**Chapter 12 - Arrays**

**Array:**

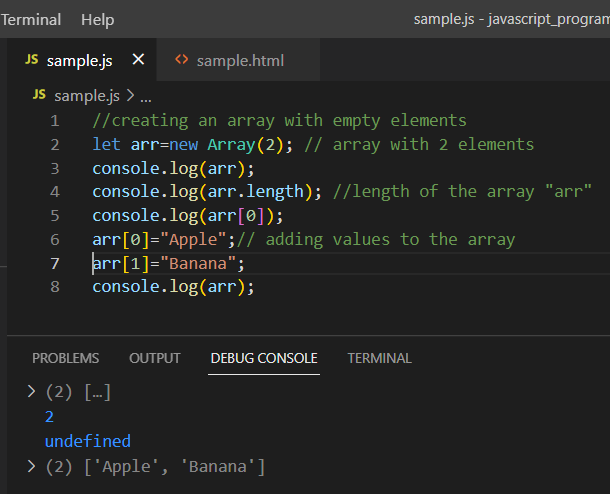
The array is a collection of multiple items under a single variable name.

* Javascript arrays are resizable and can contain a mix of different data types.
* They are accessed using non-negative integers
* They are zero-indexed; i.e first element is at index 0, next is at index 1 and the last element is at length-1.

**Array():**

The Array() constructor is used to create Array objects.

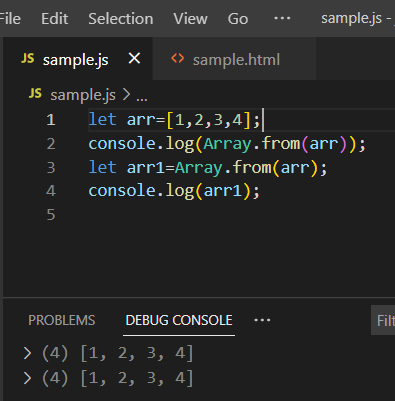
let arr=new Array[element0,element1,....,elementN];



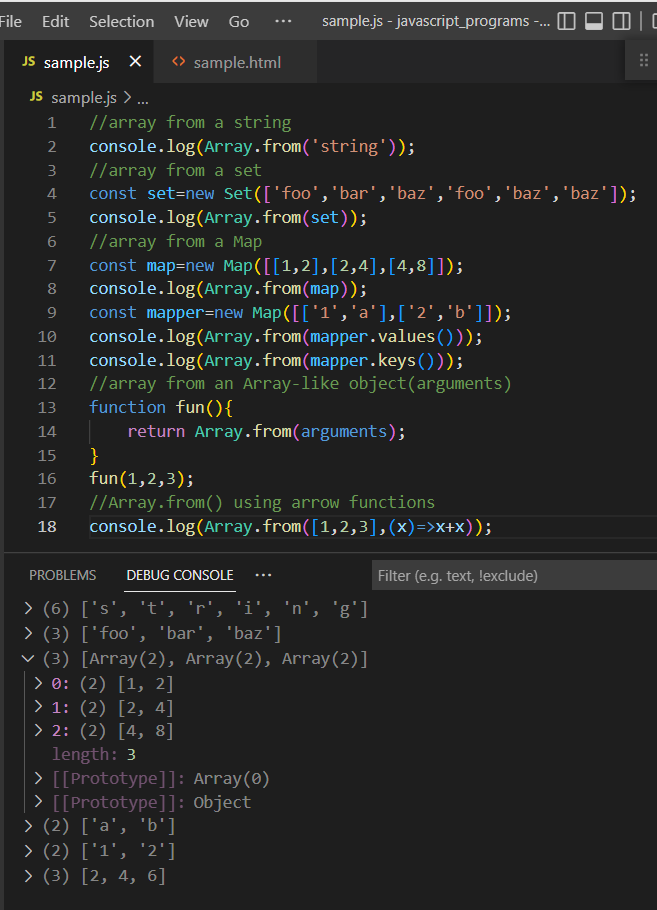
In the above example, arr is an array created with size=2. Later the values are added to the array. The length of the array is accessed with the length parameter. I.e arr.length();

**Array.from()**:

Array.from() is a static method that creates a new, shallow-copied Array instance from an iterable or array-like object.

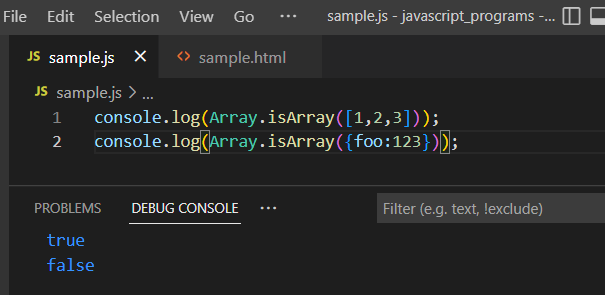


Array.from() can also be used for string, set, Map, functions and arrow functions.



