LeetCode 题解

(Java 版)

By shlip@foxmail.com

目录

1~100

1. 两数之和

分析

难度 易
博客
来源

题目

给定一个整数数组和一个目标值，找出数组中和为目标值的**两个**数。

你可以假设每个输入只对应一种答案，且同样的元素不能被重复利用。

示例:

给定 nums = [2, 7, 11, 15], target = 9

因为 nums[0] + nums[1] = 2 + 7 = 9
所以返回 [0, 1]

解答

```
public class TwoSum {
    public int[] twoSum(int[] nums, int target) {
        int len=nums.length;
        for(int i=0;i<len-1;i++)
            for(int j=i+1;j<len;j++){
                if(nums[i]+nums[j]==target)
                    return new int[]{i,j};
            }

        throw new IllegalArgumentException("No two sum solution");
    }
    public  static void main(String[] args)
    {
        int[] nums={2,7,11,15};
        int target=9;
        TwoSum ts=new TwoSum();
```

```
        int[] result=ts.twoSum(nums,9);
        for (int i=0;i<result.length;i++)
        {
            System.out.println(result[i]+"  ");
        }
    }
}
```

3. Longest Substring Without Repeating Characters
分析
难度 中
来源
https://leetcode.com/problems/longest-substring-without-repeating-characters/
题目

Given a string, find the length of the **longest substring** without repeating characters.

Example 1:

Input: "abcabcbb"
Output: 3
Explanation: The answer is "abc", with the length of 3.

Example 2:

Input: "bbbbb"
Output: 1
Explanation: The answer is "b", with the length of 1.

Example 3:

Input: "pwwkew"
Output: 3
Explanation: The answer is "wke", with the length of 3.
Note that the answer must be a **substring**, "pwke" is a *subsequence* and not a substring.
解答

```
package LeetCode;

import java.util.HashSet;

public class L3_LongestSubstringWithoutRepeatingCharacters {
    public int lengthOfLongestSubstring(String s) {
        HashSet hs=new HashSet();
        int maxlen=0,count=0;
        for(int i=0;i<s.length();i++){
            if(!hs.contains(s.charAt(i)))
            {
                hs.add(s.charAt(i));
                count++;
                if(count>maxlen)
```

```
                    maxlen=count;
                }
                else{
                    //找到重复字符在子串中的位置，在整个字符串中的位置
                    int pos=s.substring(i-count,i).indexOf(s.charAt(i));

System.out.println("i:\t"+i+";\n"+"count:\t"+count+";\n"+"pos:\t"+po
s+";\n");
                    String temp=s.substring(i-count,i-
count+pos);//substring 不包括 endindex
                    for (int j=0;j<temp.length();j++)
                        hs.remove(temp.charAt(j));//从哈希表中去掉重复字
符之前的字符
                    count=count-pos;
                }
            }
            return maxlen;
        }

        public static void main(String[] args){
            //String s="tmmzuxt";
            String s="abcabcbb";
            //String s="pwwt";
            L3_LongestSubstringWithoutRepeatingCharacters l3=new
L3_LongestSubstringWithoutRepeatingCharacters();
            System.out.println(l3.lengthOfLongestSubstring(s));
        }
}
```

5. Longest Palindromic Substring

分析

Manacher's Algorithm 马拉车算法

*这个马拉车算法 Manacher‘s Algorithm 是用来查找一个字符串的最长回文子串的线性方法，由一个叫 Manacher 的人在 1975 年发明的，这个方法的最大贡献是在于将时间复杂度提升到了线性，这是非常了不起的。对于回文串想必大家都不陌生，就是正读反读都一样的字符串，比如 "bob", "level", "noon" 等等，那么如何在一个字符串中找出最长回文子串呢，可以以每一个字符为中心，向两边寻找回文子串，在遍历完整个数组后，就可以找到最长的回文子串。但是这个方法的时间复杂度为 O(n*n)，并不是很高效，下面我们来看时间复杂度为 O(n)的马拉车算法。*

由于回文串的长度可奇可偶，比如"bob"是奇数形式的回文，"noon"就是偶数形式的回文，马拉车算法的第一步是预处理，做法是在每一个字符的左右都加上一个特殊字符，比如加上'#'，那么

bob --> #b#o#b#

noon --> #n#o#o#n#

这样做的好处是不论原字符串是奇数还是偶数个，处理之后得到的字符串的个数都是奇数个，这样就不用分情况讨论了，而可以一起搞定。接下来我们还需要和处理后的字符串 t 等长的数组 p，其中 p[i]表示以 t[i]字符为中心的回文子串的半径，若 p[i] = 1，则该回文子串就是 t[i]本身，那么我们来看一个简单的例子：

 # 1 # 2 # 2 # 1 # 2 # 2 #
1 2 1 2 5 2 1 6 1 2 3 2 1

为啥我们关心回文子串的半径呢？看上面那个例子，以中间的 '1' 为中心的回文子串 "#2#2#1#2#2#" 的半径是 6，而为添加井号的回文子串为 "22122"，长度是 5，为半径减 1。这是个普遍的规律么？我们再看看之前的那个 "#b#o#b#"，我们很容易看出来以中间的 'o' 为中心的回文串的半径是 4，而 "bob"的长度是 3，符合规律。再来看偶数个的情况"noon"，添加井号后的回文串为 "#n#o#o#n#"，以最中间的 '#' 为中心的回文串的半径是 5，而 "noon" 的长度是 4，完美符合规律。所以我们只要找到了最大的半径，就知道最长的回文子串的字符个数了。只知道长度无法确定子串，我们还需要知道子串的起始位置。

我们还是先来看中间的 '1' 在字符串 "#1#2#2#1#2#2#" 中的位置是 7，而半径是 6，貌似 7-6=1，刚好就是回文子串 "22122" 在原串 "122122" 中的起

始位置 1。那么我们再来验证下 "bob"，"o" 在 "#b#o#b#" 中的位置是 3，但是半径是 4，这一减成负的了，肯定不对。所以我们应该至少把中心位置向后移动一位，才能为 0 啊，那么我们就需要在前面增加一个字符，这个字符不能是井号，也不能是 s 中可能出现的字符，所以我们暂且就用美元号吧，毕竟是博主最爱的东西嘛。这样都不相同的话就不会改变 p 值了，那么末尾要不要对应的也添加呢，其实不用的，不用加的原因是字符串的结尾标识为'\0'，等于默认加过了。那此时 "o" 在 "$#b#o#b#" 中的位置是 4，半径是 4，一减就是 0 了，貌似没啥问题。我们再来验证一下那个数字串，中间的 '1' 在字符串 "$#1#2#2#1#2#2#" 中的位置是 8，而半径是 6，这一减就是 2 了，而我们需要的是 1，所以我们要除以 2。之前的 "bob" 因为相减已经是 0 了，除以 2 还是 0，没有问题。再来验证一下 "noon"，中间的 '#' 在字符串 "$#n#o#o#n#" 中的位置是 5，半径也是 5，相减并除以 2 还是 0，完美。可以任意试试其他的例子，都是符合这个规律的，最长子串的长度是半径减 1，起始位置是中间位置减去半径再除以 2。

那么下面我们就来看如何求 p 数组，需要新增两个辅助变量 mx 和 id，其中 id 为能延伸到最右端的位置的那个回文子串的中心点位置，mx 是回文串能延伸到的最右端的位置，这个算法的最核心的一行如下：

```
p[i] = mx > i ? min(p[2 * id - i], mx - i) : 1;
```

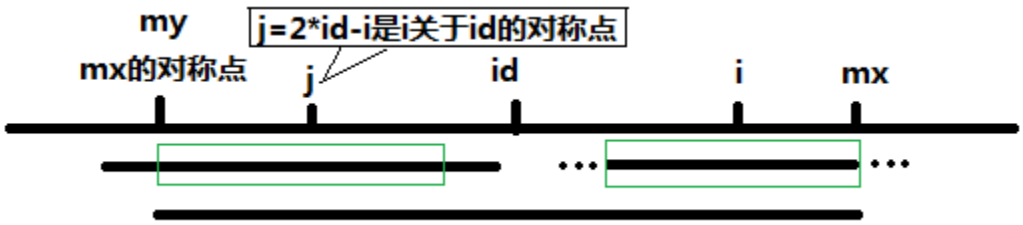
可以这么说，这行要是理解了，那么马拉车算法基本上就没啥问题了，那么这一行代码拆开来看就是

如果 mx > i，则 p[i] = min(p[2 * id - i] , mx - i)

否则，p[i] = 1

当 mx - i > P[j] 的时候，以 S[j]为中心的回文子串包含在以 S[id]为中心的回文子串中，由于 i 和 j 对称，以 S[i]为中心的回文子串必然包含在以 S[id]为中心的回文子串中，所以必有 P[i] = P[j]，见下图。

当 P[j] >= mx - i 的时候，以 S[j]为中心的回文子串不一定完全包含于以 S[id]为中心的回文子串中，但是基于对称性可知，下图中两个绿框所包围的部分是相同的，也就是说以 S[i]为中心的回文子串，其向右至少会扩张到 mx 的位置，也就是说 P[i] = mx - i。至于 mx 之后的部分是否对称，就只能老老实实去匹配了。



对于 mx <= i 的情况，无法对 P[i]做更多的假设，只能 P[i] = 1，然后再去匹配了。

难度 中
来源 https://leetcode.com/problems/longest-palindromic-substring/

题目

Given a string **s**, find the longest palindromic substring in **s**. You may assume that the maximum length of **s** is 1000.

Example 1:

Input: "babad"
Output: "bab"
Note: "aba" is also a valid answer.

Example 2:

Input: "cbbd"
Output: "bb"

解答

```
package LeetCode;

public class L5_LongestPalindromicSubstring {
   /* public String longestPalindrome(String s) {//Manacher 算法
        // Insert '#'
        if(s.length()<=1)
            return s;
        String t = "$#";
        for (int i = 0; i < s.length(); i++) {
```

```
            t += s.charAt(i);
            t += "#";
        }
        // Process t
        int[] p=new int[t.length()];//p[i]保存以 i 为中心的最长回文串的长度
        int mx = 0, id = 0, resLen = 0, resCenter = 0;
        //id 为能延伸到最右端的位置的那个回文子串的中心点位置，
        // mx 是回文串能延伸到的最右端的位置
        for (int i = 0; i < t.length(); i++) {
            p[i] = mx > i ? Math.min(p[2 * id - i], mx - i) : 1;
            while (i - p[i]>=0&&i + p[i]<t.length()&&t.charAt(i + p[i]) == t.charAt(i - p[i]))//i 的左右两侧对称，注意边界条件
                p[i]++;
            if (mx < i + p[i]) {
                mx = i + p[i];
                id = i;
            }
            if (resLen < p[i]) {//resLen 为最大回文串的长度.
                resLen = p[i];
                resCenter = i;//最大回文串的中心
            }
        }
        //return s.substring((resCenter - resLen) / 2, resLen - 1);//resLen-1 是原串中最长回文子串的长度，不是最右端下标
        return s.substring((resCenter - resLen) / 2, (resCenter +resLen) /2-1);
    }*/
   private int lo, maxLen;
    private void extendPalindrome(String s, int j, int k) {//以 j、k 为中心的回文串的最大长度
        // loop until meet invalid match
        while (j >= 0 && k < s.length() && s.charAt(j) == s.charAt(k)) {
            //从对称中心向两边扩展回文串
            j--;
            k++;
        }
        if (maxLen < k - j -1) {//正常理解下，maxLen 长度 k-j+1，但跳出循环时的 s.charAt(j) != s.charAt(k),k 多加了 1 次，j 多减了 1 次。maxLen=(k-j+1)-2
            lo = j + 1;
            maxLen = k - j - 1;
        }
```

```
    }
    public String longestPalindrome(String s) {
        int len = s.length();
        if (len < 2)
            return s;
        for (int i = 0; i < len-1; i++) {
            extendPalindrome(s, i, i);  //假设最大回文串长度为奇数
            extendPalindrome(s, i, i+1); //假设最大回文串长度为偶数
        }
        return s.substring(lo, lo + maxLen);//maxLen 为字符串实际长
度，作为 subString 第二个参数不包含在子串中
    }

    public static void main(String[] args){
        L5_LongestPalindromicSubstring l5=new
L5_LongestPalindromicSubstring();
        //String s="babad";
        String s="abcdbbfcba";
        System.out.println(l5.longestPalindrome(s));
    }
}
```

7. Reverse Integer

Given a 32-bit signed integer, reverse digits of an integer.

Example 1:

Input: 123
Output: 321

Example 2:

Input: -123
Output: -321

Example 3:

Input: 120
Output: 21

Note:

Assume we are dealing with an environment which could only store integers within the 32-bit signed integer range: $[-2^{31}, \ 2^{31} - 1]$. For the purpose of this problem, assume that your function returns 0 when the reversed integer overflows.

```
package LeetCode;
/*
7
 */
public class ReverseInteger {
    public int reverse(int x) {
        long ans=0;//java 中 int 为 32 位
        int maxint=0x7fffffff;
        int minint=0x80000000;
/*        int maxint=2147483647;
        int minint=-2147483648;*/
        while(x!=0){
            ans=ans*10+(x%10);
            x=x/10;
        }
        if(ans<minint||ans>maxint){
            ans=0;
        }
```

```
        return (int) ans;
    }

    public static void main(String[] args){
        ReverseInteger ri=new ReverseInteger();
        System.out.println(ri.reverse(1534236469 ));
    }
}
```

9. Palindrome Number（回文数）

分析

CSDN cnblogs

题目

Determine whether an integer is a palindrome. An integer is a palindrome when it reads the same backward as forward.

Example 1:

Input: 121
Output: true

Example 2:

Input: -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it becomes 121-. Therefore it is not a palindrome.

Example 3:

Input: 10
Output: false
Explanation: Reads 01 from right to left. Therefore it is not a palindrome.

Follow up:

Coud you solve it without converting the integer to a string?

解答

```
public boolean isPalindrome(int x) {
    int[] nums=new int[32];
    int i=0;
    if(x<0)//负数或   最后一位为0(后半句不严谨，单独0是对称的)
        return false;
    while(x/10>0){
```

```
            nums[i]=x%10;
            i++;
            x=x/10;
        }
        nums[i]=x%10;
        int len=0;
        for(i=31;i>=0;i--){
            if(nums[i]>0){
                len=i;//数字位数
                break;
            }
        }
        for(i=0;i<=len/2;i++){
            if(nums[i]!=nums[len-i])
                return false;
        }
        return true;
    }
```

13. Roman to Integer

分析

题目

Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, two is written as II in Roman numeral, just two one's added together. Twelve is written as, XII, which is simply X + II. The number twenty seven is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

- I can be placed before V (5) and X (10) to make 4 and 9.
- X can be placed before L (50) and C (100) to make 40 and 90.
- C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer. Input is guaranteed to be within the range from 1 to 3999.

Example 1:

Input: "III"
Output: 3

Example 2:

Input: "IV"

Output: 4

Example 3:

Input: "IX"
Output: 9

Example 4:

Input: "LVIII"
Output: 58
Explanation: L = 50, V= 5, III = 3.

Example 5:

Input: "MCMXCIV"
Output: 1994
Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.

解答

```
package LeetCode;

import java.util.HashMap;
import java.util.Map;
/*
I             1
V             5
X             10
L             50
C             100
D             500
M             1000
参考 https://www.cnblogs.com/TenosDoIt/p/3793503.html
从最后一个字符开始，如果当前字符对应的数字比上一个数字小，那么就把结果减去当前字符对应的数字，否则加上当前字符对应数字。
 */
public class L13_Roman2Integer {
    private static Character getKey(Map<Character,Integer>
map,Integer value)
    {
        Character key=null;
        for(Map.Entry<Character,Integer> entry:map.entrySet()){
            if(value.equals(entry.getValue())){
                key=entry.getKey();
            }
```

```
        }
        return key;
    }
    public int romanToInt(String s) {
        Map<Character,Integer> map=new HashMap<Character, Integer>();
        map.put('I',1);
        map.put('V',5);
        map.put('X',10);
        map.put('L',50);
        map.put('C',100);
        map.put('D',500);
        map.put('M',1000);
        int len=s.length();
        int result=map.get(s.charAt(len-1));
        for(int i=len-1;i>0;i--)
            if(map.get(s.charAt(i))>map.get(s.charAt(i-1)))
                result-=map.get(s.charAt(i-1));
            else
                result+=map.get(s.charAt(i-1));
        return result;
    }
}
```

14. Longest Common Prefix

分析

题目

Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Example 1:

Input: ["flower","flow","flight"]
Output: "fl"

Example 2:

Input: ["dog","racecar","car"]
Output: ""
Explanation: There is no common prefix among the input strings.

Note:

All given inputs are in lowercase letters a-z.

解答

```
package LeetCode;

public class L14_LongestCommonPrefix {
    public String longestCommonPrefix(String[] strs) {
        StringBuilder prefix=new StringBuilder();
        if(strs.length==0)
            return "";
        boolean flag=true;//当前循环是否有共同字母
        for(int i=0;i<strs[0].length();i++)
        {
            for(int j=1;j<strs.length;j++){//其他字符串当前位置字符是否与第一行字符串当前位置字符一致
```

```
                if(strs[j].length()<i+1){
                    flag=false;
                    //prefix.charAt(prefix.length()-1);//移除最后一个
字符
                    break;
                }else if(strs[j].charAt(i)!=strs[0].charAt(i)){
                    flag=false;
                    break;
                }else
                    continue;
            }
            if(!flag)
                break;
            prefix.append(strs[0].charAt(i));
        }
        return  prefix.toString();
    }
}
```

15. 三数之和

给定一个包含 n 个整数的数组 nums，判断 nums 中是否存在三个元素 a，b，c ，使得 $a + b + c = 0$ ？找出所有满足条件且不重复的三元组。

注意：答案中不可以包含重复的三元组。

例如，给定数组 nums = [-1, 0, 1, 2, -1, -4]，

满足要求的三元组集合为：

```
[
  [-1, 0, 1],
  [-1, -1, 2]
]
```

20. Valid Parentheses

分析

CSDN,cnblogs

题目

Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.

Note that an empty string is also considered valid.

