10 Helpful JavaScript Code Snippets

Write a function that removes all duplicate elements from an array in JavaScript. **2**

Write a function that calculates the nth Fibonacci number in JavaScript. **2**

Write a function that returns the maximum value in an array without using the Math.max() method in JavaScript. **3**

Write a function that checks whether a given string is a palindrome (i.e., reads the same backwards and forwards) in JavaScript. **3**

Write a function that returns the first non-repeated character in a given string in JavaScript. **4**

Write a function that generates a random string of a given length in JavaScript. **5**

Write a function that sorts an array of objects by a specified key in JavaScript. **5**

Write a function that flattens a nested array of any depth in JavaScript. **6**

Write a function that finds the common elements between two arrays in JavaScript. **6**

Write a function that removes all duplicate elements from an array in JavaScript.

```
function removeDuplicates(arr) {
  return [...new Set(arr)];
}
```

Explanation: The function first creates a Set from the input array using the new Set() constructor. This removes all duplicate elements because Set objects only store unique values. Then, it converts the Set back to an array using the spread operator ([...set]).

Write a function that calculates the nth Fibonacci number in JavaScript.

```
function fibonacci(n) {
  let a = 0;
  let b = 1;
  for (let i = 0; i < n; i++) {
    [a, b] = [b, a + b];
  }
  return a;
}
```

Explanation: The function uses a loop to calculate the nth Fibonacci number using two variables (a and b) to keep track of the current and previous Fibonacci numbers. It updates these variables in each iteration using destructuring assignment.

Write a function that returns the maximum value in an array without using the Math.max() method in JavaScript.

```
function max(arr) {
  let max = arr[0];
  for (let i = 1; i < arr.length; i++) {
    if (arr[i] > max) {
      max = arr[i];
    }
  }
  return max;
}
```

Explanation: The function uses a loop to iterate over the input array and update a variable (max) to the largest value encountered so far.

Write a function that checks whether a given string is a palindrome (i.e., reads the same backwards and forwards) in JavaScript.

```
function isPalindrome(str) {
```

```
  const reversed = str.split("").reverse().join("");
  return str === reversed;
}
```

Explanation: The function first uses the split() method to convert the input string to an array of characters. Then, it uses the reverse() method to reverse the order of the characters in the array. Finally, it uses the join() method to convert the reversed array back to a string and compares it to the original string.

Write a function that returns the first non-repeated character in a given string in JavaScript.

```
function firstNonRepeated(str) {
  const counts = new Map();
  for (const char of str) {
    counts.set(char, (counts.get(char) || 0) + 1);
  }
  for (const char of str) {
    if (counts.get(char) === 1) {
      return char;
    }
  }
  return null;
}
```

Explanation: The function uses a Map object to count the frequency of each character in the input string. It then iterates over the string again and returns the first character with a count of 1 (i.e., a character that appears exactly once).

Write a function that generates a random string of a given length in JavaScript.

```
function randomString(length) {
  const chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01
23456789";
  let result = "";
  for (let i = 0; i < length; i++) {
    result += chars.charAt(Math.floor(Math.random() *
chars.length));
  }
  return result;
}
```

Explanation: The function uses a loop to generate a random character from a string of all possible characters (letters and digits) for each position in the final string.

Write a function that sorts an array of objects by a specified key in JavaScript.

```
function sortByKey(arr, key) {
  return arr.sort((a, b) => {
    if (a[key] < b[key]) return -1;
    if (a[key] > b[key]) return 1;
    return 0;
  });
}
```

Explanation: The function uses the Array.sort() method to sort the input array by a specified key in each object. It takes two parameters, a and b, representing two objects being compared. The if statements in the sort() function compare the values of the specified key in each object (a[key] and b[key]) and return -1, 1, or 0 depending on whether the first object should come before, after, or at the same position as the second object in the sorted array.

Write a function that flattens a nested array of any depth in JavaScript.

```
function flatten(arr) {
  return arr.reduce((acc, val) => Array.isArray(val) ?
acc.concat(flatten(val)) : acc.concat(val), []);
}
```

Explanation: The function uses the Array.reduce() method to flatten a nested array of any depth by recursively concatenating the elements of sub-arrays into the accumulator array (acc). The conditional (Array.isArray(val)) checks if an element is an array and applies the same operation to it if so, otherwise it just concatenates the element to the accumulator array.

Write a function that finds the common elements between two arrays in JavaScript.

```
function findCommon(arr1, arr2) {
  const set1 = new Set(arr1);
```

```
  const set2 = new Set(arr2);
  return [...set1].filter(x => set2.has(x));
}
```

Explanation: The function first creates two Set objects from the input arrays (arr1 and arr2). Then, it uses the filter() method to return an array of elements that appear in both sets.