`

B. `.header`

C. `*header`

D. `header`

Answer: A

14. Changes text color:

A. `text-style`

B. `font-color`

C. `color`

D. `text-color`

Answer: C

15. `position: absolute` positions relative to:

A. Nearest positioned ancestor

B. Document body

C. Parent element

D. Viewport

Answer: A

16. Flexbox property for horizontal alignment:

    A. `align-items`

    B. `justify-content`

    C. `flex-direction`

    D. `flex-wrap`

    Answer: B

17. `z-index` controls:

    A. Stacking order

    B. Element width

    C. Animation duration

    D. Color depth

    Answer: A

18. `@media` queries enable:

    A. Responsive design

    B. Animation keyframes

    C. Variable definitions

    D. Font imports

    Answer: A

19. **How do you select an element with `id="header"`?**

    A. `#header`

    B. `.header`

    C. `*header`

    D. `header`

    Answer: A

20. **Which property changes text color?**

A. `text-style`

B. `font-color`

C. `color`

D. `text-color`

Answer: C