Example 1:

```
Input: "()"
Output: true
```

Example 2:

```
Input: "()[]{}"
Output: true
```

Example 3:

```
Input: "(]"
Output: false
```

Example 4:

```
Input: "([)]"
Output: false
```

Example 5:

```
Input: "{[]}"
Output: true
```

解答

```
package LeetCode;

import java.util.HashMap;
import java.util.Map;
import java.util.Stack;

public class L20_ValidParentheses {
    public boolean isValid(String s) {
        int len=s.length();
        Stack<Character> stack=new Stack<Character>();
        if(len==0)
            return true;
        if(len%2!=0)
            return false;
        Map<Character,Character> map=new
HashMap<Character,Character>();
        map.put('(',')');
        map.put('[',']');
        map.put('{','}');
        map.put(')',' ');
        map.put(']',' ');
        map.put('}',' ');
        //stack.push(s.charAt(0));
        for(int i=0;i<len;i++)
        {

            if (stack.empty())//栈为空
                stack.push(s.charAt(i));
            else if(s.charAt(i)!=map.get(stack.peek()))//或输入与栈顶
不匹配
                stack.push(s.charAt(i));
            else
                stack.pop();
        }
        if(stack.empty())
            return true;
        else
            return false;
    }
    public static void main(String[] args){
        L20_ValidParentheses l20=new L20_ValidParentheses();
        String test="([)]";
        boolean result=l20.isValid(test);
        System.out.println(result);
```

```
        }
    }
```

21. Merge Two Sorted Lists

分析

难度：易
博客：
https://www.cnblogs.com/flowingfog/p/9787391.html
https://blog.csdn.net/flowingfog/article/details/83049541
来源：
https://leetcode.com/problems/merge-two-sorted-lists/description/

题目

Merge two sorted linked lists and return it as a new list. The new list should be made by splicing together the nodes of the first two lists.

Example:

Input: 1->2->4, 1->3->4
Output: 1->1->2->3->4->4

解答

```
package LeetCode;

public class L21_MergeTwoSortedLists {
    public ListNode mergeTwoLists(ListNode l1, ListNode l2) {
        ListNode temp=new ListNode(0);//指示当前节点
        ListNode ln=temp;//ln 为链表头部，返回的链表头部加入了一个节点。
        while(l1!=null &&l2!=null)
        {
            if(l1.val>l2.val){
                temp.next=l2;
                l2=l2.next;
            }
            else{
                temp.next=l1;
                l1=l1.next;
            }
            temp=temp.next;//注意链表上的移动，
        }
        if(l1!=null){
            temp.next=l1;
```

```
        }else
        {
            temp.next=l2;
        }
        return ln.next;
    }
}
```

26. Remove Duplicates from Sorted Array

Given a sorted array *nums*, remove the duplicates in-place such that each element appear only *once* and return the new length.

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

Example 1:

Given *nums* = **[1,1,2]**,

Your function should return length = **2**, with the first two elements of *nums* being **1** and **2** respectively.

It doesn't matter what you leave beyond the returned length.

Example 2:

Given *nums* = **[0,0,1,1,1,2,2,3,3,4]**,

Your function should return length = **5**, with the first five elements of *nums* being modified to **0**, **1**, **2**, **3**, and **4** respectively.

It doesn't matter what values are set beyond the returned length.

Clarification:

Confused why the returned value is an integer but your answer is an array?

Note that the input array is passed in by **reference**, which means modification to the input array will be known to the caller as well.

Internally you can think of this:

```
// nums is passed in by reference. (i.e., without making a copy)
int len = removeDuplicates(nums);

// any modification to nums in your function would be known by the caller.
// using the length returned by your function, it prints the first len elements.
for (int i = 0; i < len; i++) {
```

```
        print(nums[i]);
}
```

27. Remove Element

分析

难度 易
博客
https://www.cnblogs.com/flowingfog/p/9787748.html
https://blog.csdn.net/flowingfog/article/details/83050371
来源
https://leetcode.com/problems/remove-element/description/

题目

Given an array *nums* and a value *val*, remove all instances of that value in-place and return the new length.

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

The order of elements can be changed. It doesn't matter what you leave beyond the new length.

Example 1:

Given *nums* = **[3,2,2,3]**, *val* = **3**,

Your function should return length = **2**, with the first two elements of *nums* being **2**.

It doesn't matter what you leave beyond the returned length.

Example 2:

Given *nums* = **[0,1,2,2,3,0,4,2]**, *val* = **2**,

Your function should return length = **5**, with the first five elements of *nums* containing **0**, **1**, **3**, **0**, and **4**.

Note that the order of those five elements can be arbitrary.

It doesn't matter what values are set beyond the returned length.

Clarification:

Confused why the returned value is an integer but your answer is an array?

Note that the input array is passed in by **reference**, which means modification to the input array will be known to the caller as well.

Internally you can think of this:

```
// nums is passed in by reference. (i.e., without making a copy)
int len = removeElement(nums, val);

// any modification to nums in your function would be known by the caller.
// using the length returned by your function, it prints the first len elements.
for (int i = 0; i < len; i++) {
      print(nums[i]);
}
```

解答

```
package LeetCode;

public class L27_RemoveElement {
    public int removeElement(int[] nums, int val) {
        /*
        返回移除 val 后的数组长度
        同时需要对 nums 进行修改，使得 nums 前 len 个数不包含 val 值
        前 len 个数的顺序不做要求
        考虑类似快排的交换方式
         */
        int len=nums.length;
        int start=0,end=len-1;
        while(end>=start){
            //从后往前比较
            while(end>=start&&nums[end]==val)  //如果当前值等于 val，比较下一个，直到有不等于 val 的交换位置，然后又从前往后比较
            {
                end--;
                len--;
            }
            //从前往后比较
            while(end>=start&&nums[start]!=val)//如果没有等于 val 的，比较下一个，直到有比等于 val 的交换位置
                start++;
```

```
            if(end>=start){
                int temp = nums[start];
                nums[start] = nums[end];
                nums[end] = temp;
            }
            //一次循环交换一对数值的位置
        }
        // 比较结束，nums 即基本有序。左边的值都不等于 val，右边的值等于 val
        return len;
    }
    public static void main(String[] args){
        int[] nums={1};
        int val=1;
        L27_RemoveElement re=new L27_RemoveElement();
        int len = re.removeElement(nums, val);
        // any modification to nums in your function would be known by the caller.
        // using the length returned by your function, it prints the first len elements.
        for (int i = 0; i < len; i++) {
            System.out.println(nums[i]);
        }
    }
}
```

28. Implement strStr()

分析

KMP 算法

http://jakeboxer.com/blog/2009/12/13/the-knuth-morris-pratt-algorithm-in-my-own-words/

The Partial Match Table

The key to KMP, of course, is the partial match table. The main obstacle between me and understanding KMP was the fact that I didn't quite fully grasp what the values in the partial match table really meant. I will now try to explain them in the simplest words possible.

Here's the partial match table for the pattern "abababca":

```
char:  | a | b | a | b | a | b | c | a |
index: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
value: | 0 | 0 | 1 | 2 | 3 | 4 | 0 | 1 |
```

If I have an eight-character pattern (let's say "abababca" for the duration of this example), my partial match table will have eight cells. If I'm looking at the eighth and last cell in the table, I'm interested in the entire pattern ("abababca"). If I'm looking at the seventh cell in the table, I'm only interested in the first seven characters in the pattern ("ababab"); the eighth one ("a") is irrelevant, and can go fall off a building or something. If I'm looking at the sixth cell of the in the table… you get the idea. Notice that I haven't talked about what each cell *means* yet, but just what it's referring to.

Now, in order to talk about the meaning, we need to know about **proper prefixes** and **proper suffixes**.

Proper prefix: All the characters in a string, with one or more cut off the end. "S", "Sn", "Sna", and "Snap" are all the proper prefixes of "Snape".

Proper suffix: All the characters in a string, with one or more cut off the beginning. “agrid”, “grid”, “rid”, “id”, and “d” are all proper suffixes of “Hagrid”.

With this in mind, I can now give the one-sentence meaning of the values in the partial match table:

The length of the longest proper prefix in the (sub)pattern that matches a proper suffix in the same (sub)pattern.

Let’s examine what I mean by that. Say we’re looking in the third cell. As you’ll remember from above, this means we’re only interested in the first three characters (“aba”). In “aba”, there are two proper prefixes (“a” and “ab”) and two proper suffixes (“a” and “ba”). The proper prefix “ab” does not match either of the two proper suffixes. However, the proper prefix “a” matches the proper suffix “a”. Thus, **the length of the longest proper prefix that matches a proper suffix**, in this case, is 1.

Let’s try it for cell four. Here, we’re interested in the first four characters (“abab”). We have three proper prefixes (“a”, “ab”, and “aba”) and three proper suffixes (“b”, “ab”, and “bab”). This time, “ab” is in both, and is two characters long, so cell four gets value 2.

Just because it’s an interesting example, let’s also try it for cell five, which concerns “ababa”. We have four proper prefixes (“a”, “ab”, “aba”, and “abab”) and four proper suffixes (“a”, “ba”, “aba”, and “baba”). Now, we have two matches: “a” and “aba” are both proper prefixes and proper suffixes. Since “aba” is longer than “a”, it wins, and cell five gets value 3.

Let’s skip ahead to cell seven (the second-to-last cell), which is concerned with the pattern “abababc”. Even without enumerating all the proper prefixes and suffixes, it should be obvious that there aren’t going to be any matches; all the suffixes will end with the letter “c”, and none of the prefixes will. Since there are no matches, cell seven gets 0.

Finally, let’s look at cell eight, which is concerned with the entire pattern (“abababca”). Since they both start and end with “a”, we know the value will be at least 1. However, that’s where it ends; at lengths two and up, all the suffixes contain a c, while only the last prefix (“abababc”) does. This seven-character prefix

does not match the seven-character suffix (“bababca”), so cell eight gets 1.

How to use the Partial Match Table

We can use the values in the partial match table to skip ahead (rather than redoing unnecessary old comparisons) when we find partial matches. The formula works like this:

If a partial match of length ***partial_match_length*** *is found and table[partial_match_length] > 1, we may skip ahead partial_match_length - table[partial_match_length - 1] characters.*

Let’s say we’re matching the pattern “abababca” against the text “bacbababaabcbab”. Here’s our partial match table again for easy reference:

```
char:  | a | b | a | b | a | b | c | a |
index: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
value: | 0 | 0 | 1 | 2 | 3 | 4 | 0 | 1 |
```

The first time we get a partial match is here:

```
bacbababaabcbab
  |
 abababca
```

This is a partial_match_length of 1. The value at table[partial_match_length - 1] (or table[0]) is 0, so we don’t get to skip ahead any. The next partial match we get is here:

```
bacbababaabcbab
    |||||
    abababca
```

This is a partial_match_length of 5. The value at table[partial_match_length - 1] (or table[4]) is 3. That means we get to skip ahead partial_match_length - table[partial_match_length - 1] (or 5 - table[4] or 5 - 3 or 2) characters:

```
// x denotes a skip
```

```
2
3 bacbababaabcbab
4     xx|||
5       abababca
```

This is a partial_match_length of 3. The value at table[partial_match_length - 1] (or table[2]) is 1. That means we get to skip ahead partial_match_length - table[partial_match_length - 1] (or 3 - table[2] or 3 - 1 or 2) characters:

```
1
2 // x denotes a skip
3
4 bacbababaabcbab
5       xx|
          abababca
```

At this point, our pattern is longer than the remaining characters in the text, so we know there's no match.

难度 易

来源

https://leetcode.com/problems/implement-strstr/

题目

Implement strStr().

Return the index of the first occurrence of needle in haystack, or **-1** if needle is not part of haystack.

Example 1:

Input: haystack = "hello", needle = "ll"
Output: 2

Example 2:

Input: haystack = "aaaaa", needle = "bba"
Output: -1

Clarification:

What should we return when needle is an empty string? This is a great question to ask during an interview.

For the purpose of this problem, we will return 0 when needle is an empty string. This is consistent to C's strstr() and Java's indexOf().

解答

Runtime: 4 ms, faster than 81.41% of Java online submissions for Implement strStr().
Memory Usage: 23.1 MB, less than 85.63% of Java online submissions for Implement strStr().

```
package LeetCode;

public class L28_ImplementStr {
    public int strStr(String haystack, String needle) {
        int len1=haystack.length(),len2=needle.length();
        if (len1<len2)
            return -1;
        if(len2==0)
            return 0;
        int i,j;
        for (i = 0; i<=len1-len2; i++) {
            for (j = 0; j<len2; j++) {
                if (needle.charAt(j) != haystack.charAt(i + j))
                    break;
            }
            if (j == len2) //模式串匹配完了。for 循环的中间加上限制条件的话，这儿就没法用了
                return i;
        }
        return -1;
    }
    public static void main(String[] args){
        //String haystack = "hello", needle = "ll";
        String haystack = "a", needle = "";
        L28_ImplementStr l28=new L28_ImplementStr();
        System.out.println(l28.strStr(haystack,needle));
    }
}
```

35. Search Insert Position

分析

难度 易
博客
https://www.cnblogs.com/flowingfog/p/9794691.html
https://blog.csdn.net/flowingfog/article/details/83064674
来源
https://leetcode.com/problems/search-insert-position/description/

题目

Given a sorted array and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

You may assume no duplicates in the array.

Example 1:

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

Example 2:

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

Example 3:

Input: [1,3,5,6], 7
Output: 4

Example 4:

Input: [1,3,5,6], 0
Output: 0

解答

```
package LeetCode;

public class L35_SearchInsertPosition {
    public int searchInsert(int[] nums, int target) {
```

```
        int pos=0;
        int len=nums.length;
        if(len==0)
            return pos;
        else if(len==1){
            if(nums[0]>=target)
                pos=0;
            else
                pos=1;
        }else{
            for(int i=0;i<len;i++)
            {
                if(nums[i]<target)
                    pos++;
                else
                    break;
            }
        }
        return pos;
    }
}
```

38. Count and Say

分析

难度 易
博客
https://www.cnblogs.com/flowingfog/p/9800062.html
https://blog.csdn.net/flowingfog/article/details/83096014
来源
https://leetcode.com/problems/count-and-say/description/

题目

The count-and-say sequence is the sequence of integers with the first five terms as following:

1. 1
2. 11
3. 21
4. 1211
5. 111221

1 is read off as "one 1" or 11.

11 is read off as "two 1s" or 21.

21 is read off as "one 2, then one 1" or 1211.

Given an integer n where $1 \leqslant n \leqslant 30$, generate the n^{th} term of the count-and-say sequence.

Note: Each term of the sequence of integers will be represented as a string.

Example 1:

Input: 1
Output: "1"

Example 2:

Input: 4
Output: "1211"

解答

```
package LeetCode;

public class L38_CountAndSay {
    public String countAndSay(int n) {
        if(n<=0){
            return null;
        }
        String res="1";//第一行结果
        int len = 0;
        for(int i=1;i<n;i++){//第二行及以后
            int count=1;//数字出现次数
            if(i==1){//第二行
                res="11";
            }else {//不加 else 的计划，第二行有 res，还会接着构建，第二行的 res 就会成为原本第三行展示的内容
                len = res.length();
                StringBuilder sb = new StringBuilder();
                for (int j = 1; j < len; j++) {//统计每个字符出现次数
                    if (res.charAt(j - 1) == res.charAt(j)) {
                        count++;
                    } else {
                        sb.append(count);
                        sb.append(res.charAt(j - 1));
                        count = 1;
                    }
                }
                sb.append(count);
                sb.append(res.charAt(len - 1));
                res = sb.toString();
            }
        }
        return res;
    }
    public static void main(String[] args){
        L38_CountAndSay l38=new L38_CountAndSay();
        System.out.println(l38.countAndSay(4));
    }
}
```

49. Group Anagrams

分析

难度
来源

题目

Given an array of strings, group anagrams together.

Example:

```
Input: ["eat", "tea", "tan", "ate", "nat", "bat"],
Output:
[
  ["ate","eat","tea"],
  ["nat","tan"],
  ["bat"]
]
```

Note:

- All inputs will be in lowercase.
- The order of your output does not matter.

解答

Runtime: 15 ms, faster than 99.23% of Java online submissions for Group Anagrams.

```
package LeetCode;

import java.util.Arrays;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.List;

public class L49_GroupAnagrams {
    public List<List<String>> groupAnagrams(String[] strs) {
        List<List<String>> res=new LinkedList<List<String>>();
        HashMap<String,Integer> hm=new HashMap<String,Integer>();//前
边为字符串，后边为字符串对应 list 的序号
        int listsnum=0;//list 数目
        for(int i=0;i<strs.length;i++){
            char[] temp=strs[i].toCharArray();
```

```
            Arrays.sort(temp);
            String str=String.valueOf(temp);
            if(!hm.containsKey(str))//temp.toString不是temp里的字符
组成字符串
            {
                hm.put(str,listsnum);
                List<String> list=new LinkedList<String>();
                list.add(strs[i]);
                res.add(list);
                listsnum++;
            }else{
                res.get(hm.get(str)).add(strs[i]);//找到对应的String
list，往里添加string
            }
        }
        //System.out.println(hm);
        return res;
    }
    public static void main(String[] args){
        L49_GroupAnagrams l49=new L49_GroupAnagrams();
        String[] strs={"eat", "tea", "tan", "ate", "nat", "bat"};
        System.out.println(l49.groupAnagrams(strs));
    }
}
```

50. Pow(x, n)

分析

难度 中
来源
https://leetcode.com/problems/powx-n/
博客
https://www.cnblogs.com/flowingfog/p/9873610.html
https://blog.csdn.net/flowingfog/article/details/83514297

题目

Implement pow(*x*, *n*), which calculates *x* raised to the power *n* (x^n).

