

双指针

笔记本: LeetCode
创建时间: 2020/7/4 19:19
作者: 粥粥

更新时间: 2020/7/7 23:24

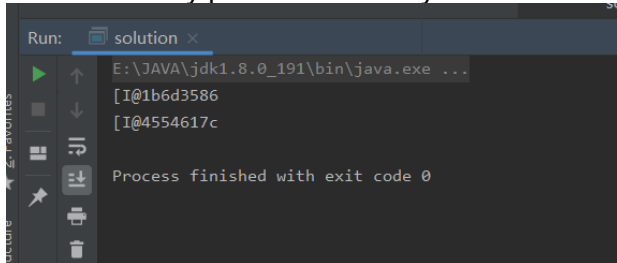
1. two sum (Input array is sorted)

<https://leetcode.com/problems/two-sum-ii-input-array-is-sorted/description/>

One point from the start and one point from the end to traverse the array.

Compare sum to target. Note the boundary conditions.

Do not directly print out the array or the address would be got.



TC: $O(n)$

Description

Solution

Submissions

Discuss (999+)

167. Two Sum II - Input array is sorted

Easy 1645 606 Add to List Share

Given an array of integers that is already **sorted in ascending order**, find two numbers such that they add up to a specific target number.

The function twoSum should return indices of the two numbers such that they add up to the target, where index1 must be less than index2.

Note:

- Your returned answers (both index1 and index2) are not zero-based.
- You may assume that each input would have *exactly* one solution and you may not use the *same* element twice.

Example:

Input: numbers = [2,7,11,15], target = 9
Output: [1,2]
Explanation: The sum of 2 and 7 is 9. Therefore index1 = 1, index2 = 2.

Java

Autocomplete

```
1 class Solution {
2     public int[] twoSum(int[] numbers, int target) {
3         if(numbers==null) return null;
4         int i=0,j=numbers.length-1;
5         while(i<j){
6             if(numbers[i]+numbers[j]==target) {return new int[]{i+1,j+1};}
7             else if(numbers[i]+numbers[j]<target) {i++;}
8             else {j--;}
9         }
10        return null;
11    }
12 }
```

2. sum of square numbers

<https://leetcode.com/problems/sum-of-square-numbers/>

See this as finding numbers in a sorted array from 0 to target. It turns to be same question as the first one.

Optimize the length of the array. That is an array from 0 to \sqrt{c} .

TC: $O(n)$

[Explore](#)
[Problems](#)
[Mock](#)
[Contest](#)
[Articles](#)
[Discuss](#)
[Store](#)

June LeetCode Challenge!
Premium

Description
Solution
Submissions
Discuss (272)

633. Sum of Square Numbers
Easy
517
325
Add to List
Share

Given a non-negative integer c , your task is to decide whether there're two integers a and b such that $a^2 + b^2 = c$.

Example 1:

Input: 5
Output: True
Explanation: $1^2 + 2^2 = 5$

Example 2:

Input: 3
Output: False

```

1 class Solution {
2     public boolean judgeSquareSum(int c) {
3         if(c<0) return false;
4         int i=0,j=(int)Math.sqrt(c);
5         while(i<=j){
6             if(i*i+j*j == c) return true;
7             else if(i*i+j*j < c) i++;
8             else j--;
9         }
10        return false;
11    }
12 }
13

```

3. reverse vowels of a string

<https://leetcode.com/problems/reverse-vowels-of-a-string/>

List is an interface and can not be instantiation directly.

I can not exchange characters inside a string. So I set a new `char[]` to store the result.

TC: $O(n)$

[Explore](#)
[Problems](#)
[Mock](#)
[Contest](#)
[Articles](#)
[Discuss](#)
[Store](#)

June LeetCode Challenge!
Premium

Description
Solution
Submissions
Discuss (932)

345. Reverse Vowels of a String
Easy
668
1115
Add to List
Share

Write a function that takes a string as input and reverse only the vowels of a string.

Example 1:

Input: "hello"
Output: "holle"

Example 2:

Input: "leetcode"
Output: "leotcede"

Note:
The vowels does not include the letter "y".

```

1 class Solution {
2     public String reverseVowels(String s) {
3         List<Character> vowels= new ArrayList<>(Arrays.asList('a','e','i','o','u','A','E','I','O','U'));
4
5         int i=0,j=s.length()-1;
6         char[] result=new char[s.length()];
7
8         while(i<=j){
9             char ci=s.charAt(i);
10            char cj=s.charAt(j);
11            if(vowels.contains(ci) && vowels.contains(cj)){
12                result[i]=s.charAt(j);
13                result[j]=s.charAt(i);
14                i++;
15                j--;
16            }else if(!vowels.contains(ci)){
17                result[i]=s.charAt(i);
18                i++;
19            }else {
20                result[j]=s.charAt(j);
21                j--;
22            }
23        }
24        return new String(result);
25    }
26 }
27

```

4. valid palindroma

<https://leetcode.com/problems/valid-palindrome-ii/description/>

First have a function to determine that s is a palinfroma or not

one point from start one point from the end, if he character is the same, move two points.

if different,move one pointer and use the function to see if substring a palindroma or not

[Explore](#)
[Problems](#)
[Mock](#)
[Contest](#)
[Articles](#)
[Discuss](#)
[Store](#)

June LeetCode Challenge!
Premium

Description
Solution
Submissions
Discuss (674)
Java
Autocomplete

680. Valid Palindrome II

Easy 1620 107 Add to List Share

Given a non-empty string `s`, you may delete **at most** one character. Judge whether you can make it a palindrome.

Example 1:

Input: "aba"
Output: True

Example 2:

Input: "abca"
Output: True
Explanation: You could delete the character 'c'.

