Top Interview 150 | Run | Submit | 0 | Premiu

Description | Editorial | Solutions | Submissions

88. Merge Sorted Array

Solved

Easy | Topics | Companies | Hint

You are given two integer arrays `nums1` and `nums2`, sorted in **non-decreasing order**, and two integers `m` and `n`, representing the number of elements in `nums1` and `nums2` respectively.

Merge `nums1` and `nums2` into a single array sorted in **non-decreasing order**.

The final sorted array should not be returned by the function, but instead be *stored inside the array* `nums1`. To accommodate this, `nums1` has a length of `m + n`, where the first `m` elements denote the elements that should be merged, and the last `n` elements are set to `0` and should be ignored. `nums2` has a length of `n`.

Example 1:

```
Input: nums1 = [1,2,3,0,0,0], m = 3, nums2 = [2,5,6], n = 3
Output: [1,2,2,3,5,6]
Explanation: The arrays we are merging are [1,2,3] and [2,5,6].
The result of the merge is [1,2,2,3,5,6] with the underlined elements coming from nums1.
```

Code

C++ | Auto

```cpp
class Solution {
public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
      if(m>0){  int nums3[m];
        for(int i=0;i<m;i++)
        nums3[i]=nums1[i];
        int j=0,k=0,x=0;
        while(j<m && k<n){
            if(nums3[j]<=nums2[k]){nums1[x]=nums3[j]; j++;x++;}
            else{nums1[x]=nums2[k]; k++;x++;}
        }
        while(k<n){ nums1[x]=nums2[k];k++;x++;}
        while(j<m){ nums1[x]=nums3[j];j++;x++;}}
        else{ for(int p=0;p<n;p++) nums1[p]=nums2[p];}

    }
};
```

Description | Editorial | Solutions | Submissions

27. Remove Element

Solved

Easy | Topics | Companies | Hint

Given an integer array `nums` and an integer `val`, remove all occurrences of `val` in `nums` **in-place**. The order of the elements may be changed. Then return *the number of elements in* `nums` *which are not equal to* `val`.

Consider the number of elements in `nums` which are not equal to `val` be `k`, to get accepted, you need to do the following things:

- Change the array `nums` such that the first `k` elements of `nums` contain the elements which are not equal to `val`. The remaining elements of `nums` are not important as well as the size of `nums`.
- Return `k`.

Code

C++ | Auto

```cpp
class Solution {
public:
    int removeElement(vector<int>& nums, int val) {
      int k=0;
      for(int i=0;i<nums.size();i++){
        if(nums[i]!=val){nums[k]=nums[i]; k++;}
      }
    return k;}
};
```

26. Remove Duplicates from Sorted Array

Solved

Easy | Topics | Companies | Hint

Given an integer array `nums` sorted in **non-decreasing order**, remove the duplicates in-place such that each unique element appears only **once**. The **relative order** of the elements should be kept the **same**. Then return *the number of unique elements in* `nums`.

Consider the number of unique elements of `nums` to be `k`, to get accepted, you need to do the following things:

- Change the array `nums` such that the first `k` elements of `nums` contain the unique elements in the order they were present in `nums` initially. The remaining elements of `nums` are not important as well as the size of `nums`.
- Return `k`.

Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array
int[] expectedNums = [...]; // The expected answer with correct length
```

C++ Auto

```cpp
class Solution {
public:
    int removeDuplicates(vector<int>& nums) {
        int n=nums.size();
        if(n==0) return 0;
        else{
            int k=1;
            for(int i=1;i<n;i++){
                if(nums[i]!=nums[i-1]){nums[k]=nums[i]; k++;}
            } return k;
        }

    }
};
```

80. Remove Duplicates from Sorted Array II

Solved

Medium | Topics | Companies

Given an integer array `nums` sorted in **non-decreasing order**, remove some duplicates in-place such that each unique element appears **at most twice**. The **relative order** of the elements should be kept the **same**.

Since it is impossible to change the length of the array in some languages, you must instead have the result be placed in the **first part** of the array `nums`. More formally, if there are `k` elements after removing the duplicates, then the first `k` elements of `nums` should hold the final result. It does not matter what you leave beyond the first `k` elements.

Return `k` *after placing the final result in the first* `k` *slots of* `nums`.

Do **not** allocate extra space for another array. You must do this by **modifying the input array** in-place with O(1) extra memory.

Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array
int[] expectedNums = [...]; // The expected answer with correct length

int k = removeDuplicates(nums); // Calls your implementation

assert k == expectedNums.length;
for (int i = 0; i < k; i++) {
    assert nums[i] == expectedNums[i];
```

```
class Solution {
public:
    int removeDuplicates(vector<int>& nums) {
        int n=nums.size();
        if(n==0) return 0;
        else if(n==1){return 1;}
        else if(n==2){return 2;}
        else{
            int k=0,i;
            for(i=0;i<n;i++){
                int j,c=0;
                for(j=i;j<n;j++){
                    if(nums[j]!=nums[i]) break;
                    c++;
                }
                if(c>=3){
                    nums[k]=nums[i];
                    nums[k+1]=nums[i];
                    i=i+c-1;
                    k=k+2;
                } else if(c==2){
                    nums[k]=nums[i];
                    nums[k+1]=nums[i];
                    i=i+1;
                    k=k+2;
                } else{
                    nums[k]=nums[i];
                    k++;}
```

Saved

Ln 1, Col

Description | Editorial | Solutions | Submissions

189. Rotate Array

Solved

Medium | Topics | Companies | Hint

Given an integer array `nums`, rotate the array to the right by `k` steps, where `k` is non-negative.