Example 1:

```
Input: 2.00000, 10
Output: 1024.00000
```

Example 2:

```
Input: 2.10000, 3
Output: 9.26100
```

Example 3:

```
Input: 2.00000, -2
Output: 0.25000
```

Explanation: $2^{-2} = \frac{1}{2^2} = \frac{1}{4} = 0.25$

Note:

- -100.0 < *x* < 100.0
- *n* is a 32-bit signed integer, within the range [-2^{31}, $2^{31} - 1$]

解答

Runtime: 12 ms, faster than 88.92% of Java online submissions for Pow(x, n).

```
package LeetCode;

public class L50_PowXN {
    public double Pow(double x, int n) {
```

```
        double result=1;
        double temp=x;
        String binN = Integer.toBinaryString(n);
        int len=binN.length();
        //System.out.println(binN);
        for(int i=0;i<len;i++){
            int pos=len-1-i;
            //从低位到高位开始计算
            if(binN.charAt(pos)=='1'){
                result*=temp;//指数位上的求和，相当于底数位上的乘积
            }
            temp*=temp;//当前位表示的大小
        }
        return result;
    }
    public double myPow(double x, int n) {
        double result=0;
        if(n==0)
            return 1;
        if(n<0)
            result=1/Pow(x,-n);
        else
            result=Pow(x,n);
        return result;
    }
    public static void main(String[] args){
        double x=2.0;
        int n=-2;
        L50_PowXN l50=new L50_PowXN();
        System.out.println(l50.myPow(x,n));
    }
}
```

53. Maximum Subarray

分析

难度 易

博客

https://blog.csdn.net/flowingfog/article/details/83112798

https://blog.csdn.net/flowingfog/article/details/83112798

来源

https://leetcode.com/problems/maximum-subarray/description/

题目

Given an integer array nums, find the contiguous subarray (containing at least one number) which has the largest sum and return its sum.

Example:

```
Input: [-2,1,-3,4,-1,2,1,-5,4],
Output: 6
Explanation: [4,-1,2,1] has the largest sum = 6.
```

Follow up:

If you have figured out the O(*n*) solution, try coding another solution using the divide and conquer approach, which is more subtle.

解答

```
package LeetCode;
/*
遍历 array，对于每一个数字，我们判断，（之前的 sum + 这个数字） 和 （这个数字） 比大小，如果（这个数字）自己就比 （之前的 sum + 这个数字） 大的话，那么说明不需要再继续加了，直接从这个数字，开始继续，因为它自己已经比之前的 sum 都大了。
```

```
        反过来，如果 （之前的sum + 这个数字）大于 （这个数字）就继续加下去。
 */
public class L53_MaximumSubarray {
    public int maxSubArray(int[] nums) {
        int len=nums.length;
        int max=-java.lang.Integer.MAX_VALUE;
        if(nums==null||len==0)
            return max=0;
        else if(len==1)
            max=nums[0];
        else{
            int curMax=0;//dp[i]表示到第 i 个字符的最大字符串和
            for(int i=0;i<len;i++){
                curMax=Math.max(nums[i],curMax+nums[i]);//注意 max 函数的第一个参数
                if(curMax>max){
                    max=curMax;
                }
            }
        }
        return max;
    }
}
```

58. Length of Last Word

分析

难度 易
博客
https://www.cnblogs.com/flowingfog/p/9808407.html
https://blog.csdn.net/flowingfog/article/details/83140705
来源
https://leetcode.com/problems/length-of-last-word/description/

题目

Given a string *s* consists of upper/lower-case alphabets and empty space characters ' ', return the length of last word in the string.

If the last word does not exist, return 0.

Note: A word is defined as a character sequence consists of non-space characters only.

Example:

Input: "Hello World"
Output: 5

解答

```
package LeetCode;

public class L58_LengthOfLastWord {
    public int lengthOfLastWord(String s) {
        String[] strs;
        strs=s.split(" ");
        if(strs.length==0)
            return 0;
        else
            return strs[strs.length-1].length();
    }
}
```

66. Plus One

分析

难度 易

博客

https://www.cnblogs.com/flowingfog/p/9808615.html

https://blog.csdn.net/flowingfog/article/details/83141163

来源

https://leetcode.com/problems/plus-one/description/

题目

Given a **non-empty** array of digits representing a non-negative integer, plus one to the integer.

The digits are stored such that the most significant digit is at the head of the list, and each element in the array contain a single digit.

You may assume the integer does not contain any leading zero, except the number 0 itself.

Example 1:

Input: [1,2,3]
Output: [1,2,4]
Explanation: The array represents the integer 123.

Example 2:

Input: [4,3,2,1]
Output: [4,3,2,2]
Explanation: The array represents the integer 4321.

解答

```
package LeetCode;
```

```java
public class L66_PlusOne {
    public int[] plusOne(int[] digits) {
        int len=digits.length;
        int[] result=new int[len+1];
        /*for(int i=0;i<result.length;i++){
            System.out.println(result[i]);
        }
        System.out.println(result);*/
        int carry=1;//加1
        for(int i=len-1;i>=0;i--){//数组最高位表示数字低位，从最高位
开始算
            digits[i]+=carry;
            if(digits[i]==10){
                digits[i]=0;
                carry=1;//向前进位
            }else
                carry=0;
        }
        if(carry==1){
            result[0]=carry;
            for(int i=0;i<len;i++){
                result[i+1]=digits[i];
            }
            return result;
        }else
            return digits;
    }
    public static  void main(String[] args){
        L66_PlusOne l66=new L66_PlusOne();
        int[] digits={9,8,7,6,5,4,3,2,1,0};
        int[] result=l66.plusOne(digits);
        for(int i=0;i<result.length;i++){
            System.out.print(result[i]);
        }
    }
}
```

67. Add Binary

分析

难度 易

博客

https://www.cnblogs.com/flowingfog/p/9815798.html

https://blog.csdn.net/flowingfog/article/details/83180810

来源

https://leetcode.com/problems/add-binary/description/

题目

Given two binary strings, return their sum (also a binary string).

The input strings are both **non-empty** and contains only characters 1 or 0.

Example 1:

```
Input: a = "11", b = "1"
Output: "100"
```

Example 2:

```
Input: a = "1010", b = "1011"
Output: "10101"
```

解答

```
package LeetCode;

public class L67_AddBinary {
    private static String getSum(String a,String b){//a 比 b 长
        int len1=a.length();
        int len2=b.length();
        int[] res=new int[len1+1];//比 a 多一位，以备进位
```

```
        StringBuilder str=new StringBuilder();
        //StringBuilder res=new StringBuilder();
        int flag=0;//表示进位
        //int len=Math.min(len1,len2);//求和部分长度
        for(int i=0;i<len2;i++){//求和部分长度
            int aNum=a.charAt(len1-1-i)-'0';//从数字低位，即数组高位
起逐位相加
            int bNum=b.charAt(len2-1-i)-'0';
            int remainder=(aNum+bNum+flag)%2;
            res[len1-i]=remainder;
            flag=(aNum+bNum+flag)/2;
        }
        for(int i=0;i<len1-len2;i++){//与 b 求和后多余部分
            int aNum=a.charAt(len1-len2-1-i)-'0';//从数字低位，即数组
高位起逐位相加
            int remainder=(aNum+flag)%2;
            res[len1-len2-i]=remainder;
            flag=(aNum+flag)/2;
        }
        if (flag==1){
            res[0]=1;
        }
        if (flag==1){
            str.append("1");
            for(int i=0;i<len1;i++){
                str.append(res[i+1]);
            }
        }else{
            for(int i=0;i<len1;i++){
                str.append(res[i+1]);
            }
        }
        return str.toString();
    }
    public String addBinary(String a, String b) {
        int len1=a.length();
        int len2=b.length();
        if(len1>len2)
            return getSum(a,b);
        else
            return getSum(b,a);
    }

    public static void main(String[] args){
```

```
        String a1="11";
        String b1="1";
        String a2="100";
        String b2="110010";
        L67_AddBinary l67=new L67_AddBinary();
        System.out.println(l67.addBinary(a2,b2));
    }
}
```

69. Sqrt(x)

分析

难度 易
博客
https://www.cnblogs.com/flowingfog/p/9817025.html
https://blog.csdn.net/flowingfog/article/details/83183977
来源
https://leetcode.com/problems/sqrtx/description/

题目

Implement int sqrt(int x).

Compute and return the square root of *x*, where *x* is guaranteed to be a non-negative integer.

Since the return type is an integer, the decimal digits are truncated and only the integer part of the result is returned.

Example 1:

Input: 4
Output: 2

Example 2:

Input: 8
Output: 2
Explanation: The square root of 8 is 2.82842..., and since
the decimal part is truncated, 2 is returned.

解答

方法 1

```
package LeetCode;
/*
蛮力，Runtime: 63 ms, faster than 6.02% of Java online submissions
for Sqrt(x).
 */
public class L69_SqrtX {
    public int mySqrt(int x) {
        int res=0;
        int curSquare=0;//记录上一轮平方数，
```

```
        int nextSquare=0;
        for(int i=0;i<=x/2;i++)
        {
            nextSquare=(i+1)*(i+1);
            if(nextSquare>x||nextSquare<curSquare)//如果平方溢出，一
定是大于 x 的
                break;
            else{
                res++;
                curSquare=nextSquare;
            }
        }
        return res;
    }

    public static void main(String[] args){
        L69_SqrtX l69=new L69_SqrtX();
        System.out.println(l69.mySqrt(2147395600));
    }
}
```

方法 2 牛顿法

```
public int mySqrt (int x) {
    if (x <= 1)
        return x;
    double assume = x / 2;
    while (Math.abs(Math.pow(assume, 2) - x) >=1) {
        assume = (assume + x/assume) / 2;//求当前值与除以 x 的结果的均
值，故能不断接近平方根
    }
    return (int) Math.floor(assume);
}
```

方法 3 //二分查找

```
public int mySqrt (int x) {
    if (x <= 1)
        return x;
    int left=1,right=Integer.MAX_VALUE;
    int res=0;
    while(left<right){
        res=left+(right-left)/2;//防止溢出
        if(res>x/res){
            right=res;
        }else{
            left=res;
        }
```

```
        if(right-left<=1)
            break;
    }
    return (left+right)/2;
}
```

70. Climbing Stairs

分析

难度 易
来源
https://leetcode.com/problems/climbing-stairs/
博客
https://www.cnblogs.com/flowingfog/p/9873735.html
https://blog.csdn.net/flowingfog/article/details/83514450

题目

You are climbing a stair case. It takes *n* steps to reach to the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Note: Given *n* will be a positive integer.

Example 1:

Input: 2
Output: 2
Explanation: There are two ways to climb to the top.
1. 1 step + 1 step
2. 2 steps

Example 2:

Input: 3
Output: 3
Explanation: There are three ways to climb to the top.
1. 1 step + 1 step + 1 step
2. 1 step + 2 steps
3. 2 steps + 1 step

解答

Runtime: 2 ms, faster than 100.00% of Java online submissions for Climbing Stairs.

```
package LeetCode;

public class L70_ClimbingStairs {
```

```
    //n 个台阶的楼梯的走法等于：
    //ways(n) = ways(n-1) + ways(n-2)
    public int climbStairs(int n) {
        int pre1=0,pre2=1,ways=0;//两步前的走法数、一步前的走法数
        if(n<=0){
            return n;
        }
        for(int i=1;i<=n;i++)
        {
            ways=pre1+pre2;
            pre1=pre2;
            pre2=ways;
        }
        return ways;
    }
}
```

75. Sort Colors

分析

难度 中
来源
https://leetcode.com/problems/sort-colors/

题目

Given an array with *n* objects colored red, white or blue, sort them [in-place](#) so that objects of the same color are adjacent, with the colors in the order red, white and blue.

Here, we will use the integers 0, 1, and 2 to represent the color red, white, and blue respectively.

Note: You are not suppose to use the library's sort function for this problem.

Example:

Input: [2,0,2,1,1,0]
Output: [0,0,1,1,2,2]

Follow up:

- A rather straight forward solution is a two-pass algorithm using counting sort.
  First, iterate the array counting number of 0's, 1's, and 2's, then overwrite array with total number of 0's, then 1's and followed by 2's.
- Could you come up with a one-pass algorithm using only constant space?

解答

```
package LeetCode;

public class L75_SortColors {
    public void swap(int[] nums,int i,int j){//i j相等就不交换
        if(i!=j){
            int temp=nums[i];
            nums[i]=nums[j];
            nums[j]=temp;
        }
    }
```

```
    public void printArray(int[] nums){
        for(int i=0;i<nums.length;i++)
            System.out.print(nums[i]+"\t");
    }
    public void sortColors(int[] nums) {
        int j=0,k=nums.length-1;//j为0的最大下标，k为2的最小下标
        for(int i=0;i<=k;i++){
/*              System.out.println(":"+i);
            printArray(nums);
            System.out.println();*/
            if(nums[i]==0){
                swap(nums,i,j);
                j++;
                //i--;
            }else if(nums[i]==2) {//交换后，重新判定当前位置的值.前边一定没有2（原本有的话，已经和后边交换过了），但后边却可能有0
                swap(nums, i, k);
                k--;
                //i--;
            }
        }
    }
    public static void main(String[] args){
        //int[] nums={2,0,2,1,1,0};
        int[] nums={2,0,1};
        //int[] nums={1,2,0};
        L75_SortColors l75=new L75_SortColors();
        l75.sortColors(nums);
        for(int i=0;i<nums.length;i++)
        {
            System.out.print(nums[i]+"\t");
        }
    }
}
```

83. Remove Duplicates from Sorted List

分析

难度 易
来源
https://leetcode.com/problems/remove-duplicates-from-sorted-list/
博客
https://www.cnblogs.com/flowingfog/p/9873786.html
https://blog.csdn.net/flowingfog/article/details/83514513

题目

Given a sorted linked list, delete all duplicates such that each element appear only *once*.

Example 1:

Input: 1->1->2
Output: 1->2

Example 2:

Input: 1->1->2->3->3
Output: 1->2->3

解答

```
package LeetCode;

/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode(int x) { val = x; }
 * }
 */
public class L83_RemoveDuplicatesFromSortedList {
    public ListNode deleteDuplicates(ListNode head) {
        if(head==null)
            return head;
        ListNode cur=head;
```

```
        //ListNode temp;
        while(cur.next!=null){
            if(cur.val!=cur.next.val)
                cur=cur.next;
            else{
                /*temp=cur.next.next;
                cur.next=temp;*/
                cur.next=cur.next.next;
            }
        }
        return head;
    }
}
```

88. Merge Sorted Array

分析

难度 易
来源
https://leetcode.com/problems/merge-sorted-array/
博客

题目

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 greater or 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]

解答

Runtime: 3 ms, faster than 99.99% of Java online submissions for Merge Sorted Array.

```
package LeetCode;

public class L88_MergeSortedArray {
    public void merge(int[] nums1, int m, int[] nums2, int n) {
        int len=m+n;
        int count=0;//从后往前按顺序插入，已确定位置数字数目
        int i=0,j=0;//i j分别为数组 nums1 nums2上的游标
        while(i<m&&j<n){
```

```
            if(nums1[m-1-i]>=nums2[n-1-j]){
                nums1[len-1-count]=nums1[m-1-i];
                i++;
                count++;
            }
            else{
                nums1[len-1-count]=nums2[n-1-j];
                j++;
                count++;
            }
        }
        if(i<m){
            for(;i<m;i++){
                nums1[len-1-count]=nums1[m-1-i];
                count++;
            }
        }else if(j<n){
            for(;j<n;j++){
                nums1[len-1-count]=nums2[n-1-j];
                count++;
            }
        }
    }
    public static void main(String[] args){
        int[] nums1=new int[6];//[1,2,3,0,0,0]
        int[] nums2=new int[3];//[2,5,6]
        nums1[0]=1;
        nums1[1]=2;
        nums1[2]=3;
        //nums2[0]=2;
        nums2[0]=4;
        nums2[1]=5;
        nums2[2]=6;
        L88_MergeSortedArray l88=new L88_MergeSortedArray();
        l88.merge(nums1,3,nums2,3);
        for(int i=0;i<nums1.length;i++){
            System.out.print(nums1[i]+"\t");
        }
    }
}
```

100. Same Tree

分析

难度　易
来源
https://leetcode.com/problems/same-tree/
博客
https://www.cnblogs.com/flowingfog/p/9878904.html
https://blog.csdn.net/flowingfog/article/details/83547390

题目

Given two binary trees, write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical and the nodes have the same value.

Example 1:

```
Input:     1         1
          / \       / \
         2   3     2   3

        [1,2,3],   [1,2,3]
```

Output: true

Example 2:

```
Input:     1         1
          /           \
         2             2

        [1,2],     [1,null,2]
```

Output: false

Example 3:

```
Input:     1         1
          / \       / \
         2   1     1   2
```

[1,2,1], [1,1,2]

Output: false

解答

参考 https://www.cnblogs.com/grandyang/p/4053384.html

```
package LeetCode;

/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode(int x) { val = x; }
 * }
 */
public class L100_SameTree {
    public boolean isSameTree(TreeNode p, TreeNode q) {
        if(p==null&&q==null)
            return true;
        if((p==null&&q!=null)||(p!=null&&q==null)||(p.val!=q.val))
            return false;
        return
isSameTree(p.left,q.left)&&isSameTree(p.right,q.right);
    }
}
```

101~200

101. Symmetric Tree

分析

难度 易

来源

https://leetcode.com/problems/symmetric-tree/

博客

https://www.cnblogs.com/flowingfog/p/9878993.html

https://blog.csdn.net/flowingfog/article/details/83547497

题目

Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

For example, this binary tree [1,2,2,3,4,4,3] is symmetric:

```
    1
   / \
  2   2
 / \ / \
3  4 4  3
```

But the following [1,2,2,null,3,null,3] is not:

```
    1
   / \
  2   2
   \   \
   3    3
```

Note:
Bonus points if you could solve it both recursively and iteratively.

解答

Runtime: 8 ms, faster than 98.44% of Java online submissions for Symmetric Tree.

```
package LeetCode;

public class L101_SymmetricTree {
    public boolean isAlike(TreeNode p, TreeNode q) {//把一棵树的对称转化为两棵树的相似
        if(p==null&&q==null)
            return true;
        if((p==null&&q!=null)||(p!=null&&q==null)||(p.val!=q.val))
            return false;
        return isAlike(p.left,q.right)&&isAlike(p.right,q.left);
    }
    public boolean isSymmetric(TreeNode root) {
        if(root==null)
            return true;
        return isAlike(root.left,root.right);
    }
}
```

102. Binary Tree Level Order Traversal

分析

难度 中

来源

https://leetcode.com/problems/binary-tree-level-order-traversal/

博客

https://www.cnblogs.com/flowingfog/p/9886238.html

https://blog.csdn.net/flowingfog/article/details/83591637

题目

Given a binary tree, return the *level order* traversal of its nodes' values. (ie, from left to right, level by level).