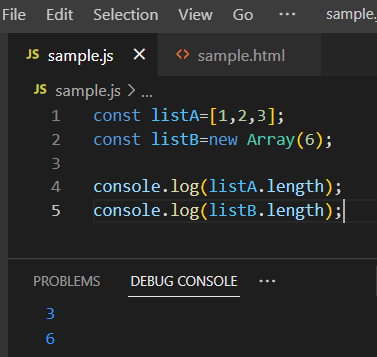
**Array.isArray():**

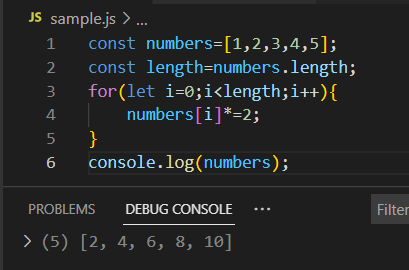
Array.isArray() method determines whether the passed value is an array or not.



**Array.prototype.length**

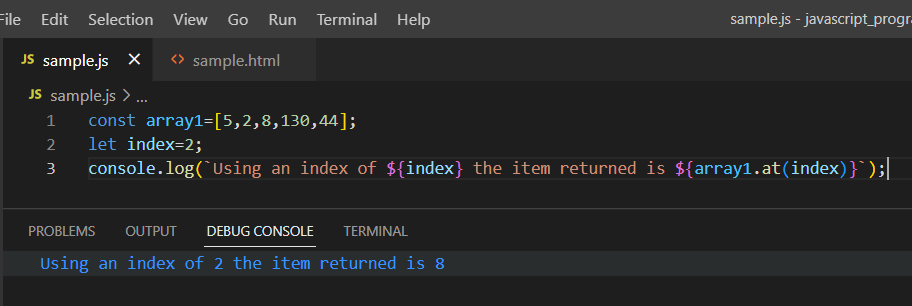
This allows to find the length of the array.

****

****

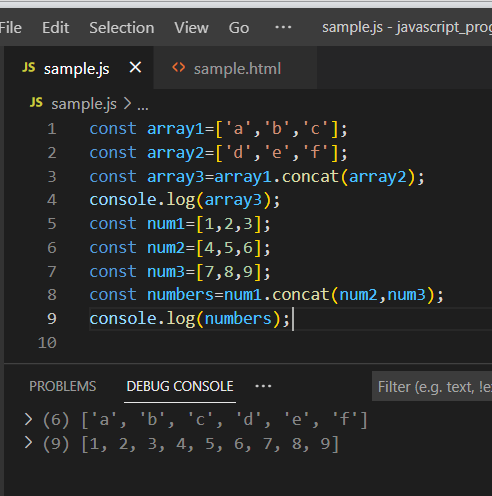
**Array.prototype.at():**

The at() method takes an integer value and returns the item at that index. This accepts negative integers which count back from the last item.



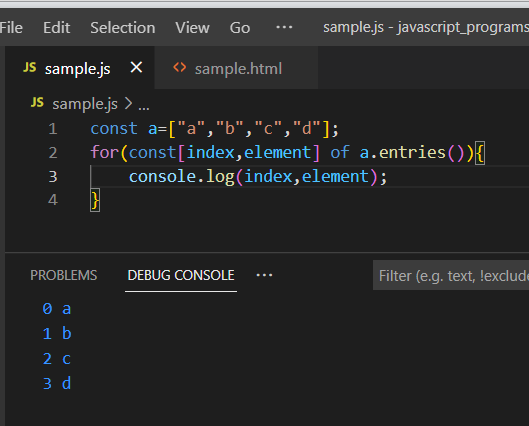
**Array.prototype.concat():**

The concat() method is used to merge two or more arrays.



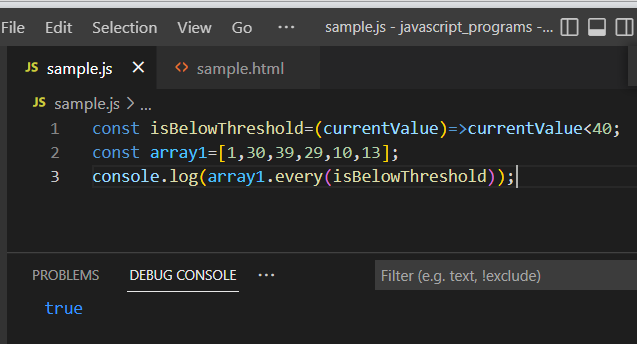
**Array.prototype.entries():**

The entries() method returns a new Array Iterator object that contains the key/value pairs for eah index in the array.



