**Data Structures: Arrays**

Here are two problems that involve the use of arrays. For each problem two different algorithms are suggested. Implement the two algorithms and consider the running time behaviour of each.

**Existing Files**

| **File path** | **Description** |
| --- | --- |
| src/missingNumber.js | Implement your solution to the missing number problem in this file. |
| src/minimumAndMaximum.js | Implement your solution to the minimum and maximum problem in this file. |
| \_\_tests\_\_/missingNumber.test.js | Tests for the missing number problem. you do not need to edit this file. |
| \_\_tests\_\_/minimumAndMaximum.test.js | Tests for the minimum and maximum problem. you do not need to edit this file. |

**Tasks**

**Find the missing number**

Given an array of length n-1 containing all the integers from 1 to n with one number missing, find the missing number. The array is not necessarily sorted.

For example, given the array: [2, 5, 1, 4] you would say that the number 3 was missing.

Implement two different algorithms for solving this problem.

1. Brute force  
   Use a loop to consider every possible integer from 1 to n and check if that integer exists in the array or not.
2. sum and subtract  
   Find the expected sum of all integers from 1 to n. Then find the actual sum of the numbers in the array. The difference is the missing number.

**Find maximum and minimum**

Given an unsorted array of numbers, find the maximum and minimum numbers in the array. For example, given the array [8, 3, 7, 9, 4, 1, 2, 5, 6] your function should return an output array with two values like this: [1, 9], where the first element is the minimum and the second element is the maximum. If the given array is empty return the empty array [].

Keep in mind that numbers in the array may also be negative.

Implement two different algorithms to solve this problem.

1. Using sort  
   Start by sorting the array using the built-in array sort() method then simply return the first and last elements of the array.  
   Note that sort() compares the string values of array elements, so the results will not be in numerical order. E.g. [5,0,-3,510000,6].sort() outputs [ -3, 0, 5, 510000, 6 ]. The sort() method will work to pass the test cases, since the numbers in the tests range within -9 and 9.
2. Using iteration  
   Iterate over the entire array and track the smallest (minimum) and largest (maximum) numbers found so far.