For example:
Given binary tree [3,9,20,null,null,15,7],

```
    3
   / \
  9  20
    /  \
   15   7
```

return its level order traversal as:

```
[
  [3],
  [9,20],
  [15,7]
]
```

解答

```
package LeetCode;

import java.util.*;
```

```
public class L102_BinaryTreeLevelOrderTraversal {
    public TreeNode makeBinaryTreeByArray(Integer[] array,int index){
        if(index<array.length&&array[index]!=null){
            int value=array[index];
            TreeNode t=new TreeNode(value);
            array[index]=0;
            t.left=makeBinaryTreeByArray(array,index*2+1);
            t.right=makeBinaryTreeByArray(array,index*2+2);
            return t;
        }else
            return null;
    }

    public List<List<Integer>> levelOrder(TreeNode root) {
        List<Integer> levelList=new ArrayList<Integer>();
        List<List<Integer>> result=new ArrayList<List<Integer>>();
        if(root==null){
            return result;
        }
        Queue<TreeNode> queue=new LinkedList<TreeNode>();
        queue.offer(root);
        int curCount=1;//记录当前层节点数
        int nextCount=0;//记录下一层节点数
        int outCount=0;//记录出队列节点数
        while(!queue.isEmpty()){
            TreeNode node=queue.poll();
            levelList.add(node.val);
            outCount++;
            if(node.left!=null){
                queue.offer(node.left);
                nextCount++;
            }
            if(node.right!=null){
                queue.offer(node.right);
                nextCount++;
            }
            if(outCount==curCount)//当前层全部出队列
            {
                if(!levelList.isEmpty())
                    result.add(levelList);
                levelList=new ArrayList<Integer>();
                curCount=nextCount;
                nextCount=0;
                outCount=0;
```

```
            }
        }
        return result;
    }

    public static void main(String[] args){
        L102_BinaryTreeLevelOrderTraversal l102=new
L102_BinaryTreeLevelOrderTraversal();
        Integer[] values={1,2,3,4,5,6,7};
        TreeNode root=l102.makeBinaryTreeByArray(values,0);
        List<List<Integer>> res=l102.levelOrder(root);
        System.out.println(res.toString());
    }
}
```

104. Maximum Depth of Binary Tree

分析

难度 易
来源
https://leetcode.com/problems/maximum-depth-of-binary-tree/
博客
https://www.cnblogs.com/flowingfog/p/9879052.html
https://blog.csdn.net/flowingfog/article/details/83547601

题目

Given a binary tree, find its maximum depth.

The maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

Note: A leaf is a node with no children.

Example:

Given binary tree [3,9,20,null,null,15,7],

```
    3
   / \
  9  20
    /  \
   15   7
```

return its depth = 3.

解答

Runtime: 0 ms, faster than 100.00% of Java online submissions for Maximum Depth of Binary Tree.

```
package LeetCode;

public class L104_MaxDepthOfBinaryTree {
    public int maxDepth(TreeNode root) {
        if(root==null){
            return 0;
        }
        int depth=1;
        depth+=Math.max(maxDepth(root.left),maxDepth(root.right));
```

```
            return depth;
        }
    }
```

107. Binary Tree Level Order Traversal II

分析

难度 易
来源
https://leetcode.com/problems/binary-tree-level-order-traversal-ii/
博客
https://www.cnblogs.com/flowingfog/p/9886149.html
https://blog.csdn.net/flowingfog/article/details/83591404

题目

Given a binary tree, return the *bottom-up level order* traversal of its nodes' values. (ie, from left to right, level by level from leaf to root).

For example:

Given binary tree [3,9,20,null,null,15,7],

```
    3
   / \
  9  20
    /  \
   15   7
```

return its bottom-up level order traversal as:

```
[
  [15,7],
  [9,20],
  [3]
]
```

解答

```
package LeetCode;

import java.util.*;
import java.util.logging.Level;

/*
```

```
 */
public class L107_BinaryTreeLevelOrderTraversalII {
    /**
     * 采用递归的方式创建一颗二叉树
     * 传入的是二叉树的数组表示法
     * 构造后是二叉树的二叉链表表示法
     */
    public TreeNode makeBinaryTreeByArray(Integer[] array,int index){
        if(index<array.length&&array[index]!=null){
            int value=array[index];
            TreeNode t=new TreeNode(value);
            array[index]=0;
            t.left=makeBinaryTreeByArray(array,index*2+1);
            t.right=makeBinaryTreeByArray(array,index*2+2);
            return t;
        }else
            return null;
    }

    public List<List<Integer>> levelOrderBottom(TreeNode root) {
        List<Integer> levelList=new ArrayList<Integer>();
        Stack<List<Integer>> levellistStack=new
Stack<List<Integer>>();
        List<List<Integer>> result=new ArrayList<List<Integer>>();
        if(root==null){
            return result;
        }
        Queue<TreeNode> queue=new LinkedList<TreeNode>();
        queue.offer(root);
        int curCount=1;//记录当前层节点数
        int nextCount=0;//记录下一层节点数
        int outCount=0;//记录出队列节点数
        while(!queue.isEmpty()){
            TreeNode node=queue.poll();
            levelList.add(node.val);
            outCount++;
            if(node.left!=null){
                queue.offer(node.left);
                //levelList.add(node.left.val);//视角从子节点加入转换
为当前节点输出，
                nextCount++;
            }
            if(node.right!=null){
                queue.offer(node.right);
```

```
                //levelList.add(node.right.val);
                nextCount++;
            }
            if(outCount==curCount)//当前层全部出队列
            {
                if(!levelList.isEmpty())
                    levellistStack.push(levelList);
                levelList=new ArrayList<Integer>();
                curCount=nextCount;
                nextCount=0;
                outCount=0;
            }
        }
        while(!levellistStack.isEmpty()){
            result.add(levellistStack.pop());
        }
        return result;
    }

       public static void main(String[] args){
        L107_BinaryTreeLevelOrderTraversalII l107=new
L107_BinaryTreeLevelOrderTraversalII();
        Integer[] values={1,2,3,4,5,6,7};
        TreeNode root=l107.makeBinaryTreeByArray(values,0);
        List<List<Integer>> res=l107.levelOrderBottom(root);
        System.out.println(res.toString());
    }
}
```

108. Convert Sorted Array to Binary Search Tree

分析

难度 易

来源

https://leetcode.com/problems/convert-sorted-array-to-binary-search-tree/

博客

https://www.cnblogs.com/flowingfog/p/9903233.html

题目

Given an array where elements are sorted in ascending order, convert it to a height balanced BST.

For this problem, a height-balanced binary tree is defined as a binary tree in which the depth of the two subtrees of *every* node never differ by more than 1.

Example:

Given the sorted array: [-10,-3,0,5,9],

One possible answer is: [0,-3,9,-10,null,5], which represents the following height balanced BST:

```
      0
     / \
   -3   9
   /   /
 -10  5
```

解答

```
package LeetCode;
```

```
import java.util.LinkedList;
import java.util.Queue;

public class L108_ConvertSortedArray2BinarySearchTree {
    public TreeNode makeBinaryTreeByArray(int[] nums,int low,int
high){
        if(low<=high){
            int mid=(high+low)/2;
            TreeNode root=new TreeNode(nums[mid]);
            //System.out.println(root.val);
            root.left=makeBinaryTreeByArray(nums,low,mid-1);
            root.right=makeBinaryTreeByArray(nums,mid+1,high);
            return root;
        }
        else
            return null;
    }
    public TreeNode sortedArrayToBST(int[] nums) {
        return makeBinaryTreeByArray(nums,0,nums.length-1);
    }
    public void levelOrderTraversal(TreeNode root){//广度优先搜索+分
层
        if(root==null){
            //System.out.println("empty tree");
            return;
        }
        Queue<TreeNode> queue=new LinkedList<TreeNode>();
        queue.add(root);
        int curCount=1;//记录当前层节点数
        int nextCount=0;//记录下一层节点数
        int outCount=0;//记录出队列节点数
        while(!queue.isEmpty()){
            TreeNode node=queue.remove();
            outCount++;
            System.out.print(node.val+"\t");
            if(node.left!=null){
                queue.add(node.left);
                nextCount++;
            }
            if(node.right!=null){
                queue.add(node.right);
                nextCount++;
            }
            if(outCount==curCount)//当前层全部出队列
```

```
            {
                System.out.println();
                curCount=nextCount;
                outCount=0;
            }
        }
        System.out.println("");
    }
    public static void main(String[] args){
        int[] nums={-10,-3,0,5,9};
        L108_ConvertSortedArray2BinarySearchTree l108=new
L108_ConvertSortedArray2BinarySearchTree();
        TreeNode root=l108.sortedArrayToBST(nums);
        l108.levelOrderTraversal(root);
    }
}
```

110. Balanced Binary Tree

分析

难度 易

来源

https://leetcode.com/problems/balanced-binary-tree/

博客

https://www.cnblogs.com/flowingfog/p/9904075.html

https://blog.csdn.net/flowingfog/article/details/83716197

题目

Given a binary tree, determine if it is height-balanced.

For this problem, a height-balanced binary tree is defined as:

a binary tree in which the depth of the two subtrees of *every* node never differ by more than 1.

Example 1:

Given the following tree [3,9,20,null,null,15,7]:

```
    3
   / \
  9  20
    /  \
   15   7
```

Return true.

Example 2:

Given the following tree [1,2,2,3,3,null,null,4,4]:

```
       1
      / \
     2   2
    / \
   3   3
  / \
 4   4
```

Return false.

解答

```
package LeetCode;

import java.util.*;

public class L110_BalancedBinaryTree {
    public TreeNode makeBinaryTreeByArray(Integer[] array,int index){
        if(index<array.length&&array[index]!=null){
            int value=array[index];
            TreeNode t=new TreeNode(value);
            array[index]=0;
            t.left=makeBinaryTreeByArray(array,index*2+1);
            t.right=makeBinaryTreeByArray(array,index*2+2);
            return t;
        }else
            return null;
    }
    public int maxDepth(TreeNode root) {//获取深度的函数单独拆出
        if(root==null)
            return 0;
        return Math.max(maxDepth(root.left),maxDepth(root.right))+1;
    }
    public boolean isBalanced(TreeNode root) {
        if(root==null)
            return true;
        return (Math.abs(maxDepth(root.left)-
maxDepth(root.right))<=1)&&isBalanced(root.left)&&isBalanced(root.rig
ht);
    }

    public void levelOrderTraversal(TreeNode root){//广度优先搜索+分
层.注意非全二叉树插入空节点
        if(root==null){
```

```
            return;
        }
        Queue<TreeNode> queue=new LinkedList<TreeNode>();
        queue.offer(root);
        int curCount=1;//记录当前层节点数
        int nextCount=0;//记录下一层节点数
        int outCount=0;//记录出队列节点数
        while(!queue.isEmpty()){
            TreeNode node=queue.poll();
            outCount++;
            if(node!=null){
                System.out.print(node.val);//空节点无值打印，直接输出
下方的 tab
                queue.offer(node.left);
                nextCount++;
                queue.offer(node.right);
                nextCount++;
            }
            System.out.print("\t");
            if(outCount==curCount)//当前层全部出队列
            {
                System.out.println();
                curCount=nextCount;
                nextCount=0;
                outCount=0;
            }
        }
    }
    public static void main(String[] args){
        L110_BalancedBinaryTree l110=new L110_BalancedBinaryTree();
        Integer[] nums={3,9,20,null,null,15,7};
        //Integer[] nums={1,2,2,3,3,null,null,4,4};
        TreeNode root=l110.makeBinaryTreeByArray(nums,0);
        //l110.levelOrderTraversal(root);
        System.out.println(l110.isBalanced(root));
    }
}
```

111. Minimum Depth of Binary Tree

分析

难度 易

来源

https://leetcode.com/problems/minimum-depth-of-binary-tree

博客

https://www.cnblogs.com/flowingfog/p/9904348.html

题目

Given a binary tree, find its minimum depth.

The minimum depth is the number of nodes along the shortest path from the root node down to the nearest leaf node.

Note: A leaf is a node with no children.

Example:

Given binary tree [3,9,20,null,null,15,7],

```
    3
   / \
  9  20
    /  \
   15   7
```

return its minimum depth = 2.

解答

```
package LeetCode;

import java.util.LinkedList;
```

```
import java.util.Queue;

public class L111_MinimumDepthOfBinaryTree {
    public TreeNode makeBinaryTreeByArray(Integer[] array,int index){
        if(index<array.length&&array[index]!=null){
            int value=array[index];
            TreeNode t=new TreeNode(value);
            array[index]=0;
            t.left=makeBinaryTreeByArray(array,index*2+1);
            t.right=makeBinaryTreeByArray(array,index*2+2);
            return t;
        }else
            return null;
    }
    public int minDepth(TreeNode root) {
        if(root==null)
            return 0;
        int left=minDepth(root.left);
        int right=minDepth(root.right);
        if(left==0||right==0)
            //一个子树高度为零，当前节点非叶子节点，高度为非空子树的
高度+1;
            // 两个子树高度都为0，当前节点为叶子节点，高度为1
            return Math.max(left,right)+1;
        else
            return Math.min(left,right)+1;
    }
    public void levelOrderTraversal(TreeNode root){//广度优先搜索+分
层.注意非全二叉树插入空节点
        if(root==null)
            return;
        Queue<TreeNode> queue=new LinkedList<TreeNode>();
        queue.offer(root);
        int curCount=1;//记录当前层节点数
        int nextCount=0;//记录下一层节点数
        int outCount=0;//记录出队列节点数
        while(!queue.isEmpty()){
            TreeNode node=queue.poll();
            outCount++;
            if(node!=null){
                System.out.print(node.val);//空节点无值打印，直接输出
下方的 tab
                queue.offer(node.left);
                nextCount++;
```

```
                queue.offer(node.right);
                nextCount++;
            }
            System.out.print("\t");
            if(outCount==curCount)//当前层全部出队列
            {
                System.out.println();
                curCount=nextCount;
                nextCount=0;
                outCount=0;
            }
        }
    }
    public static void main(String[] args){
        L111_MinimumDepthOfBinaryTree l111=new
L111_MinimumDepthOfBinaryTree();
        Integer[] nums={3,9,20,null,null,15,7};
        //Integer[] nums={1,2,2,3,3,null,null,4,4};
        TreeNode root=l111.makeBinaryTreeByArray(nums,0);
        l111.levelOrderTraversal(root);
        System.out.println(l111.minDepth(root));
    }
}
```

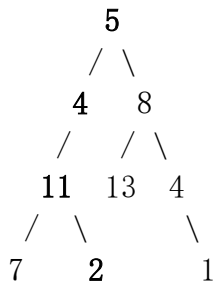
112. 路径总和

给定一个二叉树和一个目标和，判断该树中是否存在根节点到叶子节点的路径，这条路径上所有节点值相加等于目标和。

说明： 叶子节点是指没有子节点的节点。

示例:

给定如下二叉树，以及目标和 sum = 22，



返回 true, 因为存在目标和为 22 的根节点到叶子节点的路径 5->4->11->2。

```
package LeetCode;

import java.util.ArrayList;
import java.util.List;

public class PathSum {
    //ArrayList<List<Integer>> paths= new ArrayList<List<Integer>>();
    ArrayList<Integer> path=new ArrayList<Integer>();
    boolean hasPath=false;
    public  void DFS(TreeNode root,int sum,ArrayList<Integer> path){
        //System.out.println(sum);
        if(root.left==null && root.right==null)//root 为叶子节点，当前路径和为 sum;原来对是叶子节点但和不为 sum 的情况考虑不足
        {
            if(sum==0)
            {
                //path.add(root.val);
                //paths.add(new ArrayList<Integer>(path));
                hasPath=true;
                return;
                //path.remove(path.size()-1);//移除刚加上的节点
            }
        }else{
```

```
            path.add(root.val);//不加 else 的话，这句始终会执行
            if(root.left!=null)
                DFS(root.left,sum-root.left.val,path);
            //执行的过程中就把加入的子节点去掉了
            if(root.right!=null)
                DFS(root.right,sum-root.right.val,path);
            path.remove(path.size()-1);//移除加入的父节点
        }
    }
    public boolean hasPathSum(TreeNode root, int sum) {
        if(root==null){
            return false;
        }
        //path.add(root.val);
        DFS(root,sum-root.val,path);
        //return paths;
        return hasPath;
    }
}
```

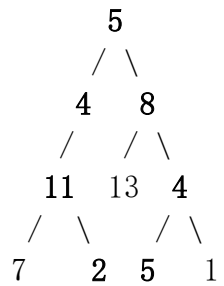
113. 路径总和 II

给定一个二叉树和一个目标和，找到所有从根节点到叶子节点路径总和等于给定目标和的路径。

说明: 叶子节点是指没有子节点的节点。

示例:

给定如下二叉树，以及目标和 sum = 22，



返回:

```
[
   [5,4,11,2],
```

```
   [5,8,4,5]
]
```

```
import java.util.*;

class L113 {
    //两个全局变量
    ArrayList<List<Integer>> paths= new ArrayList<List<Integer>>();
    ArrayList<Integer> path=new ArrayList<Integer>();
    public  void DFS(TreeNode root,int sum,ArrayList<Integer> path){
        //System.out.println(sum);//存在输入为负的情况
        if(root.left==null && root.right==null)//root 为叶子节点，当
前路径和为 sum;原来对是叶子节点但和不为 sum 的情况考虑不足
        {
            if(sum==0)
            {
                path.add(root.val);
                paths.add(new ArrayList<Integer>(path));
                path.remove(path.size()-1);//移除刚加上的节点
            }
        }else{
            path.add(root.val);//不加 else 的话，这句始终会执行
            if(root.left!=null)
                DFS(root.left,sum-root.left.val,path);
            //执行的过程中就把加入的子节点去掉了
            if(root.right!=null)
                DFS(root.right,sum-root.right.val,path);
            path.remove(path.size()-1);//移除加入的父节点
        }
    }
    public List<List<Integer>> pathSum(TreeNode root, int sum) {
        if(root==null){
            return paths;
        }
        //path.add(root.val);
        DFS(root,sum-root.val,path);
        return paths;
    }
}
```

118. Pascal's Triangle

分析

难度 易

来源

https://leetcode.com/problems/pascals-triangle/

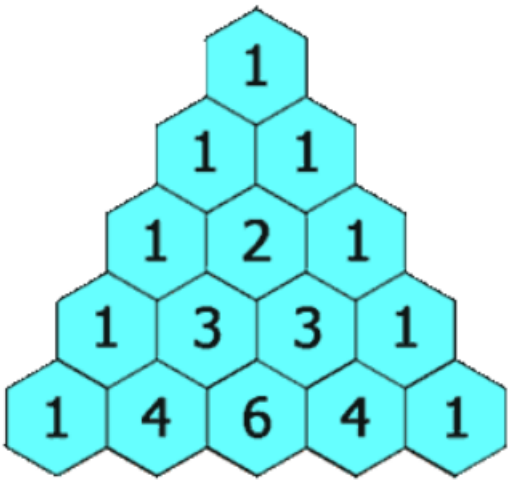
博客

https://www.cnblogs.com/flowingfog/p/9887397.html

https://blog.csdn.net/flowingfog/article/details/83615155

题目

Given a non-negative integer *numRows*, generate the first *numRows* of Pascal's triangle.