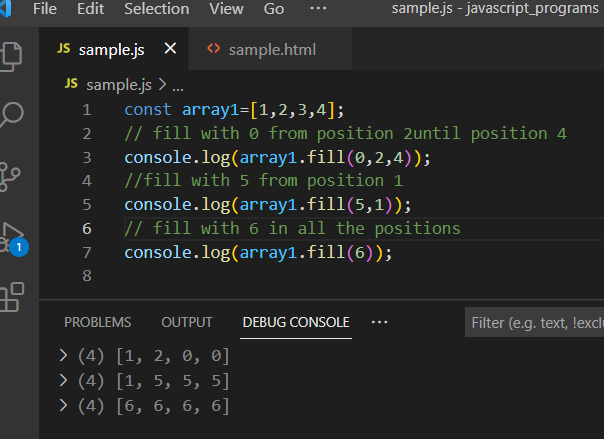
**Array.prototype.every():**

The every() method tests whether all elements in the array pass the test implemented by the provided function.



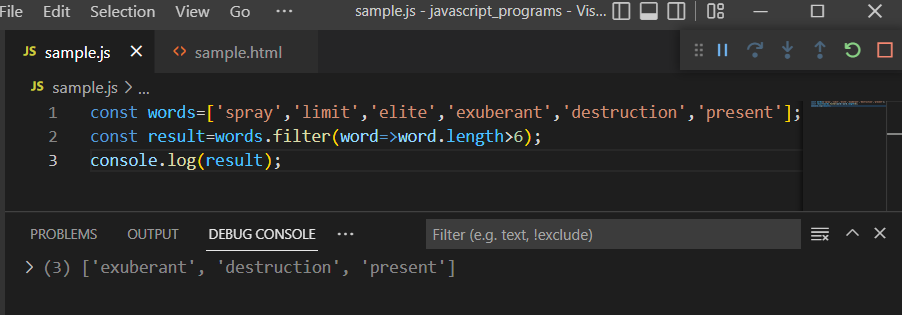
**Array.prototype.fill():**

The fill() method changes all elements in an array to a static value, from a start index(default 0) to an end index(default array.length). It returns the modified array.



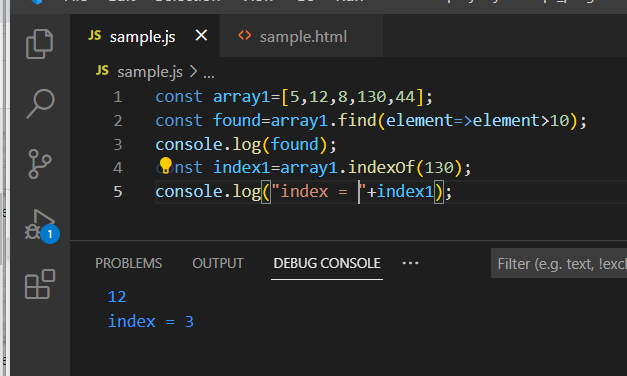
**Array.prototype.filter():**

The filter() method creates a shallow copy of a portion of a given array, filtered down to just the elements from the given array that pass the test implemented by the provided function.



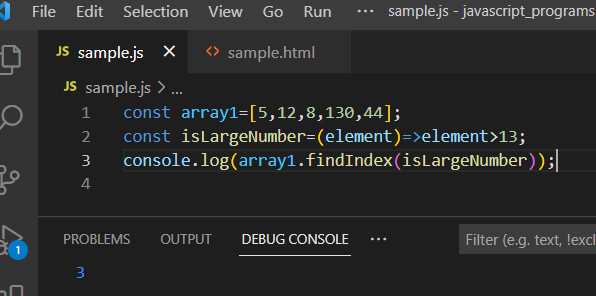
**Array.prototype.find():**

The find() method returns the first element in the array that satisfies the condition.



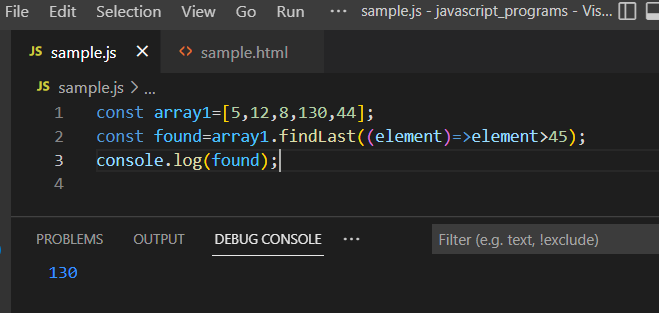
**Array.prototype.findIndex():**

The findIndex() method returns the index of the first element in an array that satisfies the provided condition.



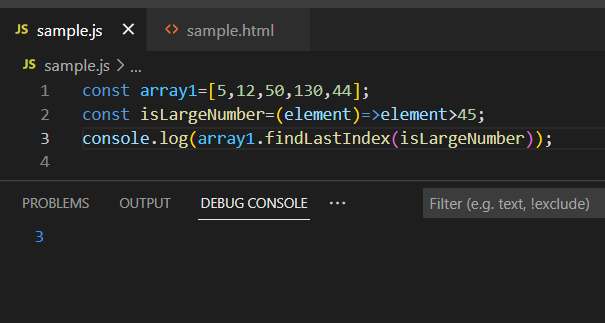
**Array.prototype.findLast():**

The findLast() method returns the value of the last element in an array that satisfies the condition.

