

```
1  /*
2
3  =====
4
5      Arrays Interview Questions 🔥
6
7  =====
8
9  */
10
11 console.clear();
12
13 /*
14
15 1. Remove all falsy values (false, 0, "", null, undefined, NaN) from an array
16
17 */
18
19 const booleanArray = [0, 1, false, 2, "", 3, null, undefined, NaN, 4, true];
20
21 function removeFalsyValues(arr) {
22
23     // filter(Boolean) removes all falsy values by only keeping truthy ones
24
25     return arr.filter(Boolean);
26 }
27
28 const filteredArray = removeFalsyValues(booleanArray);
29
30 console.log("The filtered array is:", filteredArray);
31
32 /*
33
34 2. Reverse an array without using .reverse()
35
36 */
37
```

O

```
38 const originalArray = [1, 2, 3, 4, 5];
39
40 const reversedArray = [];
41
42 // Loop from the last index and push into a new array
43
44 for (let i = originalArray.length - 1; i >= 0; i--) {
45
46     reversedArray.push(originalArray[i]);
47 }
48
49 console.log("The reversed array is:", reversedArray);
50
51 /*
52 3. Clone an array using slice() and Array.from()
53
54 */
55
56 const original = [1, 2, 3];
57
58 // slice() creates a shallow copy
59
60 const clone1 = original.slice();
61
62 // Array.from() creates a shallow copy
63
64 const clone2 = Array.from(original);
65
66 console.log("Cloned Arrays:", clone1, clone2);
67
68 /*
69
70
71 4. Merge multiple arrays using concat()
72
73 */
74
75 const arr1 = [1, 2];
```

```
76
77 const arr2 = [3, 4];
78
79 // concat merges arrays and returns a new one
80
81 const merged = arr1.concat(arr2, [5, 6]);
82
83 console.log("Merged Array:", merged);
84
85 /*
86
87 5. Convert an array to a string using join()
88
89 */
90
91 const words = ["Hello", "World"];
92
93 // join(" ") combines elements with space between them
94
95 console.log("Joined String:", words.join(" "));
96
97 /*
98
99 6. Count occurrences using reduce()
100
101 */
102
103 const votes = ["yes", "no", "yes", "yes", "no"];
104
105 // Tally each element's count in an object
106
107 const voteCount = votes.reduce((acc, vote) => {
108
109     acc[vote] = (acc[vote] || 0) + 1;
110
111     return acc;
112
113 }, {});
```

```
114
115 console.log("Vote Count:", voteCount);
116
117 /*
118
119 7. Remove duplicates manually using indexOf()
120
121 */
122
123 const duplicates = [1, 2, 3, 2, 4, 1];
124
125 const unique = [];
126
127 // Only push if element is not already present
128
129 for (let i = 0; i < duplicates.length; i++) {
130
131     if (unique.indexOf(duplicates[i]) === -1) {
132
133         unique.push(duplicates[i]);
134     }
135 }
136
137 console.log("Unique Array:", unique);
138
139 /*
140
141 8. Convert string to array using split()
142
143 */
144
145 const sentence = "Split this sentence";
146
147 // split(" ") breaks sentence by spaces
148
149 console.log("Array from String:", sentence.split(" "));
150
151 /*
```

```
152
153 9. Flatten a 1-level nested array using concat()
154
155 */
156
157 const nested = [1, [2, 3], 4];
158
159 // concat with spread to flatten one level
160
161 const flattened = [].concat(nested[0], nested[1], nested[2]);
162
163 console.log("Flattened Array:", flattened);
164
165 /*
166
167 10. Replace elements using map()
168
169 */
170
171 const nums = [1, 2, 3, 4];
172
173 // map returns a new array with modified values
174
175 const doubled = nums.map(function (num) {
176     return num * 2;
177 });
178
179 console.log("Doubled:", doubled);
180
181 /*
182
183
184 11. Check if all elements are even using every()
185
186 */
187
188 const evenCheck = [2, 4, 6].every(function (num) {
189
```

```
190     return num % 2 === 0;
191
192 });
193
194 console.log("All Even:", evenCheck);
195
196 /*
197
198 12. Check if any element is negative using some()
199
200 */
201
202 const mixed = [3, -1, 5];
203
204 const hasNegative = mixed.some(function (num) {
205
206     return num < 0;
207
208 });
209
210 console.log("Contains Negative:", hasNegative);
211
212 /*
213
214 13. Find the max value using reduce()
215
216 */
217
218 const values = [5, 2, 8, 1];
219
220 // Compare each value and keep the highest
221
222 const max = values.reduce(function (a, b) {
223
224     return a > b ? a : b;
225
226 });
227
```

```
228 console.log("Max Value:", max);
229
230 /*
231
232 14. Replace or delete elements using splice()
233
234 */
235
236 const tools = ["hammer", "wrench", "screwdriver"];
237
238 // Replace 'wrench' at index 1 with 'pliers'
239
240 tools.splice(1, 1, "pliers");
241
242 console.log("After Splice:", tools);
243
244 /*
245
246 15. Get a portion of an array using slice()
247
248 */
249
250 const colors = ["red", "green", "blue", "yellow"];
251
252 // Extract elements from index 1 to 3 (excluding 3)
253
254 const sliced = colors.slice(1, 3);
255
256 console.log("Sliced Part:", sliced);
257
258 /*
259
260 16. Check if an element exists using includes()
261
262 */
263
264 console.log("Has blue?", colors.includes("blue")); // true or false
265
```

```
266  /*
267
268  17. Get index using indexOf() and lastIndexOf()
269
270  */
271
272  // First occurrence
273
274  console.log("First index of 'red':", colors.indexOf("red"));
275
276  // Last occurrence
277
278  console.log("Last index of 'blue':", colors.lastIndexOf("blue"));
279
280  /*
281
282  18. Iterate using forEach()
283
284  */
285
286  colors.forEach(function (color, index) {
287
288      // Access both index and value
289
290      console.log(index + ":", color);
291  });
292
293  /*
294
295  19. Sort array of objects by age using sort()
296
297  */
298
299  const users = [
300
301      { name: "Heet", age: 22 },
302
303      { name: "Aman", age: 20 },
```



```
304 ];
305
306 // Sort by age (ascending)
307
308 users.sort(function (a, b) {
309     return a.age - b.age;
310 });
311
312
313 console.log("Sorted Users by Age (Ascending):", users);
314
315 /*
316
317 20. Sum even numbers using filter() and reduce()
318
319 */
320
321 const data = [1, 2, 3, 4, 5, 6];
322
323 // Filter evens, then sum them
324
325 const evenSum = data
326     .filter(function (n) {
327         return n % 2 === 0;
328     })
329     .reduce(function (a, b) {
330         return a + b;
331     }, 0);
332
333 console.log("Sum of Evens:", evenSum);
334
335
336 /*
337
338
339
340
341
```

```
342 21. Convert arguments to array using Array.from()
343
344 */
345
346 function argsToArray() {
347     // Array.from converts arguments object to real array
348
349     const argsArray = Array.from(arguments);
350
351     console.log("Arguments as Array:", argsArray);
352 }
353
354
355 argsToArray(10, 20, 30);
```