In Pascal's triangle, each number is the sum of the two numbers directly above it.

Example:

Input: 5
Output:
```
[
     [1],
```

```
    [1,1],
   [1,2,1],
  [1,3,3,1],
 [1,4,6,4,1]
]
```

解答

```
package LeetCode;

import java.util.ArrayList;
import java.util.List;

public class L118_PascalTriangle {
    public List<List<Integer>> generate(int numRows) {
        if(numRows<0)
            return null;
        List<List<Integer>> result=new ArrayList<List<Integer>>();
        List<Integer> list=new ArrayList<Integer>();//记录当前层数值
        for(int i=0;i<numRows;i++){
            list.add(1);
            for(int j=i-1;j>0;j--){//一行一行赋值，对每行从右向左赋值，避免修改下一个数字要用到两个加数
                list.set(j,list.get(j-1)+list.get(j));
            }
            result.add(new ArrayList<Integer>(list));
        }
        return result;
    }
    public static void main(String[] args){
        L118_PascalTriangle l118=new L118_PascalTriangle();
        System.out.println(l118.generate(5));
    }
}
```

119. Pascal's Triangle II

分析

难度 易

来源

https://leetcode.com/problems/pascals-triangle-ii/

博客

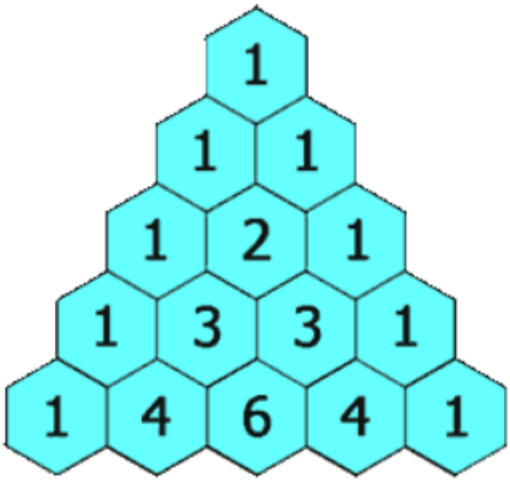
https://www.cnblogs.com/flowingfog/p/9895400.html

https://blog.csdn.net/flowingfog/article/details/83652943

题目

Given a non-negative index k where $k \leqslant 33$, return the k^{th} index row of the Pascal's triangle.

Note that the row index starts from 0.



In Pascal's triangle, each number is the sum of the two numbers directly above it.

Example:

Input: 3
Output: [1,3,3,1]

Follow up:

Could you optimize your algorithm to use only *O*(*k*) extra space?

解答

```
package LeetCode;

import java.util.ArrayList;
import java.util.List;

public class L119_PascalTriangleII {
    public List<Integer> getRow(int rowIndex) {
        if(rowIndex<0)
            return null;
        //从 0 开始的第 k 行，即（k+1）对应斐波那契数列的最后一行
        int listLen=rowIndex+1;
        List<Integer> list=new ArrayList<Integer>();//记录当前层数值
        for(int i=0;i<listLen;i++)//不知道为什么这句写在这里比下边二层循环的外层快，
            list.add(1);//整个都赋 1，初试值，各层边界值
        for(int i=0;i<listLen;i++){
            for(int j=i-1;j>0;j--){//一行一行赋值，对每行从右向左赋值，避免修改下一个数字要用到两个加数
                list.set(j,list.get(j-1)+list.get(j));
            }
        }
        return list;
    }
    public static void main(String[] args){
        long time1=System.currentTimeMillis();
        L119_PascalTriangleII l119=new L119_PascalTriangleII();
        System.out.println(l119.getRow(10));//从 0 开始的第 k 行，即（k+1）对应斐波那契数列的最后一行
        long time2=System.currentTimeMillis();
        System.out.println(time2-time1);
    }
}
```

121. Best Time to Buy and Sell Stock

分析

难度 易

来源

https://leetcode.com/problems/best-time-to-buy-and-sell-stock/

博客

https://www.cnblogs.com/flowingfog/p/9907929.html

https://blog.csdn.net/flowingfog/article/details/83745382

题目

Say you have an array for which the i^{th} element is the price of a given stock on day i.

If you were only permitted to complete at most one transaction (i.e., buy one and sell one share of the stock), design an algorithm to find the maximum profit.

Note that you cannot sell a stock before you buy one.

Example 1:

Input: [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.
Not 7-1 = 6, as selling price needs to be larger than buying price.

Example 2:

Input: [7,6,4,3,1]
Output: 0
Explanation: In this case, no transaction is done, i.e. max profit = 0.

解答

Runtime: 1 ms, faster than 99.91% of Java online submissions for Best Time to Buy and Sell Stock.

```
package LeetCode;

public class L121_BestTime2BuyAndSellStock {
      public int maxProfit(int[] prices) {
        int orginLen=prices.length;
        if(orginLen<2)
            return 0;
        int min=prices[0];
        int profit=0;
        for(int i=1;i<prices.length;i++){
            if(prices[i]<min)
                min=prices[i];
            if (prices[i] > prices[i - 1])//减少不必要的Math.max运算
                profit=Math.max(profit,prices[i]-min);
        }
        return profit;
    }
    public static void main(String[] args){
        L121_BestTime2BuyAndSellStock l121=new
L121_BestTime2BuyAndSellStock();
        int[] prices={7,1,5,3,6,4};
        long timer1=System.currentTimeMillis();
        System.out.println(l121.maxProfit(prices));
        long timer2=System.currentTimeMillis();
    }
}
```

122. Best Time to Buy and Sell Stock II

分析

难度 易

来源

https://leetcode.com/problems/best-time-to-buy-and-sell-stock-ii/

博客

https://www.cnblogs.com/flowingfog/p/9908070.html

https://blog.csdn.net/flowingfog/article/details/83745818

题目

Say you have an array for which the i^{th} element is the price of a given stock on day i.

Design an algorithm to find the maximum profit. You may complete as many transactions as you like (i.e., buy one and sell one share of the stock multiple times).

Note: You may not engage in multiple transactions at the same time (i.e., you must sell the stock before you buy again).

Example 1:

Input: [7,1,5,3,6,4]
Output: 7
Explanation: Buy on day 2 (price = 1) and sell on day 3 (price = 5), profit = 5-1 = 4.
Then buy on day 4 (price = 3) and sell on day 5 (price = 6), profit = 6-3 = 3.

Example 2:

Input: [1,2,3,4,5]
Output: 4
Explanation: Buy on day 1 (price = 1) and sell on day 5 (price = 5), profit = 5-1 = 4.

Note that you cannot buy on day 1, buy on day 2 and sell them later, as you are
engaging multiple transactions at the same time. You must sell before buying again.

Example 3:

Input: [7,6,4,3,1]
Output: 0
Explanation: In this case, no transaction is done, i.e. max profit = 0.

解答

Runtime: 1 ms, faster than 99.60% of Java online submissions for Best Time to Buy and Sell Stock II.

```
package LeetCode;

public class L122_BestTime2BuyAndSellStockII {
    public int maxProfit(int[] prices) {//允许多次操作，问题相当于求出差值序列中正值之和
        if(prices.length<2)
            return 0;
        int profit=0;
        for (int i = 1; i <prices.length; i++) {
            profit=((prices[i] - prices[i - 1])>0?(profit+(prices[i] - prices[i - 1])):(profit));
        }
        return profit;
    }
    public static void main(String[] args){
        L122_BestTime2BuyAndSellStockII l122=new L122_BestTime2BuyAndSellStockII();
        //int[] prices={7,1,5,3,6,4};
        int[] prices={1,2,3,4,5};
        System.out.println(l122.maxProfit(prices));
    }
}
```

125. Valid Palindrome

分析

难度 易

来源

https://leetcode.com/problems/valid-palindrome/

博客

https://www.cnblogs.com/flowingfog/p/9913647.html

https://blog.csdn.net/flowingfog/article/details/83782013

题目

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

Note: For the purpose of this problem, we define empty string as valid palindrome.

Example 1:

Input: "A man, a plan, a canal: Panama"
Output: true

Example 2:

Input: "race a car"
Output: false

解答

Runtime: 6 ms, faster than 72.53% of Java online submissions for Valid Palindrome.

```
package LeetCode;

public class L125_ValidPalindrome {
    public boolean isPalindrome(String s) {
        s=s.toLowerCase();
        if(s.length()<=1)
            return true;
```

```
        int head=0,tail=s.length()-1;
        while(head<=tail){
            if(!Character.isLetterOrDigit(s.charAt(head)))
            {
                head++;
                continue;
            }
            if(!Character.isLetterOrDigit(s.charAt(tail)))
            {
                tail--;
                continue;
            }
            if(s.charAt(head)==s.charAt(tail))
            {
                head++;
                tail--;
            }else
                return false;
        }
        return true;
    }
    public static void main(String[] args){
        L125_ValidPalindrome l125=new L125_ValidPalindrome();
        String s="A man, a plan, a canal: Panama";
        //String s="race a car";
        System.out.println(l125.isPalindrome(s));
    }
}
```

136. Single Number

分析

难度 易
来源
https://leetcode.com/problems/single-number
博客
https://www.cnblogs.com/flowingfog/p/9917973.html
https://blog.csdn.net/flowingfog/article/details/83794150

题目

Given a **non-empty** array of integers, every element appears *twice* except for one. Find that single one.

Note:

Your algorithm should have a linear runtime complexity. Could you implement it without using extra memory?

Example 1:

Input: [2,2,1]
Output: 1

Example 2:

Input: [4,1,2,1,2]
Output: 4

解答

```
package LeetCode;

public class L136_SingleNumber {
    public int singleNumber(int[] nums) {
        int result=0;
        for(int i=0;i<nums.length;i++){
            result^=nums[i];
        }
        return result;
    }
    public static void main(String[] args){
        L136_SingleNumber l136=new L136_SingleNumber();
```

```
        //int[] nums={2,2,1};
        int[] nums={4,1,2,1,2};
        System.out.println(1136.singleNumber(nums));
    }
}
```

137. Single Number II

分析

参考 https://cloud.tencent.com/developer/article/1131945

我们想要达到的效果其实是——

	a	b
初始状态：	*0*	*0*
第一次碰见某个数 x：	*0*	*x（把 x 记录在 b 中）*
第二次碰见某个数 x：	*x*	*0（把 x 记录在 a 中）*
第三次碰见某个数 x：	*0*	*0（把 a 和 b 都清空，可以处理其他数）*

还记得我们之前处理“所有元素都出现两次，只有一个特殊元素出现一次”的问题吗？其实我们那会想要达到的状态也是——

	a
初始状态：	*0*
第一次碰见某个数 x：	*x(把 x 记录在 a 中）*
第二次碰见某个数 x：	*0(把 a 清空）*

然后我们刚好就找到了异或运算可以处理这个问题。
那么这次我们同样利用异或运算，看能不能设计出一种变换的方法让 a 和 b 按照上述变换规则，进行转换。
b=0 时碰到 x，就变成 x；b=x 时再碰到 x，就变成 0，这个不就是异或吗？所以我们也许可以设计 b=b xor x。
但是当 b=0 时再再碰到 x，这时候 b 还是要为 0，但这时候不同的是 a=x，而前两种情况都是 a=0。所以我们可以设计成：b=(b xor x)&~a
同样道理，我们可以设计出：a=(a xor x)&~b
在这种变换规则下，a 和 b 都能按照我们设定的状态来发生转化。最后那个只出现一次的元素必定存储在 b 中。

难度 中
来源
https://leetcode.com/problems/single-number-ii/
博客
https://www.cnblogs.com/flowingfog/p/9919342.html
https://blog.csdn.net/flowingfog/article/details/83795664

题目

Given a **non-empty** array of integers, every element appears *three* times except for one, which appears exactly once. Find that single one.

Note:

Your algorithm should have a linear runtime complexity. Could you implement it without using extra memory?

Example 1:

Input: [2,2,3,2]
Output: 3

Example 2:

Input: [0,1,0,1,0,1,99]
Output: 99

解答

```
package LeetCode;

//位操作
public class L137_SingleNumberII {
    /*public int singleNumber(int[] nums) {
        int result=0;
        for(int i=0;i<32;i++)//逐位对result进行设置
        {
            int sum=0;
            for(int j=0;j<nums.length;j++){
                //如果这些数字的当前位为1，就用sum记录当前位为1的
数目
                //如果是1向左平移i位，求且的结果就不等于1了
                sum+=(nums[j]>>i) &1;
            }
            result|=(sum%3)<<i;
        }
        return result;
    }*/
     public int singleNumber(int[] nums) {
        int a =0;
        int b =0;
        for(int i=0;i<nums.length;i++){
            b = b ^ nums[i] & ~a;
            a = a ^ nums[i] & ~b;
        }
        return a|b;
    }
    public static void main(String[] args){
        L137_SingleNumberII l137=new L137_SingleNumberII();
        int[] nums={0,1,0,1,0,1,99};
```

```
        System.out.println(l137.singleNumber(nums));
    }
}
```

141. Linked List Cycle

分析

难度 易

来源

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

博客

https://www.cnblogs.com/flowingfog/p/9923621.html

https://blog.csdn.net/flowingfog/article/details/83827215

题目

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

Follow up:

Can you solve it without using extra space?

解答

Runtime: 0 ms, faster than 100.00% of Java online submissions for Linked List Cycle.

```
package LeetCode;

public class L141_LinkedListCycle {
    public boolean hasCycle(ListNode head) {
        //快慢指针。简单版本不需要返回循环开始节点
        if(head==null)
            return false;
        ListNode fast=head;
        ListNode slow=head;
        while(true){//fast.next 不空则 slow.next 不空
          if( fast.next==null||fast.next.next==null)
              return false;
          else
          {
              fast=fast.next.next;
```

```
                slow=slow.next;
                if(slow==fast)
                    return true;
            }
        }
    }
    public static void main(String[] args){
        int[] nums={-21,10,17,8,4,26,5,35,33,-7,-16,27,-12,6,29,-
12,5,9,20,14,14,2,13,-24,21,23,-21,5};
        ListNode head;
        ListNode pointer;
        pointer=head=new ListNode(nums[0]);
        for(int i=1;i<nums.length;i++)
        {
            pointer.next=new ListNode(nums[i]);
            pointer=pointer.next;
        }
        pointer=head;
        while(pointer!=null){
            System.out.print(pointer.val+"\t");
            pointer=pointer.next;
        }
        L141_LinkedListCycle l141=new L141_LinkedListCycle();
        System.out.println(l141.hasCycle(head));
    }
}
```

142. Linked List Cycle II

分析

难度 中
来源
https://leetcode.com/problems/linked-list-cycle-ii
博客
https://www.cnblogs.com/flowingfog/p/9926318.html
https://blog.csdn.net/flowingfog/article/details/83833548

环之前节点数 a；从环起始节点到快慢指针相遇位置，节点数 b。快慢指针 fast slow 相遇，慢指针走过 a+b，快指针走过 2(a+b)。快慢指针相遇时，创建第二个慢指针 slow2。同上，slow 再走过 a+b 后，slow 走过 2(a+b)，slow2 走过 a+b，二者在快慢指针相遇处相遇。slow slow2 速度相同，往前回推，二者从环开始处即相遇然后并行*

题目

Given a linked list, return the node where the cycle begins. If there is no cycle, return null.

Note: Do not modify the linked list.

Follow up:
Can you solve it without using extra space?

解答

```
package LeetCode;

public class L142_LinkedListCycleII {
    public ListNode buildList(int[] nums,boolean withCircle){
        int len=nums.length;
        ListNode head;
        ListNode pointer;
        pointer=head=new ListNode(nums[0]);
        for(int i=1;i<nums.length;i++)
        {
            pointer.next=new ListNode(nums[i]);
            pointer=pointer.next;
        }
        if(withCircle){
```

```
            int cicleStartNum=(int)(Math.random()*(len-1));
            ListNode cicleStart=head;
            for(int i=0;i<cicleStartNum;i++){
                cicleStart=cicleStart.next;
            }
            pointer.next=cicleStart;//环
        }
        return head;
    }
    public ListNode detectCycle(ListNode head) {
        ListNode fast=head;
        ListNode slow=head;
        while( fast!=null&&fast.next!=null){
            fast=fast.next.next;
            slow=slow.next;
            if(slow==fast){
                ListNode slow2=head;
                while(slow2!=slow){
                    slow=slow.next;
                    slow2=slow2.next;
                }
                return slow2;
            }
        }
        return null;
    }
    public static void main(String[] args){
        L142_LinkedListCycleII l142=new L142_LinkedListCycleII();
        int[] nums={-21,10,17,8,4,26,5,35,33,-7};
        ListNode ln;
        if(Math.random()>0.5)
            ln=l142.buildList(nums,true);
        else
            ln=l142.buildList(nums,false);
        ListNode result=l142.detectCycle(ln);
        if(result!=null)
            System.out.println(result.val);
        else
            System.out.println("没有环");
    }
}
```

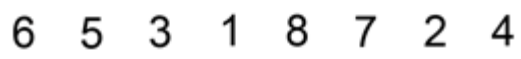
147. Insertion Sort List

分析

难度 中
来源
https://leetcode.com/problems/insertion-sort-list/

题目

Sort a linked list using insertion sort.