****

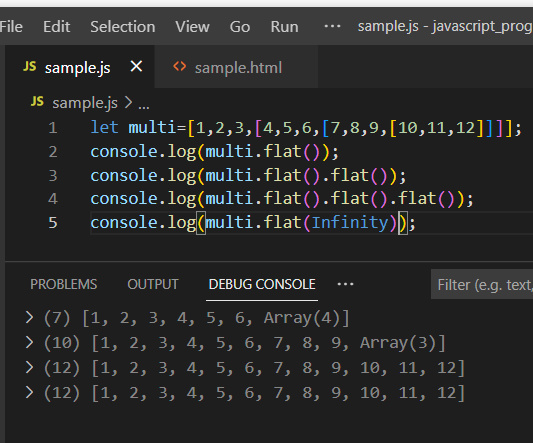
**Array.prototype.findLastIndex():**

The findLastIndex() method returns the index of the last element in an array that satisfies the condition.



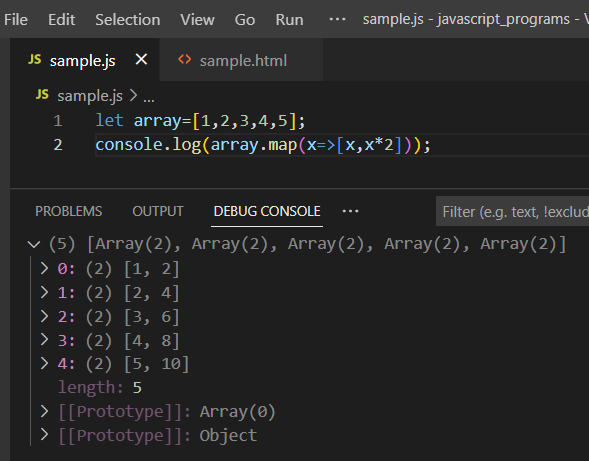
**Array.prototype.flat():**

The flat() method creates a new array with all sub-array elements concatenated into it recursively up to the specified depth.



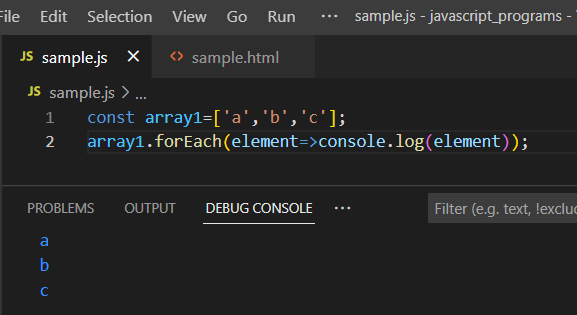
**Array.prototype.flatMap():**

It returns a new array formed by applying a given callback function to each element of the calling array and then flattening the result by one level.

****

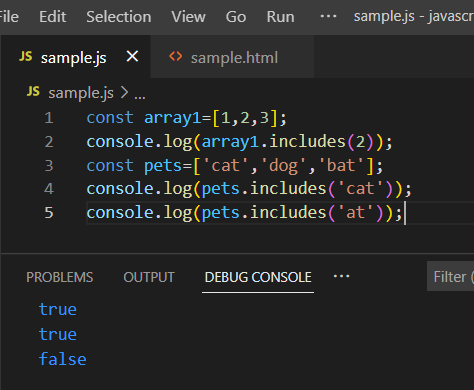
**Array.prototype.forEach():**

Calls a function for each element in the calling array.



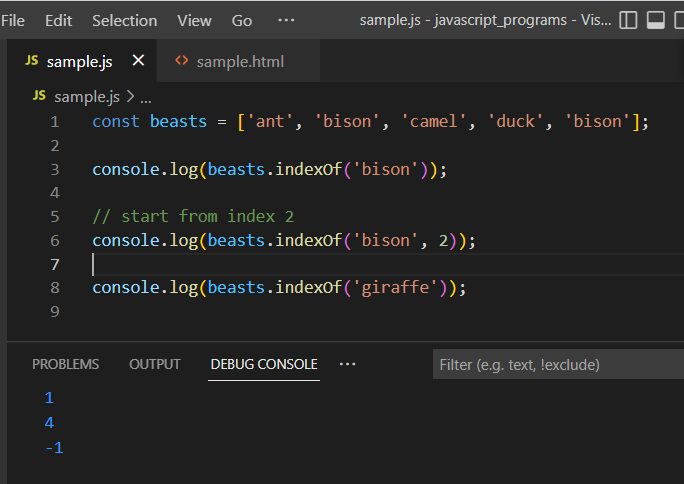
**Array.prototype.includes():**

Determines whether the calling array contains a value, returning true or false as appropriate.



