1. Find Common Elements Between Two Arrays

Question: Given two arrays, find their common elements.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let commonElements = arr1.filter(element => arr2.includes(element));
console.log(commonElements); // Output: [2, 1, 3]
```

Explanation:

1. **filter()**:

   Iterates through each element in arr1.

   Checks if the element exists in arr2 using includes().

2. **includes()**:

   Returns true if the element exists in arr2, otherwise false.

3. **Result**:

   The elements that return true are added to the commonElements array.

2. Find Common Elements Without Duplicates

Question: Find common elements, but ensure the result has no duplicates.

Code:

```
let arr1 = [2, 1, 4, 6, 3, 2];
let arr2 = [1, 7, 8, 3, 2, 2];

let commonElements = [...new Set(arr1.filter(element => arr2.includes(element)))];
console.log(commonElements); // Output: [2, 1, 3]
```

Explanation:

1. **Set**:Removes duplicates from the array.
2. **Process**:

   Use filter() to find common elements.

Pass the result to new Set() to remove duplicates.

Convert back to an array using the spread operator [...].

3. Find Elements Unique to Each Array

Question: Find elements that are unique to each array (not common).

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let uniqueToArr1 = arr1.filter(element => !arr2.includes(element));
let uniqueToArr2 = arr2.filter(element => !arr1.includes(element));
let uniqueElements = [...uniqueToArr1, ...uniqueToArr2];

console.log(uniqueElements); // Output: [4, 6, 7, 8]
```

Explanation:

1. **Unique to arr1**:

   Use filter() to find elements in arr1 that do not exist in arr2.

2. **Unique to arr2**:

   Similarly, find elements in arr2 that do not exist in arr1.

3. **Combine Results**:

   Merge both arrays using the spread operator [...].

4. Find Union of Two Arrays

Question: Combine two arrays and remove duplicates.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let union = [...new Set([...arr1, ...arr2])];
console.log(union); // Output: [2, 1, 4, 6, 3, 7, 8]
```

Explanation:

1. **Combine Arrays**:

Use the spread operator to merge arr1 and arr2.

2. **Remove Duplicates**:

   Pass the merged array into new Set() to eliminate duplicates.

5. Find Intersection Using Sets

Question: Find common elements using Set for better performance.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let set1 = new Set(arr1);
let intersection = arr2.filter(element => set1.has(element));

console.log(intersection); // Output: [1, 3, 2]
```

Explanation:

1. **Create Set**:

   Convert arr1 into a Set to enable quick lookups.

2. **Find Intersection**:

   Use filter() on arr2 and check if set1 contains the element.

6. Find Difference Between Two Arrays

Question: Find elements in arr1 that are not in arr2.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let difference = arr1.filter(element => !arr2.includes(element));
console.log(difference); // Output: [4, 6]
```

Explanation:

1. **Filter Elements**:

   Use filter() to find elements in arr1 that do not exist in arr2.

7. Find Symmetric Difference

Question: Find elements that are in either array but not in both.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let diff1 = arr1.filter(element => !arr2.includes(element));
let diff2 = arr2.filter(element => !arr1.includes(element));
let symmetricDifference = [...diff1, ...diff2];

console.log(symmetricDifference); // Output: [4, 6, 7, 8]
```

Explanation:

1. **Unique to arr1**:

   Find elements in arr1 not in arr2.

2. **Unique to arr2**:

   Find elements in arr2 not in arr1.

3. **Combine Results**:

   Merge the two results to get the symmetric difference.

8. Find Common Elements in Three Arrays

Question: Find common elements in three arrays.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];
let arr3 = [3, 2, 9, 1];

let commonElements = arr1.filter(element => arr2.includes(element) &&
arr3.includes(element));
console.log(commonElements); // Output: [1, 3, 2]
```

Explanation:

1. **Filter for All Arrays**:

   Use filter() to check if an element exists in both arr2 and arr3.

9. Check if Two Arrays Are Disjoint

Question: Check if two arrays have no common elements.

Code:

```
let arr1 = [2, 4, 6];
let arr2 = [1, 7, 8];

let isDisjoint = arr1.every(element => !arr2.includes(element));
console.log(isDisjoint); // Output: true
```

Explanation:

1. **every()**:

   Checks if every element in arr1 does not exist in arr2.

10. Count Common Elements

Question: Count the number of common elements between two arrays.

Code:

```
let arr1 = [2, 1, 4, 6, 3];
let arr2 = [1, 7, 8, 3, 2];

let commonCount = arr1.filter(element => arr2.includes(element)).length;
console.log(commonCount); // Output: 3
```

Explanation:

1. **Find Common Elements**:

   Use filter() to find common elements.

2. **Count**:

   Use .length to count the number of elements in the resulting array.