A graphical example of insertion sort. The partial sorted list (black) initially contains only the first element in the list.
With each iteration one element (red) is removed from the input data and inserted in-place into the sorted list

Algorithm of Insertion Sort:

1. Insertion sort iterates, consuming one input element each repetition, and growing a sorted output list.
2. At each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list, and inserts it there.
3. It repeats until no input elements remain.

Example 1:

Input: 4->2->1->3
Output: 1->2->3->4

Example 2:

Input: -1->5->3->4->0
Output: -1->0->3->4->5

解答

```
package LeetCode;

public class L147_InsertionSortList {
    public ListNode insertionSortList(ListNode head) {//插入排序，而非冒泡排序，对每一个节点，向前查找合适的位置插入。当前节点之前部分有序
        if(head==null)
            return head;
        ListNode prehead=new ListNode(Integer.MIN_VALUE);
        prehead.next=head;//应对节点插入 head 的情况。
        ListNode cur=head.next;//插入排序从第二个节点开始
        ListNode precur=head;//precur 表示 cur 前面的节点
        ListNode temp;
        while(cur!=null){
            if(precur.val>cur.val){//从前往后找到插入位置。前面的值大，后边的值小，即找到待插入节点
                //此时，只移动 cur
                temp=new ListNode(cur.val);//保存需要往前进行插入操作的节点的值。
                cur=cur.next;
                precur.next=cur;//在原位置删除需要进行插入操作的节点
                ListNode pointer=prehead;
                while(temp.val>pointer.next.val)//找到 temp.val
                {
                    pointer=pointer.next;
                }
                ListNode temp2=pointer.next;
                pointer.next=temp;
                temp.next=temp2;
            }else{//同时移动 cur 和 precur
                precur=precur.next;
                cur=cur.next;
            }
        }
        return prehead.next;
    }
    public static void main(String[] args){
        int[] nums={-1,5,3,4,0};
        ListNode head=ListNode.buildListNode(nums);
        //ListNode.printList(head);
```

```
        L147_InsertionSortList l147=new L147_InsertionSortList();
        l147.insertionSortList(head);
        ListNode.printList(head);
    }
}
```

148. Sort List

分析

归并排序介绍

*将两个的有序数列合并成一个有序数列，我们称之为"**归并**"。*
*归并排序(Merge Sort)就是利用归并思想对数列进行排序。根据具体的实现，归并排序包括"**从上往下**"和"**从下往上**"2 种方式。*

*1. **从下往上的归并排序**：将待排序的数列分成若干个长度为 1 的子数列，然后将这些数列两两合并；得到若干个长度为2 的有序数列，再将这些数列两两合并；得到若干个长度为4 的有序数列，再将它们两两合并；直接合并成一个数列为止。这样就得到了我们想要的排序结果。(参考下面的图片)*
*2. **从上往下的归并排序**：它与"从下往上"在排序上是反方向的。它基本包括3 步：*
① 分解 -- 将当前区间一分为二，即求分裂点 mid = (low + high)/2;
② 求解 -- 递归地对两个子区间 a[low...mid] 和 a[mid+1...high]进行归并排序。递归的终结条件是子区间长度为 1。
③ 合并 -- 将已排序的两个子区间 a[low...mid]和 a[mid+1...high]归并为一个有序的区间 a[low...high]。

时间复杂度 O(nlgn)

难度 中
来源
https://leetcode.com/problems/sort-list/

题目

Sort a linked list in $O(n \log n)$ time using constant space complexity.

Example 1:

Input: 4->2->1->3
Output: 1->2->3->4

Example 2:

Input: -1->5->3->4->0
Output: -1->0->3->4->5

解答

```
package LeetCode;

public class L148_SortList {
    public ListNode merge(ListNode l1, ListNode l2) {//合并两个有序列
表
        ListNode prehead=new ListNode(-1);
        ListNode pointer=prehead;
        while(l1!=null&&l2!=null){
            if(l1.val>l2.val){
                pointer.next=l2;
                pointer=pointer.next;
                l2=l2.next;
            }else{
                pointer.next=l1;
                pointer=pointer.next;
                l1=l1.next;
            }
        }
        if(l1!=null){
            pointer.next=l1;
        }else
            pointer.next=l2;
        return prehead.next;
    }
    public ListNode sortList(ListNode head) {
        if(head==null||head.next==null)
            return head;
      /*  ListNode prehead=new ListNode(Integer.MIN_VALUE);
        prehead.next=head;*/
        //ListNode preslow=head,slow=head,fast=head;//快慢指针
        //ListNode preslow=prehead,slow=head,fast=head;//快慢指针
        ListNode preslow=null,slow=head,fast=head;//快慢指针
        while(fast!=null&&fast.next!=null){
            //preslow=preslow.next;
            preslow=slow;
            /*if(slow!=head)//需要反复对 if 里的条件进行判断，效率差
                preslow=preslow.next;*/
            slow=slow.next;
            fast=fast.next.next;
        }
        preslow.next=null;//分为两个链表
        head=sortList(head);//对两个子链表分别合并
        slow=sortList(slow);
        return merge(head,slow);
```

```
        //return prehead.next;
    }
    public static void main(String[] args){
    /*    int[] nums1={1,3,5};
        int[] nums2={2,4,6};
        ListNode l1=ListNode.buildListNode(nums1);
        ListNode l2=ListNode.buildListNode(nums2);
        L148_SortList l148=new L148_SortList();
        ListNode.printList(l148.merge(l1,l2));*/
        int[] nums={4,2,1,3};
        ListNode head=ListNode.buildListNode(nums);
        L148_SortList l148=new L148_SortList();
        ListNode.printList(l148.sortList(head));
    }
}
```

155. Min Stack

分析

难度 易

来源

https://leetcode.com/problems/min-stack/

博客

https://www.cnblogs.com/flowingfog/p/9937550.html

https://blog.csdn.net/flowingfog/article/details/83904284

题目

Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

- push(x) -- Push element x onto stack.
- pop() -- Removes the element on top of the stack.
- top() -- Get the top element.
- getMin() -- Retrieve the minimum element in the stack.

Example:

```
MinStack minStack = new MinStack();
minStack.push(-2);
minStack.push(0);
minStack.push(-3);
minStack.getMin();   --> Returns -3.
minStack.pop();
minStack.top();      --> Returns 0.
minStack.getMin();   --> Returns -2.
```

解答

```
package LeetCode;

import java.util.LinkedList;
```

```
import java.util.Stack;

/**
 * Your MinStack object will be instantiated and called as such:
 * MinStack obj = new MinStack();
 * obj.push(x);
 * obj.pop();
 * int param_3 = obj.top();
 * int param_4 = obj.getMin();
 */
public class L155_MinStack {
   /*  private LinkedList<Integer> stack=null;
    int min=Integer.MAX_VALUE;
    public L155_MinStack() {
        stack=new LinkedList<Integer>();
    }

    public void push(int x) {
        if(x<=min){
            stack.addLast(min);//把之前的min压栈
            min=x;
        }
        stack.addLast(x);
    }

    public void pop() {
        if(min==stack.removeLast())
            min=stack.removeLast();
    }

    public int top() {
        return stack.getLast();
    }

    public int getMin() {
        return min;
    }*/
   //数组实现
     /*实时改变min
   Runtime: 76 ms, faster than 46.13% of Java online submissions for
Min Stack.
    */
  /* private Integer[] stack=null;
    private int maxSize=10000;
```

```
    private int top=-1;
    int min=Integer.MAX_VALUE;
    public L155_MinStack() {
        stack=new Integer[maxSize];
    }

    public void push(int x) {
        if(top==maxSize-1){//栈满
            maxSize*=2;
            Integer[] temp=stack;
            stack=new Integer[maxSize];
            for(int i=0;i<temp.length;i++){
                stack[i]=temp[i];
            }
        }
        stack[++top]=x;
        if(stack[top]<min)
            min=stack[top];
    }

    public void pop() {
        if(stack[top]==min){
            min=Integer.MAX_VALUE;
            for(int i=0;i<top;i++)
            {
                if(stack[i]<min)
                    min=stack[i];
            }
        }
        top--;
    }

    public int top() {
        return stack[top];
    }

    public int getMin() {
        return min;
    }*/
    /*注意最小值的保存，否则后边没法求 min
    使用 java 的 stack
     */
    private Stack<Integer> stack=null;
    int min=Integer.MAX_VALUE;
```

```
    public L155_MinStack() {
        stack=new Stack<Integer>();
    }

    public void push(int x) {//int 比 Integer 快好多啊
        if(x<=min){
            stack.push(min);//把之前的 min 压栈
            min=x;
        }
        stack.push(x);
    }

    public void pop() {
        if(min==stack.pop())//当前栈顶数字等于最小值，栈顶出栈，最小
数字为下方数字，栈顶入栈前的最小数字。不管等不等，pop 操作都执行过了
            min=stack.pop();//把最小值出栈，余下正常栈顶
    }

    public int top() {
        return stack.peek();
    }

    public int getMin() {
        return min;
    }
}
```

157. Read N Characters Given Read4

The API: int read4(char *buf) reads 4 characters at a time from a file.

The return value is the actual number of characters read. For example, it returns 3 if there is only 3 characters left in the file.

By using the read4 API, implement the function int read(char *buf, int n) that reads n characters from the file.

Note:
The read function will only be called once for each test case.

160. Intersection of Two Linked Lists

分析

难度 易

来源

https://leetcode.com/problems/intersection-of-two-linked-lists/

博客

https://www.cnblogs.com/flowingfog/p/9941070.html

https://blog.csdn.net/flowingfog/article/details/83934543

把第一个链表的结尾指向第二个链表的开头。
如果两个链表有交集，则新的链表有环
环的开始处即两个链表交集的起始节点

题目

Write a program to find the node at which the intersection of two singly linked lists begins.

For example, the following two linked lists:

```
A:          a1 → a2
                   ↘
                     c1 → c2 → c3
                   ↗
B:     b1 → b2 → b3
```

begin to intersect at node c1.

Notes:

- If the two linked lists have no intersection at all, return `null`.
- The linked lists must retain their original structure after the function returns.
- You may assume there are no cycles anywhere in the entire linked structure.

- Your code should preferably run in O(n) time and use only O(1) memory.

解答

```
package LeetCode;

public class L160_IntersectionOfTwoLinkedLists {
    public ListNode getIntersectionNode(ListNode headA, ListNode
headB) {
        if(headA==null||headB==null)
            return null;
        ListNode temp=headA;
        while(temp.next!=null)
        {
            temp=temp.next;
        }
        temp.next=headB;
        ListNode fast=headA;
        ListNode slow=headA;
        while( fast!=null&&fast.next!=null){
            fast=fast.next.next;
            slow=slow.next;
            if(slow==fast){
                ListNode slow2=headA;
                while(slow2!=slow){
                    slow=slow.next;
                    slow2=slow2.next;
                }
                temp.next=null;//恢复原链表结构
                return slow2;
            }
        }
        temp.next=null;
        return null;
    }
}
```

167. Two Sum II - Input array is sorted

分析

难度 易

来源

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

博客

https://www.cnblogs.com/flowingfog/p/9947171.html

https://blog.csdn.net/flowingfog/article/details/83993850

题目

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.
```

解答

```
package LeetCode;
import java.util.Arrays;

public class L167_TwoSumII_InputArraySorted {
```

```
    public int[] twoSum(int[] numbers, int target) {
        int front=0,back=numbers.length-1;
        while(numbers[front]+numbers[back]!=target){
            if(numbers[front]+numbers[back]>target)
                back--;
            else
                front++;
        }
        return new int[]{front+1,back+1};//下标计数从 1 开始
    }
    public static void main(String[] args){
        L167_TwoSumII_InputArraySorted l167=new
L167_TwoSumII_InputArraySorted();
        int[] numbers = {2,7,11,15};
        int target = 9;
        int[] result=l167.twoSum(numbers,target);
   /*     for (int res:result) {
            System.out.println(res+"\t");
        }*/
        System.out.println(Arrays.toString(result));
    }
}
```

168. Excel Sheet Column Title

分析

难度 易
来源
https://leetcode.com/problems/excel-sheet-column-title/
解答
https://www.cnblogs.com/flowingfog/p/9947861.html
https://blog.csdn.net/flowingfog/article/details/83995526

题目

Given a positive integer, return its corresponding column title as appear in an Excel sheet.

For example:

```
    1 -> A
    2 -> B
    3 -> C
    ...
    26 -> Z
    27 -> AA
    28 -> AB
    ...
```

Example 1:

```
Input: 1
Output: "A"
```

Example 2:

```
Input: 28
Output: "AB"
```

Example 3:

```
Input: 701
Output: "ZY"
```

解答

```java
package LeetCode;
/*
十进制转 26 进制
注意，z 表示 26，而非 0
 */
public class L168_ExcelSheetColumnTitle {
    public String convertToTitle(int n) {
        StringBuilder sb=new StringBuilder();
        while(n>0){
            if(n%26!=0){
                sb.append((char)(n%26+64));
                n/=26;
            }
            else{
                sb.append('Z');
                n=n/26-1;
            }
        }
        return sb.reverse().toString();
    }
    public static void main(String[] args){
        L168_ExcelSheetColumnTitle l168=new
L168_ExcelSheetColumnTitle();
        System.out.println(l168.convertToTitle(52));
    }
}
```

169. Majority Element

分析

难度 易

来源

https://leetcode.com/problems/majority-element/

博客

https://www.cnblogs.com/flowingfog/p/9956535.html

https://blog.csdn.net/flowingfog/article/details/84061949

题目

Given an array of size *n*, find the majority element. The majority element is the element that appears **more than ⌊** n/2 **⌋** times.

You may assume that the array is non-empty and the majority element always exist in the array.

Example 1:

```
Input: [3,2,3]
Output: 3
```

Example 2:

```
Input: [2,2,1,1,1,2,2]
Output: 2
```

解答

```
package LeetCode;

import java.util.Arrays;
import java.util.Stack;

public class L169_MajorityElement {
    /*
    //34ms
```

```
    public int majorityElement(int[] nums) {
        Stack<Integer> stack=new Stack<Integer>();
        for(int i=0;i<nums.length;i++)
        {
            if(stack.isEmpty()||stack.peek()==nums[i])
                stack.push(nums[i]);
            else
                stack.pop();
        }
        return stack.peek();
    }*/
    /*
    //5ms
    public int majorityElement(int[] nums) {
        int count=0,result=0;
        for(int i=0;i<nums.length;i++)
        {
            if(count==0)
                result=nums[i];
            if(result==nums[i])
                count++;
            else
                count--;
        }
        return result;
    }*/
    //3ms
    public int majorityElement(int[] nums) {
        Arrays.sort(nums);
        return nums[nums.length/2];
    }
    public static void main(String[] args){
        L169_MajorityElement l169=new L169_MajorityElement();
        //int[] nums={3,2,3};
        int[] nums={2,2,1,1,1,2,2};
        System.out.println(l169.majorityElement(nums));
    }
}
```

171. Excel Sheet Column Number

分析

难度 易

来源

https://leetcode.com/problems/excel-sheet-column-number

博客

https://www.cnblogs.com/flowingfog/p/9956657.html

题目

Given a column title as appear in an Excel sheet, return its corresponding column number.

For example:

```
    A -> 1
    B -> 2
    C -> 3
    ...
    Z -> 26
    AA -> 27
    AB -> 28
    ...
```

Example 1:

```
Input: "A"
Output: 1
```

Example 2:

```
Input: "AB"
Output: 28
```

Example 3:

Input: "ZY"
Output: 701

解答

Runtime: 1 ms, faster than 100.00% of Java online submissions for Excel Sheet Column Number.

```
package LeetCode;

public class L171_ExcelSheetColumnNumber {
    public int titleToNumber(String s) {
        int len=s.length();
        int colNum=0;
        int base=1;//当前位的 1 表示的大小
        for(int i=len-1;i>=0;i--){
            colNum+=(int)(s.charAt(i)-64)*base;
            base*=26;
        }
        return colNum;
    }
    public static void main(String[] args){
        L171_ExcelSheetColumnNumber l171=new
L171_ExcelSheetColumnNumber();
        String s="AB";
        System.out.println(l171.titleToNumber(s));
    }
}
```

172. Factorial Trailing Zeroes

分析

难度 易

来源

https://leetcode.com/problems/factorial-trailing-zeroes/

博客

https://www.cnblogs.com/flowingfog/p/9968226.html

https://blog.csdn.net/flowingfog/article/details/84136093

题目

Given an integer *n*, return the number of trailing zeroes in *n*!.

Example 1:

```
Input: 3
Output: 0
Explanation:  3! = 6, no trailing zero.
```

Example 2:

```
Input: 5
Output: 1
Explanation:  5! = 120, one trailing zero.
```

Note: Your solution should be in logarithmic time complexity.

解答

```
package LeetCode;

public class L172_FactorialTrailingZeroes {
    public int trailingZeroes(int n) {
        long factor=5;
        int res=0;
        while(n/factor>0){//n 不变，除数不断增大
```

```
            //n=n/factor;
            res+=n/factor;
            factor*=5;//5的幂,如25，含有多个5
        }
        return res;
    }
    public static void main(String[] args){
        L172_FactorialTrailingZeroes l172=new
L172_FactorialTrailingZeroes();
        System.out.println(l172.trailingZeroes(30));
    }
}
```

175. Combine Two Tables

分析

难度 易

来源

https://leetcode.com/problems/combine-two-tables/

博客

https://www.cnblogs.com/flowingfog/p/9969388.html

https://blog.csdn.net/flowingfog/article/details/84139583

SQL LEFT JOIN 关键字

LEFT JOIN 关键字会从左表（table_name1）那里返回所有的行，即使在右表（table_name2）中没有匹配的行。

LEFT JOIN 关键字语法

```
SELECT column_name(s)
FROM table_name1
LEFT JOIN table_name2
ON table_name1.column_name=table_name2.column_name
```

题目

Table: Person

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| PersonId    | int     |
| FirstName   | varchar |
| LastName    | varchar |
```

```
+-------------+---------+
```
PersonId is the primary key column for this table.