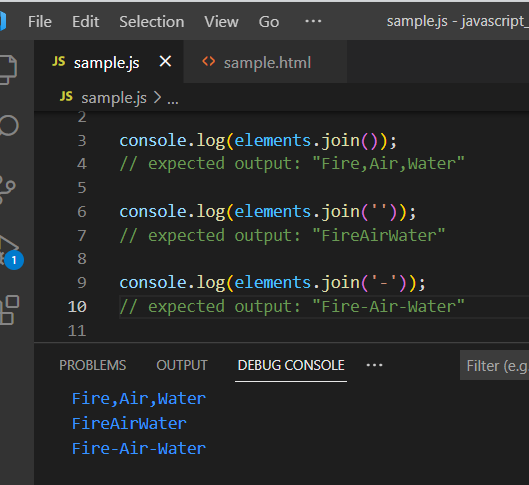
**Array.prototype.indexOf():**

Returns the first (least) index at which a given element can be found in the calling array.



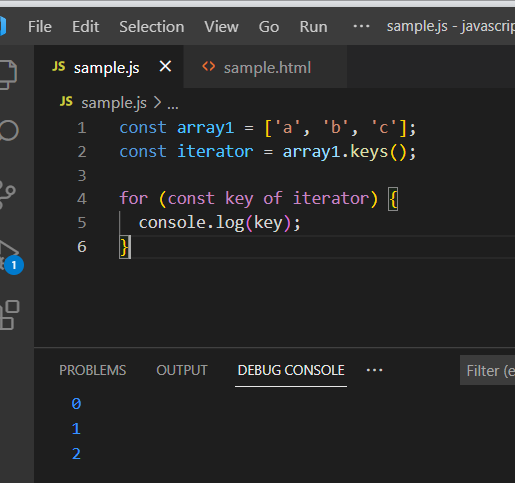
**Array.prototype.join():**

Joins all elements of an array into a string.



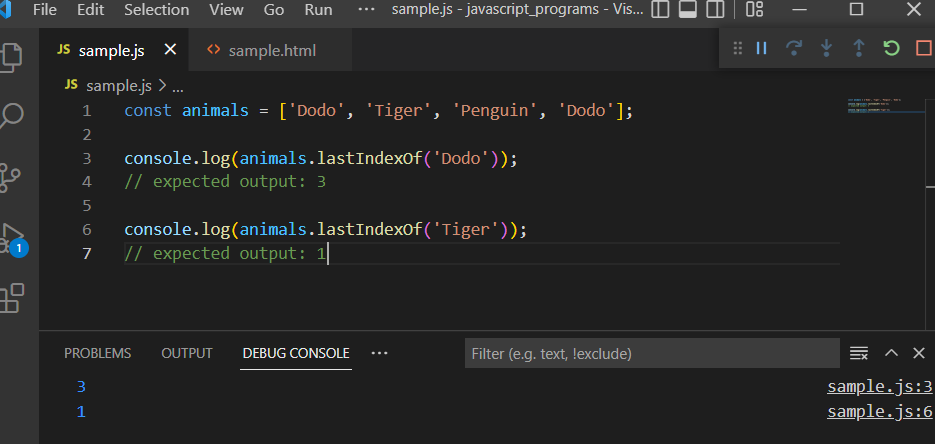
**Array.prototype.keys():**

Returns a new array iterator that contains the keys for each index in the calling array.



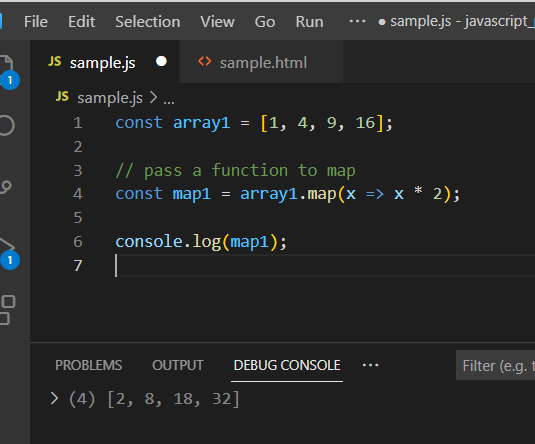
**Array.prototype.lastIndexOf():**

Returns the last(greatest) index at which a given element can be found in the calling array, or -1 if none is found.



