```
/*  Problem: Write a function to reverse an array
    Input - [1, 2, 3, 4, 5]
    Output - [5, 4, 3, 2, 1]
*/

const reverseArray = arr => {
  let start = 0;
  let end = arr.length - 1;
  while (start < end) {
    // Swap elements
    [arr[start], arr[end]] = [arr[end], arr[start]];
    start++;
    end--;
  }
  return arr;
};
```

```javascript
/*  Problem: Move Zeros to the End
    Input - [0, 1, 0, 3, 12]
    Output - [1, 3, 12, 0, 0]
*/

const moveZerosToEnd = arr => {
  let pointer = 0;
  for (let i = 0; i < arr.length; i++) {
    if (arr[i] !== 0) {
      [arr[i], arr[pointer]] = [arr[pointer], arr[i]];
      pointer++;
    }
  }
  return arr;
};
```

```
/*  Problem: Find first Non-Repeating Element
    Input - [4, 5, 1, 2, 0, 4]
    Output - 5
*/

const firstNonRepeating = arr => {
  const frequency = {};
  for (let num of arr) {
    frequency[num] = (frequency[num] || 0) + 1;
  }
  for (let num of arr) {
    if (frequency[num] === 1) {
      return num;
    }
  }
  // Return -1 if no non-repeating element is found
  return -1;
};
```

```
/*  Problem: Remove Duplicates from Array
    Input - [1, 2, 2, 3, 4, 4, 5]
    Output - [1, 2, 3, 4, 5]
*/

const removeDuplicates = arr => {
  return [...new Set(arr)];
};
```

```
/*  Problem: Find Missing Number in Array
    Input - [1, 2, 4, 5], n=5
    Output - 3
*/

const findMissingNumber = (arr, n) => {
  const expectedSum = (n * (n + 1)) / 2;
  const actualSum = arr.reduce((sum, num) => {
    return sum + num;
  }, 0);
  return expectedSum - actualSum;
};
```

```
/*  Problem: Find Array Pairs with Target Sum
    Input - [2, 7, 11, 15], target=9
    Output - [[2, 7]]
*/

const findPairs = (arr, target) => {
  const seen = new Set();
  const pairs = [];
  for (let num of arr) {
    const complement = target - num;
    if (seen.has(complement)) {
      pairs.push([complement, num]);
    }
    seen.add(num);
  }
  return pairs;
};
```

```javascript
/*  Problem: Rotate an Array by k Positions
    Input - [1, 2, 3, 4, 5, 6, 7], k=3
    Output - [5, 6, 7, 1, 2, 3, 4]
*/

const rotateArray = (arr, k) => {
  // Handle k greater than array length
  k = k % arr.length;
  return [...arr.slice(-k), ...arr.slice(0, -k)];
};
```

```javascript
/*  Problem: Find Intersection of Two Arrays
    Input - arr1 = [1, 2, 2, 1], arr2 = [2, 2]
    Output - [2, 2]
*/

const intersection = (arr1, arr2) => {
  const map = new Map();
  const result = [];
  for (let num of arr1) {
    map.set(num, (map.get(num) || 0) + 1);
  }
  for (let num of arr2) {
    if (map.get(num) > 0) {
      result.push(num);
      map.set(num, map.get(num) - 1);
    }
  }
  return result;
};
```

```javascript
/*  Problem: Find Largest Sum of Contiguous Subarray (Kadane's Algorithm)
    Input - [-2, 1, -3, 4, -1, 2, 1, -5, 4]
    Output - 6 (The subarray [4, -1, 2, 1] has the largest sum)
*/

const maxSubArraySum = arr => {
  let maxSum = arr[0];
  let currentSum = arr[0];
  for (let i = 1; i < arr.length; i++) {
    currentSum = Math.max(arr[i], currentSum + arr[i]);
    maxSum = Math.max(maxSum, currentSum);
  }
  return maxSum;
};
```

```javascript
/*  Problem: Character Frequency Count in a String
    Input - aaaabbcccccc
    Output - a4b2c5
*/

const characterFrequency = (str) => {
  let frequency = {};
  let result = '';

  // Count occurrences of each character
  for (let char of str) {
    frequency[char] = (frequency[char] || 0) + 1;
  }

  // Build the compressed string
  for (let char in frequency) {
    result += char + frequency[char];
  }

  return result;
};
```