BitByteBard

88. Merge Sorted Array

Solution 1

```
var merge = function(nums1, m, nums2, n)
{
  for(let i=m, j=0; j<n; i++,j++){
      nums1[i] = nums2[j];
  }
  nums1.sort((a, b) => a - b);
}
```

Time Complexity: O(nlogn)
Space Complexity: O(1)

1

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

88. Merge Sorted Array

Solution 2

```
var merge = function(nums1, m, nums2, n) {
  let i = m-1;
  let j = n-1;
  let k = m+n-1;
  while(j>=0 && i>=0){
    if(nums1[i] < nums2[j]){
      nums1[k] = nums2[j];
      j--;
      k--;
    }else{
      nums1[k] = nums1[i];
      i--;
      k--;
    }
  }
  while(j>=0){
    nums1[k] = nums2[j];
    j--;
    k--;
  }
};
```

Time Complexity: O(n)

Space Complexity: O(1)

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

27. Remove Element

Solution 1

```
var removeElement = function(nums, val) {
  let k = 0;
 for(let i=0; i<nums.length; i++){
    if(val !== nums[i]){
      nums[k] = nums[i];
      k++;
    }
  }
  return k;
};
```

Time Complexity: O(n)
Space Complexity: O(1)

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

26. Remove Duplicates from Sorted Array

Solution 1

```
var removeDuplicates = function(nums) {
  let flag = 0;
   for(let i=1; i<nums.length; i++){
    if(nums[i]!=nums[flag]){
      flag++;
      nums[flag] = nums[i];
    }
  }

  return flag + 1;
};
```

Time Complexity: O(n)

Space Complexity: O(1)

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

80. Remove Duplicates from Sorted Array II

Solution 1

```
var removeDuplicates = function(nums) {
  let flag = 2;
   for(let i=2; i<nums.length; i++){
    if(nums[i]!=nums[flag-2]){
      nums[flag] = nums[i];
      flag++;
    }
  }
  return flag;
};
```

Time Complexity: O(n)
Space Complexity: O(1)

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

169. Majority Element

Solution 1

```
var majorityElement = function(nums) {
  let numsSort = nums.sort();
  const n = Math.ceil(nums.length/2);
  let element = numsSort[n];
  return element;
};
```

Time Complexity: O(1)

Space Complexity: O(nlogn)

#TopInterviewQuestion #Programming #Leetcode

BitByteBard

169. Majority Element

Solution 2

```
var majorityElement = function(nums) {
  let element = 0;
  let count = 0;

  for(let i=0; i<nums.length; i++){
    if(count===0){
      element = nums[i];
      count++;
    }else if(element==nums[i]){
      count++;
    }else{
      count--;
    }
  }
  return element;
};
```

Time Complexity: O(n)

Space Complexity: O(1)