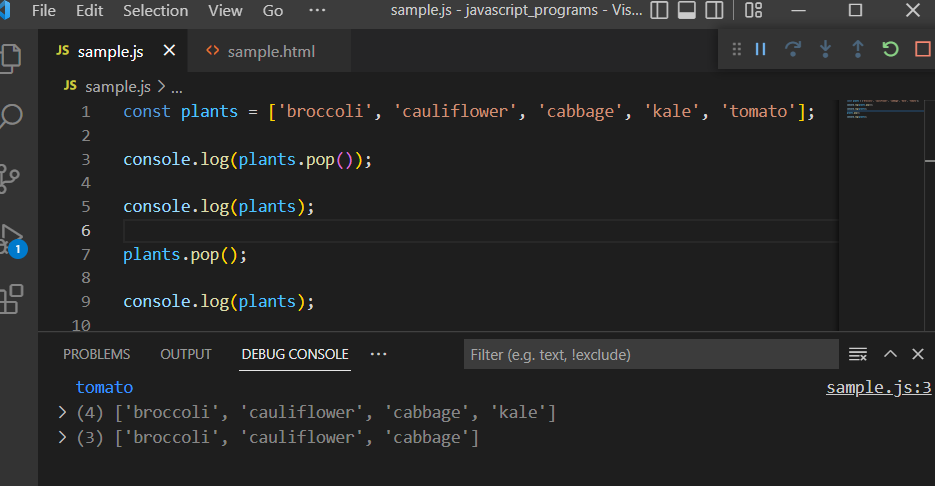
**Array.prototype.map():**

Returns a new array containing the results of invoking a function on every element in the calling array.



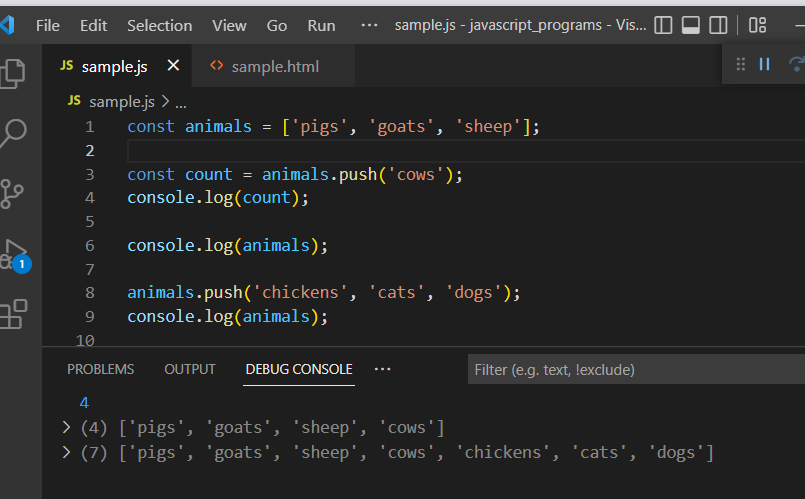
**Array.prototype.pop():**

Removes the last element from an array and returns that element.



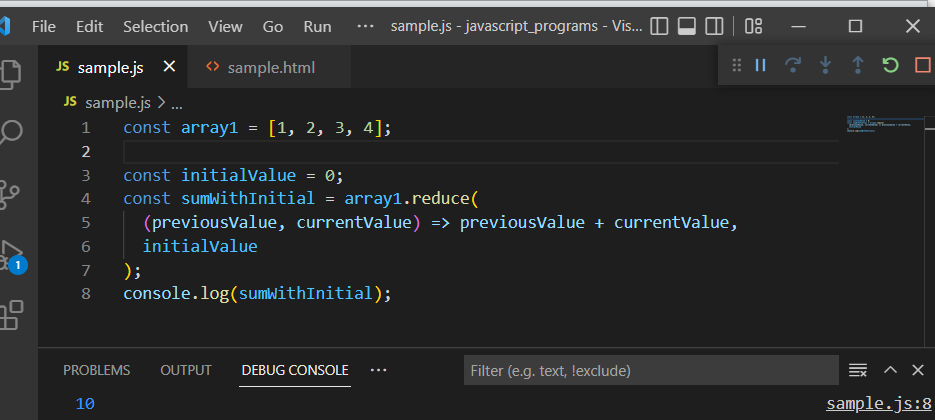
**Array.prototype.push():**

Adds one or more elements to the end of an array and returns the new length of the array.