Table: Address

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| AddressId   | int     |
| PersonId    | int     |
| City        | varchar |
| State       | varchar |
+-------------+---------+
```
AddressId is the primary key column for this table.

Write a SQL query for a report that provides the following information for each person in the Person table, regardless if there is an address for each of those people:

FirstName, LastName, City, State

解答

Runtime: 184 ms, faster than 99.52% of MySQL online submissions for Combine Two Tables.

```
# Write your MySQL query statement below
select Person.FirstName, Person.LastName,Address.City,Address.State
from Person left join Address on Person.PersonId=Address.PersonId
order by Person.FirstName;
```

176. Second Highest Salary

分析

难度 易

来源

https://leetcode.com/problems/second-highest-salary/submissions/

博客

题目

Write a SQL query to get the second highest salary from the Employee table.

```
+----+--------+
| Id | Salary |
+----+--------+
| 1  | 100    |
| 2  | 200    |
| 3  | 300    |
+----+--------+
```

For example, given the above Employee table, the query should return 200 as the second highest salary. If there is no second highest salary, then the query should return null.

```
+---------------------+
| SecondHighestSalary |
+---------------------+
| 200                 |
+---------------------+
```

解答

```
# Write your MySQL query statement below
select max(Salary) as SecondHighestSalary from Employee where
Salary<(select max(Salary) from Employee);
```

181. Employees Earning More Than Their Managers

分析

难度 易

来源

https://leetcode.com/problems/employees-earning-more-than-their-managers/

博客

题目

The Employee table holds all employees including their managers. Every employee has an Id, and there is also a column for the manager Id.

```
+----+-------+--------+-----------+
| Id | Name  | Salary | ManagerId |
+----+-------+--------+-----------+
| 1  | Joe   | 70000  | 3         |
| 2  | Henry | 80000  | 4         |
| 3  | Sam   | 60000  | NULL      |
| 4  | Max   | 90000  | NULL      |
+----+-------+--------+-----------+
```

Given the Employee table, write a SQL query that finds out employees who earn more than their managers. For the above table, Joe is the only employee who earns more than his manager.

```
+----------+
| Employee |
+----------+
| Joe      |
+----------+
```

解答

```
# Write your MySQL query statement below
select E1.Name as Employee from Employee as E1,Employee as E2 where E1.ManagerId=E2.Id and E1.Salary > E2.Salary;
```

182. Duplicate Emails

分析

难度 易

来源

https://leetcode.com/problems/duplicate-emails/

博客

Group by,having

题目

Write a SQL query to find all duplicate emails in a table named Person.

```
+----+---------+
| Id | Email   |
+----+---------+
| 1  | a@b.com |
| 2  | c@d.com |
| 3  | a@b.com |
+----+---------+
```

For example, your query should return the following for the above table:

```
+---------+
| Email   |
+---------+
| a@b.com |
+---------+
```

Note: All emails are in lowercase.

解答

```
# Write your MySQL query statement below

select Email from Person group by(Email) having count(Id)>1
```

183. Customers Who Never Order

分析

难度 易

来源

https://leetcode.com/problems/customers-who-never-order/

解答

https://www.cnblogs.com/flowingfog/p/9977251.html

https://blog.csdn.net/flowingfog/article/details/84196680

题目

Suppose that a website contains two tables, the Customers table and the Orders table. Write a SQL query to find all customers who never order anything.

Table: Customers.

```
+----+-------+
| Id | Name  |
+----+-------+
| 1  | Joe   |
| 2  | Henry |
| 3  | Sam   |
| 4  | Max   |
+----+-------+
```

Table: Orders.

```
+----+------------+
| Id | CustomerId |
+----+------------+
| 1  | 3          |
| 2  | 1          |
+----+------------+
```

Using the above tables as example, return the following:

```
+-----------+
| Customers |
+-----------+
| Henry     |
| Max       |
+-----------+
```

解答

Runtime: 220 ms, faster than 94.79% of MySQL online submissions for Customers Who Never Order.

```
# Write your MySQL query statement below
select Name as Customers from Customers where Id not in (select CustomerId from Orders);
```

189. Rotate Array

分析

难度 易
来源
https://leetcode.com/problems/rotate-array/submissions/
解答
https://www.cnblogs.com/flowingfog/p/9980117.html
https://blog.csdn.net/flowingfog/article/details/84206068

题目

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

Example 1:

Input: [1,2,3,4,5,6,7] and 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: [-1,-100,3,99] and 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]

Note:

- Try to come up as many solutions as you can, there are at least 3 different ways to solve this problem.
- Could you do it in-place with O(1) extra space?

解答

Runtime: 1 ms, faster than 52.53% of Java online submissions for Rotate Array.

```java
package LeetCode;

public class L189_RotateArray {
    // Reverse the first n - k elements,
    // the last k elements, and then all the n elements.
    public void rotate(int[] nums, int k) {
        int len=nums.length;
        k=k%len;
        int temp=0;
        for(int i=0;i<(len-k)/2;i++){
            temp=nums[i];
            nums[i]=nums[len-k-1-i];
            nums[len-k-1-i]=temp;
        }
        for(int i=0;i<k/2;i++){
            temp=nums[len-k+i];//前半段下标 0~len-k-1;后半段 len-
k~len-1
            nums[len-k+i]=nums[len-1-i];
            nums[len-1-i]=temp;
        }
        for(int i=0;i<len/2;i++){
            temp=nums[i];
            nums[i]=nums[len-1-i];
            nums[len-1-i]=temp;
        }
    }

    public static void main(String[] args){
        int[] nums={1,2,3,4,5,6,7};
        L189_RotateArray l189=new L189_RotateArray();
        l189.rotate(nums,3);
        for (int num:nums) {
            System.out.print(num+"\t");
        }
    }
}
```

190. Reverse Bits

分析

难度 易

来源

https://leetcode.com/problems/reverse-bits/submissions/

博客

https://www.cnblogs.com/flowingfog/p/10016980.html

https://blog.csdn.net/flowingfog/article/details/84501850

题目

Reverse bits of a given 32 bits unsigned integer.

Example:

Input: 43261596
Output: 964176192
Explanation: 43261596 represented in binary as
00000010100101000001111010011100,
return 964176192 represented in binary as
00111001011110000010100101000000.

Follow up:

If this function is called many times, how would you optimize it?

解答

```
package LeetCode;

import java.util.Stack;

public class L190_ReverseBits {
    public int reverseBits(int n) {
        int result=0;
```

```
        for(int i=0;i<31;i++){
            result+=n&1;//取 n 最低位
            n>>>=1;
            result<<=1;//result=result<<1;左移，移动后原低位在前，高
位在后
        }
        result+=n&1;//最后一个在反转后为个位，不右移；放在循环中则执
行多次判断
        return result;
    }
    public static void main(String[] args){
        L190_ReverseBits l190=new L190_ReverseBits();
        int n=43261596;
        System.out.println(l190.reverseBits(n));
    }
}
```

191. Number of 1 Bits

分析

难度 易

来源

https://leetcode.com/problems/number-of-1-bits/

博客

https://www.cnblogs.com/flowingfog/p/10017055.html

https://blog.csdn.net/flowingfog/article/details/84502040

题目

Write a function that takes an unsigned integer and returns the number of '1' bits it has (also known as the Hamming weight).

Example 1:

```
Input: 11
Output: 3
Explanation: Integer 11 has binary representation
00000000000000000000000000001011
```

Example 2:

```
Input: 128
Output: 1
Explanation: Integer 128 has binary representation
00000000000000000000000010000000
```

解答

```
package LeetCode;

public class L191_NumberOf1Bits {
    public int hammingWeight(int n) {
        int result=0;
        for(int i=0;i<32;i++){
            if (1==(n&1))
```

```
                result++;
            n>>>=1;
        }
        return result;
    }
    public static void main(String[] args){
        L191_NumberOf1Bits l191=new L191_NumberOf1Bits();
        int n=11;
        System.out.println(l191.hammingWeight(n));
    }
}
```

192. Word Frequency

分析

写bash，不太会啊……
难度 中
来源
https://leetcode.com/problems/word-frequency/
博客
https://www.cnblogs.com/flowingfog/p/10021002.html
https://blog.csdn.net/flowingfog/article/details/84549167

题目

Write a bash script to calculate the frequency of each word in a text file words.txt.

For simplicity sake, you may assume:

- words.txt contains only lowercase characters and space ' ' characters.
- Each word must consist of lowercase characters only.
- Words are separated by one or more whitespace characters.

Example:

Assume that words.txt has the following content:

```
the day is sunny the the
the sunny is is
```

Your script should output the following, sorted by descending frequency:

```
the 4
is 3
sunny 2
day 1
```

Note:

- Don't worry about handling ties, it is guaranteed that each word's frequency count is unique.
- Could you write it in one-line using Unix pipes?

解答

https://leetcode.com/problems/word-frequency/discuss/55443/My-simple-solution-(one-line-with-pipe)

```
cat words.txt | tr -s ' ' '\n' | sort | uniq -c | sort -r | awk '{ print $2, $1 }'
```

tr -s: truncate the string with target string, but only remaining one instance (e.g. multiple whitespaces)

sort: To make the same string successive so that uniq could count the same string fully and correctly.

uniq -c: uniq is used to filter out the repeated lines which are successive, -c means counting

sort -r: -r means sorting in descending order

awk '{ print $2, $1 }': To format the output, see here.

193. Valid Phone Numbers

分析

难度 易

来源

https://leetcode.com/problems/valid-phone-numbers/

博客

https://www.cnblogs.com/flowingfog/p/10022723.html

https://blog.csdn.net/flowingfog/article/details/84556067

题目

Given a text file file.txt that contains list of phone numbers (one per line), write a one liner bash script to print all valid phone numbers.

You may assume that a valid phone number must appear in one of the following two formats: (xxx) xxx-xxxx or xxx-xxx-xxxx. (x means a digit)

You may also assume each line in the text file must not contain leading or trailing white spaces.

Example:

Assume that file.txt has the following content:

```
987-123-4567
123 456 7890
(123) 456-7890
```

Your script should output the following valid phone numbers:

```
987-123-4567
(123) 456-7890
```

解答

https://leetcode.com/problems/valid-phone-numbers/discuss/55478/Grep-e-solution-with-detailed-explanation-good-for-those-new-to-regex

```
grep -e '\(^[0-9]\{3\}-[0-9]\{3\}-[0-9]\{4\}$\)' -e '\(^([0-9]\{3\})[ ]\{1\}[0-9]\{3\}-\([0-9]\{4\}\)$\)'  file.txt
```

1. *In Bash, we use \ to escape next one trailing character;*
2. *^ is used to denote the beginning of a line*
3. *$ is used to denote the end of a line*
4. *{M} is used to denote to match exactly M times of the previous occurence/regex*
5. *(...) is used to group pattern/regex together*

Back to this problem: it requires us to match two patterns, for better readability, I used -e and separate the two patterns into two regexes, the first one matches this case: xxx-xxx-xxxx and the second one matches this case: (xxx) xxx-xxxx

加上-P（使用 Perl 的正则引擎）即可过滤出目标数据

```
grep -P '^(\d{3}-|\(\d{3}\) )\d{3}-\d{4}$' file.txt
```

注意上方的空格

```
mine    grep '^(\d{3}-|\(\d{3}\)[ ]{1})\d{3}-\d{4}$'  file.txt
```

这里使用[ ]{1}表示一个空格

195. Tenth Line

分析
难度 易
来源
https://leetcode.com/problems/tenth-line/
题目

Given a text file file.txt, print just the 10th line of the file.

Example:

Assume that file.txt has the following content:

```
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
Line 10
```

Your script should output the tenth line, which is:

```
Line 10
```

Note:

1. If the file contains less than 10 lines, what should you output?
2. There's at least three different solutions. Try to explore all possibilities.

解答

```
tail -n+10 file.txt|head -1
```

196. Delete Duplicate Emails

分析

题目

Write a SQL query to **delete** all duplicate email entries in a table named Person, keeping only unique emails based on its *smallest* **Id.**

```
+----+------------------+
| Id | Email            |
+----+------------------+
| 1  | john@example.com |
| 2  | bob@example.com  |
| 3  | john@example.com |
+----+------------------+
```

Id is the primary key column for this table.

For example, after running your query, the above Person table should have the following rows:

```
+----+------------------+
| Id | Email            |
+----+------------------+
| 1  | john@example.com |
| 2  | bob@example.com  |
+----+------------------+
```

Note:

Your output is the whole Person table after executing your sql. Use delete statement.

解答

Every derived table must have its own alias

```
# Write your MySQL query statement below

delete from Person where Id not in

(

    select * from (
```

```
    select MIN(Id) from Person as p1
        group by p1.Email) p
);
```

197. Rising Temperature

分析

难度 易

来源

https://leetcode.com/problems/rising-temperature/

inner join

TO_DAYS

题目

Given a Weather table, write a SQL query to find all dates' Ids with higher temperature compared to its previous (yesterday's) dates.

```
+---------+------------------+------------------+
| Id(INT) | RecordDate(DATE) | Temperature(INT) |
+---------+------------------+------------------+
|       1 |       2015-01-01 |               10 |
|       2 |       2015-01-02 |               25 |
|       3 |       2015-01-03 |               20 |
|       4 |       2015-01-04 |               30 |
+---------+------------------+------------------+
```

For example, return the following Ids for the above Weather table:

```
+----+
| Id |
+----+
|  2 |
|  4 |
+----+
```

解答

```
# Write your MySQL query statement below
select w1.Id from Weather w1 inner join Weather w2
```

```
on TO_DAYS(w1.Recorddate)=TO_DAYS(w2.RecordDate)+1 and
w1.Temperature>w2.Temperature;
```
或
```
# Write your MySQL query statement below
select w1.Id from Weather w1 ,Weather w2
where TO_DAYS(w1.Recorddate)=TO_DAYS(w2.RecordDate)+1 and
w1.Temperature>w2.Temperature;
```

198. House Robber

分析
难度
来源

题目

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security system connected and **it will automatically contact the police if two adjacent houses were broken into on the same night.**

Given a list of non-negative integers representing the amount of money of each house, determine the maximum amount of money you can rob tonight **without alerting the police.**

Example 1:

Input: [1,2,3,1]
Output: 4
Explanation: Rob house 1 (money = 1) and then rob house 3 (money = 3).
Total amount you can rob = 1 + 3 = 4.

Example 2:

Input: [2,7,9,3,1]
Output: 12
Explanation: Rob house 1 (money = 2), rob house 3 (money = 9) and rob house 5 (money = 1).
Total amount you can rob = 2 + 9 + 1 = 12.
解答

```
package LeetCode;

public class L198_HouseRobber {
    public int rob(int[] nums) {
        int len = nums.length;
        if(len==0)
            return 0;
        int prev1 = nums[0], prev2=0, temp;
        for (int i = 1; i < len; i++) {
```

```
            temp = prev1;
            prev1 = Math.max(prev2 + nums[i], prev1);
            prev2 = temp;
        }
        return prev1;
    }
    public static void main(String[] args){
        int[] input={1,2,3,1};
        L198_HouseRobber hr=new L198_HouseRobber();
        System.out.println(hr.rob(input));
    }
}
```

201~300

202. Happy Number

分析

难度 易
来源
https://leetcode.com/problems/happy-number/

题目

Write an algorithm to determine if a number is "happy".

A happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers.

Example:

Input: 19
Output: true
Explanation:
$1^2 + 9^2 = 82$
$8^2 + 2^2 = 68$
$6^2 + 8^2 = 100$
$1^2 + 0^2 + 0^2 = 1$

解答

```
package LeetCode;

import java.util.HashSet;

public class L202_HappyNumber {
    HashSet<Integer> set=new HashSet<Integer>();
    public boolean isHappy(int n) {
        if(n==1)
            return true;
        set.add(n);
```

```
        int res=0;
        while(n>0)
        {
            res+=(n%10)*(n%10);
            n=n/10;
        }
        if(set.contains(res))
            return false;
        return isHappy(res);
    }
    public static void main(String[] args){
        int input=19;
        L202_HappyNumber hn=new L202_HappyNumber();
        System.out.println(hn.isHappy(input));
    }
}
```

203. Remove Linked List Elements

分析
难度 易
来源
https://leetcode.com/problems/remove-linked-list-elements/
题目

Remove all elements from a linked list of integers that have value *val*.

Example:

Input: 1->2->6->3->4->5->6, *val* = 6
Output: 1->2->3->4->5
解答

```
package LeetCode;

public class L203_RemoveLinkedListElements {
    public ListNode buildList(int[] nums){
        ListNode head;
        ListNode pointer;
        pointer=head=new ListNode(nums[0]);
        for(int i=1;i<nums.length;i++)
        {
            pointer.next=new ListNode(nums[i]);
            pointer=pointer.next;
        }
        return head;
    }
    public ListNode removeElements(ListNode head, int val) {
        if(head==null)
            return null;
        ListNode dummyHead=new ListNode(-1);
        dummyHead.next=head;
        ListNode prev,cur;
        prev=dummyHead;
        cur=head;
        while(cur!=null)
        {
            if(cur.val==val)
                prev.next=cur.next;
            else
                prev=cur;
```

```
            cur=cur.next;
        }
        return dummyHead.next;
    }
    public static void main(String[] args){
        int[] nums={1,2,6,3,4,5,6};
        int n=6;
        L203_RemoveLinkedListElements 1203=new
L203_RemoveLinkedListElements();
        ListNode head=1203.buildList(nums);
        1203.removeElements(head,n);
        while(head!=null){
            System.out.print(head.val+"\t");
            head=head.next;
        }
    }
}
```

204. Count Primes

分析

题目

Count the number of prime numbers less than a non-negative number, ***n***.

Example:

Input: 10
Output: 4
Explanation: There are 4 prime numbers less than 10, they are 2, 3, 5, 7.

解答

```
package LeetCode;

public class L204_CountPrimes {
    public int countPrimes(int n){
        if(n<2)
            return 0;
        int count=0;
        boolean[] notPrime=new boolean[n];//notPrime 即是非素数。默认赋值为 false，如此定义可以不用初始化
        for(int i=2;i<Math.sqrt(n);i++){
            if(!notPrime[i])//i 是素数
            {
                for(int j=i;i*j<n;j++)//能保证覆盖所有合数吗？合数的重复赋值有多少？
                    notPrime[i*j]=true;
            }
        }
        for(int i=2;i<n;i++)
        {
            if(!notPrime[i]){//素数
                count++;
            }
        }
        return count;
```

```
    }
    public static void main(String[] args){
        L204_CountPrimes l204=new L204_CountPrimes();
        int input=10;
        System.out.println(l204.countPrimes(input));
    }
}
```

205. Isomorphic Strings（同形字符串）

分析

题目

Given two strings *s* and *t*, determine if they are isomorphic.

Two strings are isomorphic if the characters in *s* can be replaced to get *t*.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character but a character may map to itself.

Example 1:

Input: *s* = "egg", *t* = "add"
Output: true

Example 2:

Input: *s* = "foo", *t* = "bar"
Output: false

Example 3:

Input: *s* = "paper", *t* = "title"
Output: true

Note:

You may assume both *s* and *t* have the same length.

解答

```
package LeetCode;

import java.util.HashMap;
import java.util.Map;

public class L205_IsomorphicStrings {
```

```
    public boolean isIsomorphic(String s,String t){
        if(s==null||t==null||s.length()!=t.length())
            return false;
        Map map=new HashMap();//使用 map 做字母映射
        char[] sArr=s.toCharArray();
        char[] tArr=t.toCharArray();
        for(int i=0;i<s.length();i++){
            if(map.get(sArr[i])==null)//键未做映射，分两种情况，值已
存在，值不存在
            {
                if(map.containsValue(tArr[i]))//值映射到了其他键
                    return false;
                else
                    map.put(sArr[i],tArr[i]);
            }
            else
                if(tArr[i]!=(Character) map.get(sArr[i]))
                    return false;
        }
        return true;
    }

    public static void main(String[] args){
        L205_IsomorphicStrings l205=new L205_IsomorphicStrings();
        String s="ab";
        String t="aa";
        System.out.println(l205.isIsomorphic(s,t));
    }
}
```

206. Reverse Linked List

分析
题目

Reverse a singly linked list.

Example:

Input: 1->2->3->4->5->NULL
Output: 5->4->3->2->1->NULL

Follow up:

A linked list can be reversed either iteratively or recursively. Could you implement both?

解答

214. Shortest Palindrome

分析

https://leetcode.com/problems/shortest-palindrome/discuss/60098/My-7-lines-recursive-Java-solution

The idea is to use two anchors j and i to compare the String from beginning and end.
If j can reach the end, the String itself is Palindrome. Otherwise, we divide the String by j, and get mid = s.substring(0, j) and suffix.

We reverse suffix as beginning of result and recursively call shortestPalindrome to get result of mid then appedn suffix to get result.

