JavaScript Array Interview Questions — Cheat Sheet with Solutions

Easy, Medium, and Hard array questions with ready-to-run JavaScript solutions. Use the practice space to paste and run these in your console or editor.

Q: Find the largest number in an array.

Ans: Solution: Use Math.max with spread operator.

```
const arr = [10, 5, 30, 40];
console.log(Math.max(...arr)); // 40
```

Q: Find the smallest number in an array.

Ans: Solution: Use Math.min with spread operator.

```
const arr = [10, 5, 30, 40];
console.log(Math.min(...arr)); // 5
```

Q: Reverse an array without using .reverse().

Ans: Solution: Use reduce to build reversed array.

```
const arr = [1, 2, 3, 4];
const reversed = arr.reduce((acc, val) => [val, ...acc], []);
console.log(reversed); // [4,3,2,1]
```

Q: Check if all elements are even numbers.

Ans: Solution: Use .every().

```
const arr = [2, 4, 6];
console.log(arr.every(num => num % 2 === 0)); // true
```

Q: Remove duplicates from an array.

Ans: Solution: Use Set or reduce.

```
const arr = [1, 2, 2, 3, 4, 4];
const unique = [...new Set(arr)];
console.log(unique); // [1,2,3,4]
```

Q: Find the second largest number in an array.

Ans: Solution: Create unique set, sort desc, take index 1.

```
const arr = [10, 20, 40, 30];
const unique = [...new Set(arr)].sort((a,b)=>b-a);
console.log(unique[1]); // 30
```

Q: Count frequency of elements in an array.

Ans: Solution: Use reduce to build frequency map.

```
const arr = ['a','b','a','c','b','a'];
const freq = arr.reduce((acc, val) => {
  acc[val] = (acc[val] || 0) + 1;
  return acc;
}, {});
console.log(freq); // { a:3, b:2, c:1 }
```

Q: Flatten a nested array.

Ans: Solution: Use flat(Infinity) or recursive function.

```
const arr = [1, [2, [3, 4]], 5];
console.log(arr.flat(Infinity)); // [1,2,3,4,5]
```

Q: Find pair of numbers whose sum is target (return indices).

Ans: Solution: Use a hash map to track values to indices (one-pass).

```
const arr = [2,7,11,15];
const target = 9;
const map = {};
for (let i = 0; i < arr.length; i++) {
  const diff = target - arr[i];
  if (map[diff] !== undefined) console.log([map[diff], i]);
  map[arr[i]] = i;
}
// Output: [0,1]
```

Q: Rotate array by k steps.

Ans: Solution: Use slice and spread to reorder.

```
const rotate = (arr, k) => {
  k = k % arr.length;
  return [...arr.slice(-k), ...arr.slice(0, -k)];
};
console.log(rotate([1,2,3,4,5],2)); // [4,5,1,2,3]
```

Q: Find intersection of two arrays.

Ans: Solution: Use Set for efficiency and filter.

```
const a = [1,2,3,4];
const b = [3,4,5,6];
const setB = new Set(b);
const intersection = [...new Set(a.filter(x => setB.has(x)))];
console.log(intersection); // [3,4]
```

Q: Find missing number in array 1..n.

Ans: Solution: Use sum formula n*(n+1)/2 minus actual sum.

```
const arr = [1,2,4,5];
const n = 5;
const total = (n*(n+1))/2;
const sum = arr.reduce((a,b)=>a+b,0);
console.log(total - sum); // 3
```

Q: Maximum subarray sum (Kadane's Algorithm).

Ans: Solution: Iterate maintaining current and max sums.

```
const arr = [-2,1,-3,4,-1,2,1,-5,4];
let maxSum = arr[0], currSum = 0;
for (let num of arr) {
  currSum = Math.max(num, currSum + num);
  maxSum = Math.max(maxSum, currSum);
}
console.log(maxSum); // 6
```

Q: 3Sum — find unique triplets that sum to 0.

Ans: Solution: Sort and use two-pointer technique to avoid duplicates.

```
const arr = [-1,0,1,2,-1,-4];
arr.sort((a,b)=>a-b);
const res = [];
for (let i=0;i<arr.length-2;i++){
  if (i>0 && arr[i]==arr[i-1]) continue;
  let l=i+1, r=arr.length-1;
  while(l<r){
    const s = arr[i]+arr[l]+arr[r];
    if (s===0) { res.push([arr[i],arr[l],arr[r]]); l++; r--; while(arr[l]===arr[l-1]) l++
    else if (s<0) l++; else r--;
  }
}
console.log(res); // [[-1,-1,2],[-1,0,1]]
```

Q: Implement your own .map().

Ans: Solution: Add myMap to Array.prototype that iterates and applies callback.

```
Array.prototype.myMap = function(cb){
  const res = [];
  for (let i=0;i<this.length;i++) res.push(cb(this[i], i, this));
  return res;
};
console.log([1,2,3].myMap(x=>x*2)); // [2,4,6]
```

■ Quick Tricks

Remove duplicates → [...new Set(arr)]
Sort descending → arr.sort((a,b)=>b-a)
Flatten nested array → arr.flat(Infinity)
Reverse string → str.split('').reverse().join('')
Find frequency → arr.reduce((a,v)=>(a[v]=(a[v]||0)+1,a),{})

■ Practice Space