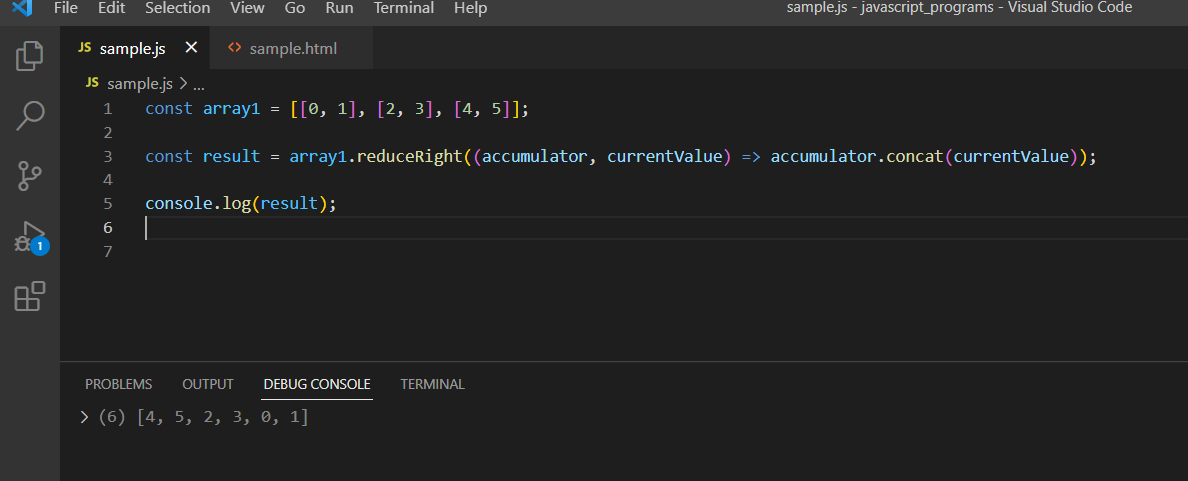
**Array.prototype.reduce():**

Executes a user-supplied “reducer” callback function on each element of the array to reduce it to a single value.



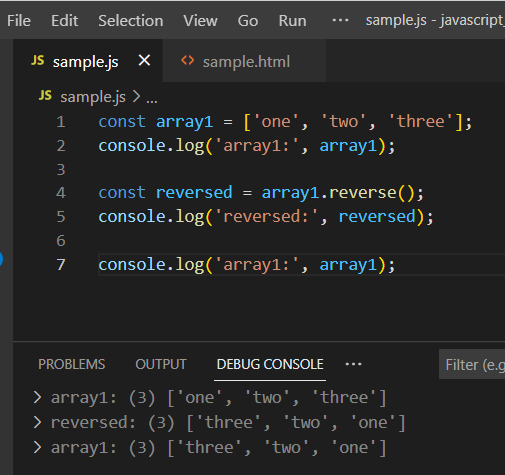
**Array.prototype.reduceRight():**

Executes a user-supplied “reducer” callback function on each element of the array to reduce it to a single value.



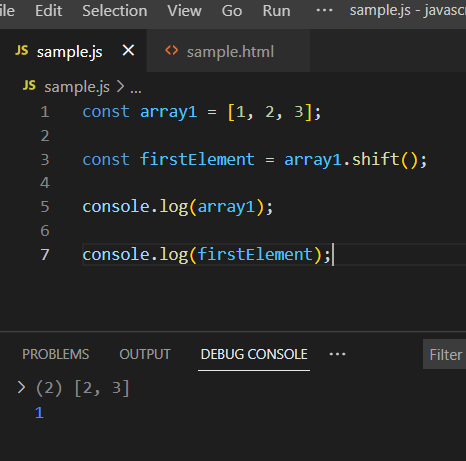
**Array.prototype.reverse():**

Reverses the order of the elements of an array in place.



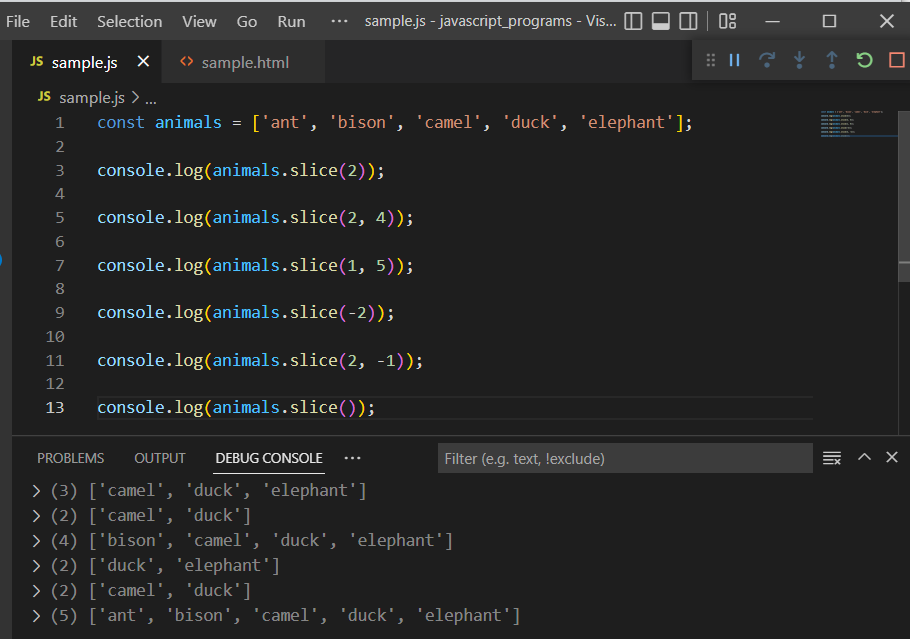
**Array.prototype.shift():**

Removes the first element from an array and returns a new array.