Note:

- The string will only contain lowercase characters a-z. The maximum length of the string is 50000.

```

1 class Solution {
2     public boolean validPalindrome(String s) {
3         int i=0,j=s.length()-1;
4         while(i<j){
5             if(s.charAt(i)==s.charAt(j)){
6                 i++;
7                 j--;
8             }else{
9                 return isPalindrome(s,i+1,j) || isPalindrome(s,i,j-1);
10            }
11        }
12        return true;
13    }
14
15    public boolean isPalindrome(String s, int i,int j){
16        while(i<=j){
17            if(s.charAt(i)!=s.charAt(j)){
18                i++;
19                j--;
20            }else {
21                return false;
22            }
23        }
24        return true;
25    }
26 }
27
28

```

5. merge sorted array

<https://leetcode.com/problems/merge-sorted-array/>

Two point moves from the end of the nums1 and nums2.

[Explore](#)
[Problems](#)
[Mock](#)
[Contest](#)
[Articles](#)
[Discuss](#)
[Store](#)

June LeetCode Challenge!
Premium

Description
Solution
Submissions
Discuss (999+)
Java
Autocomplete

88. Merge Sorted Array

Easy 2203 4158 Add to List Share

Given two sorted integer arrays `nums1` and `nums2`, merge `nums2` into `nums1` as one sorted array.

Note:

- The number of elements initialized in `nums1` and `nums2` are `m` and `n` respectively.
- You may assume that `nums1` has enough space (size that is **equal** to `m + n`) to hold additional elements from `nums2`.

Example:

Input:
`nums1 = [1,2,3,0,0,0]`, `m = 3`
`nums2 = [2,5,6]`, `n = 3`

Output: `[1,2,2,3,5,6]`

```

1 class Solution {
2     public void merge(int[] nums1, int m, int[] nums2, int n) {
3         int i=m-1,j=n-1;
4         int index=m+n-1;
5         while(index>=0){
6             if(i<0){
7                 nums1[index]=nums2[j];
8                 j--;
9             }else if(j<0){
10            }else if(nums1[i]>nums2[j]){
11                nums1[index]=nums1[i];
12                i--;
13            }else{
14                nums1[index]=nums2[j];
15                j--;
16            }
17            index--;
18        }
19    }
20 }
21

```

6. linked list cycle

<https://leetcode.com/problems/linked-list-cycle/>

To find a ring, we can use a map to store node, and if the next node has been shown in the map before, we can see there is a ring.

But this cost space complexity.

So we use two pointer, one is slower and one is faster. One pointer move one step each time and the other one move two steps each time.

If there is a ring, two point will meet.

Day 7

Explore

Problems

Mock

Contest

Articles

Discuss

Store

June LeetCode Challenge!

Premium

Description

Solution

Submissions

Discuss (999+)

141. Linked List Cycle

Easy

2861

478

Add to List

Share

Given a linked list, determine if it has a cycle in it.

To represent a cycle in the given linked list, we use an integer `pos` which represents the position (0-indexed) in the linked list where tail connects to. If `pos` is `-1`, then there is no cycle in the linked list.

Example 1:

Input: head = [3,2,0,-4], pos = 1
Output: true
Explanation: There is a cycle in the linked list, where tail connects to the second node.

Java

Autocomplete

```

1  /**
2   * Definition for singly-linked list.
3   * class ListNode {
4   *     int val;
5   *     ListNode next;
6   *     ListNode(int x) {
7   *         val = x;
8   *         next = null;
9   *     }
10  * }
11  */
12  public class Solution {
13  public boolean hasCycle(ListNode head) {
14      if(head == null) return false;
15      ListNode l1=head, l2=head.next;
16      while(l1!=null && l2 !=null && l2.next!=null){
17          if(l1==l2){
18              return true;
19          }else{
20              l1=l1.next;
21              l2=l2.next.next;
22          }
23      }
24      return false;
25  }
26  }

```

Your previous code was restored from your local storage. [Reset to default](#)

7. Longest word in dictionary through deleting

<https://leetcode.com/problems/longest-word-in-dictionary-through-deleting/>

Use two points to determin if it is a substring
 compareTo(): lexicographical order
 TC:O(n)

Day 7

Explore

Problems

Mock

Contest

Articles

Discuss

Store

June LeetCode Challenge!

Premium

Description

Solution

Submissions

Discuss (297)

524. Longest Word in Dictionary through Deleting

Medium

539

233

Add to List

Share

Given a string and a string dictionary, find the longest string in the dictionary that can be formed by deleting some characters of the given string. If there are more than one possible results, return the longest word with the smallest lexicographical order. If there is no possible result, return the empty string.

Example 1:

Input:
s = "abpcplea", d = ["ale","apple","monkey","plea"]

Output:
"apple"

Example 2:

Input:
s = "abpcplea", d = ["a","b","c"]

Output:
"a"

Java

Autocomplete

```

1  class Solution {
2  public String findLongestWord(String s, List<String> d) {
3      String longestString="";
4      for(String target : d){
5          int l1=longestString.length(),l2=target.length();
6          if(isSubString(s,target)){
7              if(l1<l2 || (l1==l2 && target.compareTo(longestString)<0)){
8                  longestString=target;
9              }
10             }else
11                 continue;
12         }
13         return longestString;
14     }
15
16     public boolean isSubString(String s, String target){
17         int i=0,j=0;
18         while(i<s.length() && j<target.length()){
19             if(s.charAt(i)==target.charAt(j)){
20                 i++;
21                 j++;
22             }else{
23                 i++;
24             }
25         }
26         return j==target.length();
27     }
28 }

```

Your previous code was restored from your local storage. [Reset to default](#)