Example 1:

```
Input: nums = [1,2,3,4,5,6,7], k = 3
Output: [5,6,7,1,2,3,4]
Explanation:
rotate 1 steps to the right: [7,1,2,3,4,5,6]
rotate 2 steps to the right: [6,7,1,2,3,4,5]
rotate 3 steps to the right: [5,6,7,1,2,3,4]
```

Example 2:

```
Input: nums = [-1,-100,3,99], k = 2
Output: [3,99,-1,-100]
Explanation:
rotate 1 steps to the right: [99,-1,-100,3]
rotate 2 steps to the right: [3,99,-1,-100]
```

Constraints:

Code

C++ | Auto

```cpp
class Solution {
public:
    void rotate(vector<int>& nums, int k) {
        int n=nums.size();
        int nums1[n];
        for(int i=0;i<n;i++){
        if(k<n) nums1[i]=nums[(i+n-k)%n];
        else nums1[i]=nums[(i+n-(k%n))%n];}
        for(int j=0;j<n;j++)
        nums[j]=nums1[j];



    }
};
```

Restored from local | Upgrade to Cloud Saving

121. Best Time to Buy and Sell Stock

Solved

Easy | Topics | Companies

You are given an array `prices` where `prices[i]` is the price of a given stock on the ith day.

You want to maximize your profit by choosing a **single day** to buy one stock and choosing a **different day in the future** to sell that stock.

Return *the maximum profit you can achieve from this transaction*. If you cannot achieve any profit, return `0`.

Example 1:

```
Input: prices = [7,1,5,3,6,4]
Output: 5
Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1
= 5.
Note that buying on day 2 and selling on day 1 is not allowed because you must
buy before you sell.
```

Example 2:

```
Input: prices = [7,6,4,3,1]
Output: 0
Explanation: In this case, no transactions are done and the max profit = 0.
```

C++ Auto

```cpp
    // }

  // }
  int maxProfit(vector<int>& prices) {
      //int n=prices.size();
      //if(n==1) return 0;
       //else{
       //if(maxdiff(prices,n)>0) return maxdiff(prices,n);
       //else return 0;
       // }
       int n=prices.size();
       if(n==1) return 0;
       else{
          int l=0,r=1,max1=0;
          while(r<n){
              if(prices[r]>prices[l]){
                  max1=max(prices[r]-prices[l],max1);

              } else{l=r;}
              r++;
          } return max1;
       }
       }
};
```

Saved

55. Jump Game

Solved

Medium | Topics | Companies

You are given an integer array `nums`. You are initially positioned at the array's **first index**, and each element in the array represents your maximum jump length at that position.

Return `true` *if you can reach the last index, or* `false` *otherwise.*

Example 1:

```
Input: nums = [2,3,1,1,4]
Output: true
Explanation: Jump 1 step from index 0 to 1, then 3 steps to the last index.
```

Example 2:

```
Input: nums = [3,2,1,0,4]
Output: false
Explanation: You will always arrive at index 3 no matter what. Its maximum jump length is 0, which makes it impossible to reach the last index.
```

Constraints:

C++ | Auto

```cpp
class Solution {
public:
    bool canJump(vector<int>& nums) {
        int n=nums.size();
        int goal=n-1;
        for(int i=n-2;i>=0;--i){
            if(nums[i]+i>=goal) goal=i;
        } if(goal==0) return true;
        else return false;

    }
};
```

45. Jump Game II

Solved

Medium | Topics | Companies

You are given a **0-indexed** array of integers `nums` of length `n`. You are initially positioned at `nums[0]`.

Each element `nums[i]` represents the maximum length of a forward jump from index `i`. In other words, if you are at `nums[i]`, you can jump to any `nums[i + j]` where:

- `0 <= j <= nums[i]` and
- `i + j < n`

Return *the minimum number of jumps to reach* `nums[n - 1]`. The test cases are generated such that you can reach `nums[n - 1]`.

Example 1:

```
Input: nums = [2,3,1,1,4]
Output: 2
Explanation: The minimum number of jumps to reach the last index is 2. Jump 1 step from index 0 to 1, then 3 steps to the last index.
```

Example 2:

```
Input: nums = [2,3,0,1,4]
Output: 2
```

C++ | Auto

```cpp
class Solution {
public:
    int jump(vector<int>& nums) {



        int n = nums.size();
        if (n <= 1) return 0;

        int jumps = 0, currentEnd = 0, farthest = 0;

        for (int i = 0; i < n - 1; ++i) {
            farthest = max(farthest, i + nums[i]);

            if (i == currentEnd) {
                jumps++;
                currentEnd = farthest;

                if (currentEnd >= n - 1) {
                    break;
                }
            }
        }

        return jumps;


```

Saved

274. H-Index

Solved

Medium | Topics | Companies | Hint

Given an array of integers `citations` where `citations[i]` is the number of citations a researcher received for their `i^th` paper, return *the researcher's h-index.*

According to the definition of h-index on Wikipedia: The h-index is defined as the maximum value of `h` such that the given researcher has published at least `h` papers that have each been cited at least `h` times.

Example 1:

```
Input: citations = [3,0,6,1,5]
Output: 3
Explanation: [3,0,6,1,5] means the researcher has 5 papers in total and each of
them had received 3, 0, 6, 1, 5 citations respectively.
Since the researcher has 3 papers with at least 3 citations each and the
remaining two with no more than 3 citations each, their h-index is 3.
```

Example 2:

```
Input: citations = [1,3,1]
Output: 1
```

Python | Auto

```python
class Solution(object):
    def hIndex(self, citations):
        n=len(citations)
        c=0
        citations=sorted(citations,reverse=True)
        for i in range(1,n+1,1):
            if(citations[i-1]<i):
                c=1
                break
        if(c==1):
            return i-1
        else:
            return n
```