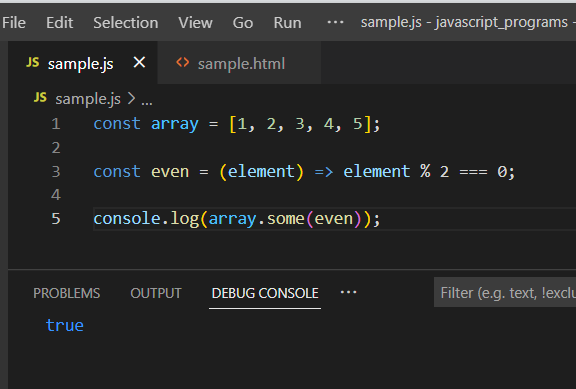
**Array.prototype.slice():**

Extracts a section of the calling array and returns a new array.



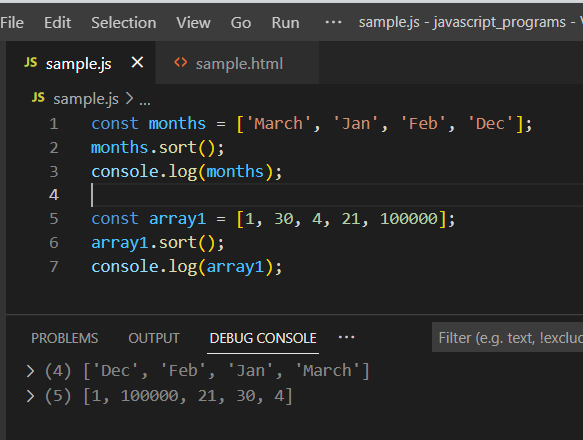
**Array.prototype.some():**

Returns true if at least one element in the calling array satisfies the condition.



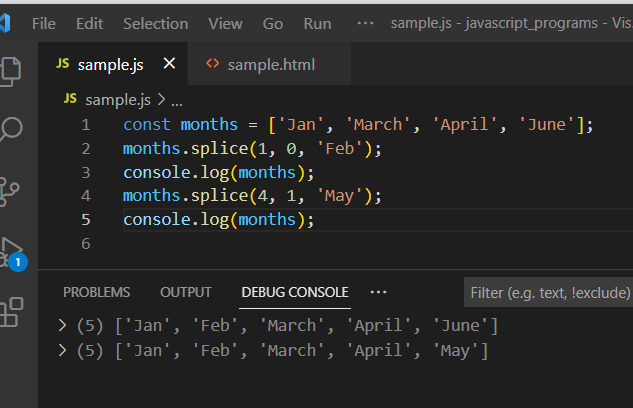
**Array.prototype.sort():**

Sorts the elements of an array in place and returns the array.

****

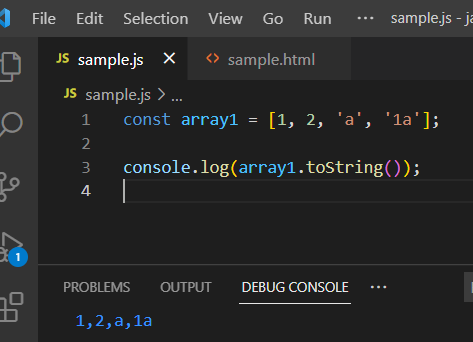
**Array.prototype.splice():**

Adds and/or removes elements from an array.



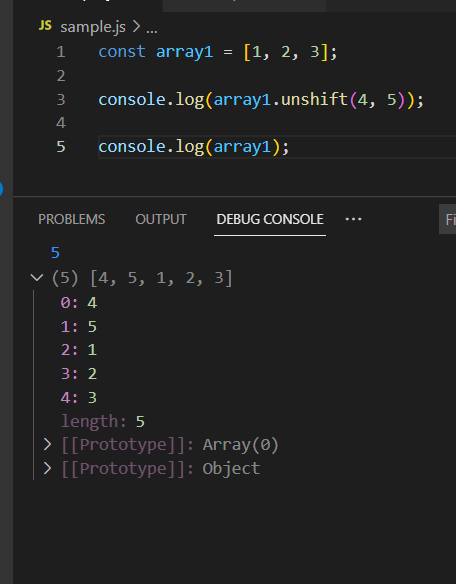
**Array.prototype.toString():**

Returns a string representing the calling array and its elements.overrides the object.prototype.toString() method.



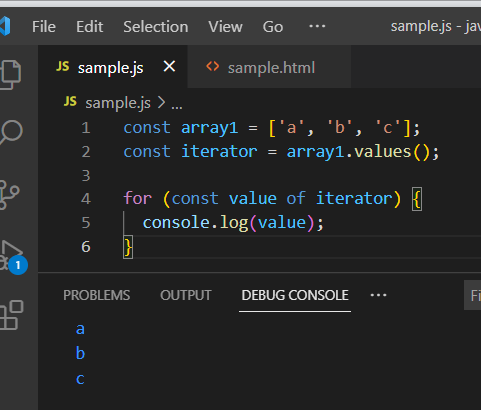
**Array.prototype.unshift():**

Adds one or more elements to the front of an array and returns the new length of the array.



**Array.prototype.values():**

Returns the new array iterator object that contains the values for each index in the array.

****