```
    int j = 0;
    for (int i = s.length() - 1; i >= 0; i--) {
        if (s.charAt(i) == s.charAt(j)) { j += 1; }
    }
    if (j == s.length()) { return s; }
    String suffix = s.substring(j);
    return new StringBuffer(suffix).reverse().toString() +
shortestPalindrome(s.substring(0, j)) + suffix;
```

解法分析

The intuition: We know that it is possible to turn any string into a palindrome by simply concatenating it with its reverse: Ex. aabba + abbaa = aabbaabbaa. But this is not the shortest palindrome. If we could find a palindrome that is a prefix of aabba we could place it in the middle. By placing it in the middle we no longer would need to reverse it and we could only need to reverse the remainder of the string.
So the longest prefix of aabba that is a palindrome is aa. We can place aa in the middle and perform the reverse trick around it: abb + aa + bba = abbaabba which is the shortest palindrome.

```
    for (int i = s.length() - 1; i >= 0; i--) {
        if (s.charAt(i) == s.charAt(j)) { j += 1; }
    }
```

This loop is tricky. Since the goal is to reduce the number of elements that you have to reverse and concatenate we calculate j as the index which splits the array into two parts.

- *s.substring(j) is the suffix that (from the calculation of equal elements) has to be reversed in order to create a palindrome*
- *s.substring(0,j) is the prefix which may or may not be a palindrome, but is passed to the next recursive call in order to determine.*

Because i goes from len-1 to 0 and j goes from 0 to len-1 they will always cross and s.charAt(i) will equal s.charAt(j). This means j will always be greater than 0, causing the recursion to terminate and guaranteeing a worse case of O(N^2)

难度 难
来源 https://leetcode.com/problems/shortest-palindrome/

题目

Given a string ***s***, you are allowed to convert it to a palindrome by adding characters in front of it. Find and return the shortest palindrome you can find by performing this transformation.

Example 1:

Input: "aacecaaa"
Output: "aaacecaaa"

Example 2:

Input: "abcd"
Output: "dcbabcd"

解答

```
package LeetCode;

public class L214_ShortestPalindrome {
    /*public String shortestPalindrome(String s) {
        if(s.length()<=1)
            return s;
        int max=0;//从头开始的最长对称子串最右侧下标
        for(int i=0;i<s.length();i++)//i 作为对称串的右边界的下标
        {
```

```
            //System.out.println(i+":");
            int axis=i/2;//对称轴
            int j;
            if(i%2==0) {//到 i 的子串长度为奇数。对称中心为 i/2
                for(j=1;j<=axis;j++)
                {
                    //System.out.println("\t"+j);
                    if(s.charAt(axis-j)==s.charAt(axis+j))//判断对称
轴两侧值是否相等
                        continue;
                    else
                        break;
                }
                if(j>axis)
                    max=i;
            }
            else{//到 i 的子串长度为偶数。对称中心为 i/2, i/2+1
                for(j=0;j<=axis;j++)
                {
                    //System.out.println("\t"+j);
                    if(s.charAt(axis-j)==s.charAt(axis+1+j))
                        continue;
                    else
                        break;
                }
                if(j>axis)
                    max=i;
            }

        }
        //System.out.println(max);
        StringBuilder sb=new StringBuilder();
        return
sb.append(s).reverse().append(s.substring(max+1)).toString();
    }
    */

    public String shortestPalindrome(String s) {
        int j=0;
        for(int i=s.length()-1;i>=0;i--){
            if(s.charAt(i)==s.charAt(j))
                j++;
        }
        if(j==s.length())//整个字符串对称
```

```
            return s;
        return new
StringBuilder().append(s.substring(j)).reverse().toString()+shortestP
alindrome(s.substring(0,j))+s.substring(j);
    }
    public static void main(String[] args){
        L214_ShortestPalindrome l214=new L214_ShortestPalindrome();
        //String s="aacecaaa";
        //String s="abcd";
        String s="abbacd";
        //Input
        //"abbacd"
        //Output
        //"dcabbabbacd"
        //Expected
        //"dcabbacd"
        String result=l214.shortestPalindrome(s);
        System.out.println(result);
    }
}
```

217. Contains Duplicate

分析

题目

Given an array of integers, find if the array contains any duplicates.

Your function should return true if any value appears at least twice in the array, and it should return false if every element is distinct.

Example 1:

Input: [1,2,3,1]
Output: true

Example 2:

Input: [1,2,3,4]
Output: false

Example 3:

Input: [1,1,1,3,3,4,3,2,4,2]
Output: true
解答

```
package LeetCode;

import java.util.Arrays;
import java.util.HashMap;

public class L217_ContainsDuplicate {
    public boolean containsDuplicate(int[] nums) {
        Arrays.sort(nums);
        if(nums.length<=1)
            return false;
        for(int i=0;i<nums.length-1;i++)
        {
            if(nums[i]==nums[i+1])
                return true;
        }
        return false;
```

```
    }
    public static void main(String[] args){

    }
}
```

219. Contains Duplicate II

分析

难度 易
来源
https://leetcode.com/problems/contains-duplicate-ii/

题目

Given an array of integers and an integer *k*, find out whether there are two distinct indices *i* and *j* in the array such that **nums[i] = nums[j]** and the **absolute** difference between *i* and *j* is at most *k*.

Example 1:

```
Input: nums = [1,2,3,1], k = 3
Output: true
```

Example 2:

```
Input: nums = [1,0,1,1], k = 1
Output: true
```

Example 3:

```
Input: nums = [1,2,3,1,2,3], k = 2
Output: false
```

解答

```
package LeetCode;

import java.util.HashSet;

public class L219_ContainsDuplicateII {
    public boolean containsNearbyDuplicate(int[] nums, int k) {
        HashSet hm=new HashSet();
        for(int i=0;i<nums.length&&i<k;i++){//建立一个大小为k的集合
            if(hm.contains(nums[i]))
                return true;
            else
                hm.add(nums[i]);
        }
        for(int i=k;i<nums.length;i++){//集合作为滑动窗口，比较是否有距离小于k的相同的两个值
```

```
            if(hm.contains(nums[i]))
                return true;
            else
            {
                hm.add(nums[i]);
                hm.remove(nums[i-k]);
            }
        }
        return false;
    }

    public static void main(String[] args){
        int[] nums={1,2,3,1,2,3};
        L219_ContainsDuplicateII l219=new L219_ContainsDuplicateII();
        System.out.println(l219.containsNearbyDuplicate(nums,2));
    }
}
```

220. Contains Duplicate III

分析

难度 中
来源

题目

Given an array of integers, find out whether there are two distinct indices *i* and *j* in the array such that the **absolute** difference between **nums[i]** and **nums[j]** is at most *t* and the **absolute** difference between *i* and *j* is at most *k*.

Example 1:

Input: nums = [1,2,3,1], k = 3, t = 0
Output: true

Example 2:

Input: nums = [1,0,1,1], k = 1, t = 2
Output: true

Example 3:

Input: nums = [1,5,9,1,5,9], k = 2, t = 3
Output: false

解答

```
package LeetCode;
import java.util.TreeSet;

public class L220_ContainsDuplicateIII {
    public boolean containsNearbyAlmostDuplicate(int[] nums, int
k,int t) {
        TreeSet<Long> ts=new TreeSet<Long>();
        for(int i=0;i<nums.length;i++){//建立一个大小为k的集合,把
nums[i]和nums[i]的相近值都添加进去
            Long num=(long)nums[i];
            Long floor=ts.floor(num+t);//小于nums[i]+t的最大整数
            Long ceil=ts.ceiling(num-t);//大于nums[i]-t的最小整数
            if((floor!=null&&floor>=num)||(ceil!=null&&ceil<=num))
                return true;
```

```
                else
                    ts.add(num);
                if(i>=k)
                    ts.remove((long)nums[i-k]);
            }
            return false;
        }

        public static void main(String[] args){
            //int[] nums={1,2,3,1};
            //int k=3,t=0;
            //int[] nums={1,0,1,1};
            //int k=1,t=2;
            int[] nums={1,5,9,1,5,9};
            int k=2,t=3;
            L220_ContainsDuplicateIII l220=new
L220_ContainsDuplicateIII();

System.out.println(l220.containsNearbyAlmostDuplicate(nums,k,t));
        }
}
```

225. Implement Stack using Queues

分析
难度 易
来源
https://leetcode.com/problems/implement-stack-using-queues/
题目

Implement the following operations of a stack using queues.

- push(x) -- Push element x onto stack.
- pop() -- Removes the element on top of the stack.
- top() -- Get the top element.
- empty() -- Return whether the stack is empty.

Example:

```
MyStack stack = new MyStack();

stack.push(1);
stack.push(2);
stack.top();   // returns 2
stack.pop();   // returns 2
stack.empty(); // returns false
```

Notes:

- You must use *only* standard operations of a queue -- which means only `push to back`, `peek/pop from front`, `size`, and `is empty` operations are valid.
- Depending on your language, queue may not be supported natively. You may simulate a queue by using a list or deque (double-ended queue), as long as you use only standard operations of a queue.
- You may assume that all operations are valid (for example, no pop or top operations will be called on an empty stack).

解答

```
package LeetCode;

import java.util.LinkedList;
import java.util.Queue;

public class L225_ImplementStackUsingQueues {
    /**
```

```
 * Your MyStack object will be instantiated and called as such:
 * MyStack obj = new MyStack();
 * obj.push(x);
 * int param_2 = obj.pop();
 * int param_3 = obj.top();
 * boolean param_4 = obj.empty();
 */
/** Initialize your data structure here. */
Queue<Integer> queue;

public L225_ImplementStackUsingQueues() {
    this.queue=new LinkedList<Integer>();
    //this.queue=new PriorityQueue<Integer>();//优先级队列为什么不行？加入到此 queue 中的元素天然有序，通过 java.util.Comparable 实现。改变了队列的原有顺序
}

/** Push element x onto stack. */
public void push(int x) {//转换队列中元素的顺序，后进的放在最前
    queue.add( x);
    for(int i=0;i<queue.size()-1;i++)//<queue.size()的话，相当于重新进出一遍，不能实现倒置
        queue.add(queue.poll());
}

/** Removes the element on top of the stack and returns that element. */
public int pop() {
    return queue.poll();
}

/** Get the top element. */
public int top() {
    return queue.peek();
}

/** Returns whether the stack is empty. */
public boolean empty() {
    return queue.isEmpty();
}

public static void main(String[] args){
    L225_ImplementStackUsingQueues l225=new L225_ImplementStackUsingQueues();
```

```
        1225.push(1);
        1225.push(2);
        System.out.println(1225.top());
        System.out.println(1225.pop());
        System.out.println(1225.empty());
    }
}
```

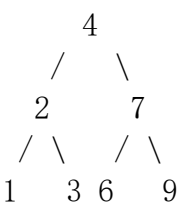
226. Invert Binary Tree

分析
题目

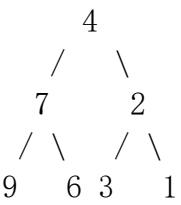
Invert a binary tree.

Example:

Input:



Output:



Trivia:

This problem was inspired by this original tweet by Max Howell:

Google: 90% of our engineers use the software you wrote (Homebrew), but you can’t invert a binary tree on a whiteboard so f*** off.

解答

```
package LeetCode;

public class L226_InvertBinaryTree {
    public TreeNode invertTree(TreeNode root) {
        if(root==null)
            return null;
        TreeNode left=root.right;//直接交换左右子节点
        TreeNode right=root.left;
        root.left=invertTree(left);//root 的子节点与翻转后的节点建立
关联
        root.right=invertTree(right);
        return root;
```

```
        }
    }
```

231. Power of Two

分析
题目

Given an integer, write a function to determine if it is a power of two.

Example 1:

Input: 1
Output: true
Explanation: $2^0 = 1$

Example 2:

Input: 16
Output: true
Explanation: $2^4 = 16$

Example 3:

Input: 218
Output: false
解答

```
package LeetCode;

public class L231_PowerOfTwo {
    public boolean isPowerOfTwo(int n) {
        if(n<=0)
            return false;
        return ((n&(n-1))==0);//2的幂次，最高位为1，其余位全为0.与n-1各位都不相同，为0
    }
    public static void main(String[] args){
        L231_PowerOfTwo l231=new L231_PowerOfTwo();
        int num=218;
        System.out.println(l231.isPowerOfTwo(num));
    }
}
```

232. Implement Queue using Stacks

分析
题目

Implement the following operations of a queue using stacks.

- push(x) -- Push element x to the back of queue.
- pop() -- Removes the element from in front of queue.
- peek() -- Get the front element.
- empty() -- Return whether the queue is empty.

Example:

```
MyQueue queue = new MyQueue();

queue.push(1);
queue.push(2);
queue.peek();  // returns 1
queue.pop();   // returns 1
queue.empty(); // returns false
```

Notes:

- You must use *only* standard operations of a stack -- which means only `push to top`, `peek/pop from top`, `size`, and `is empty` operations are valid.
- Depending on your language, stack may not be supported natively. You may simulate a stack by using a list or deque (double-ended queue), as long as you use only standard operations of a stack.
- You may assume that all operations are valid (for example, no pop or peek operations will be called on an empty queue).

解答

```
package LeetCode;

import java.util.Stack;

public class L232_ImplementQueueUsingStacks {
    /** Initialize your data structure here. */
    Stack<Integer> stack1;
    Stack<Integer> stack2;
    public L232_ImplementQueueUsingStacks() {
        this.stack1=new Stack<Integer>();
```

```
        this.stack2=new Stack<Integer>();
    }

    /** Push element x to the back of queue. */
    public void push(int x) {
        stack1.push(x);
    }

    /** Removes the element from in front of queue and returns that
element. */
    public int pop() {
        while(!stack1.isEmpty()){
            stack2.push(stack1.pop());
        }
        int temp=stack2.pop();
        while(!stack2.isEmpty()){
            stack1.push(stack2.pop());
        }
        return temp;
    }

    /** Get the front element. */
    public int peek() {
        while(!stack1.isEmpty()){
            stack2.push(stack1.pop());
        }
        int temp =stack2.peek();
        while(!stack2.isEmpty()){
            stack1.push(stack2.pop());
        }
        return temp;
    }

    /** Returns whether the queue is empty. */
    public boolean empty() {
        return stack1.isEmpty();
    }

    public static void main(String[] args){
        L232_ImplementQueueUsingStacks l232=new
L232_ImplementQueueUsingStacks();
        l232.push(1);
        l232.push(2);
        System.out.println(l232.peek());
```

```
        System.out.println(l232.pop());
        System.out.println(l232.empty());
    }
}
```

234. Palindrome Linked List

分析
难度 易
来源
https://leetcode.com/problems/palindrome-linked-list/
题目

Given a singly linked list, determine if it is a palindrome.

Example 1:

Input: 1->2
Output: false

Example 2:

Input: 1->2->2->1
Output: true

Follow up:

Could you do it in O(n) time and O(1) space?

解答

```
package LeetCode;

public class L234_PalindromeLinkedList {
    public ListNode buildList(int[] nums){
        ListNode head;
        ListNode pointer;
        pointer=head=new ListNode(nums[0]);
        for(int i=1;i<nums.length;i++)
        {
            pointer.next=new ListNode(nums[i]);
            pointer=pointer.next;
        }
        return head;
    }
    public ListNode reverList(ListNode ln){//翻转从 ln 开始的列表
        if(ln==null||ln.next==null)
            return ln;
        ListNode cur,next,temp;
```

```
            cur=ln.next;
            next=ln;
            next.next=null;//初始化翻转后的列表尾部
            while(cur!=null){
                temp=cur.next;
                cur.next=next;
                next=cur;
                cur=temp;
            }
            return next;
        }
        public boolean isPalindrome(ListNode head) {//列表是否回文
            //快慢指针
            ListNode fast=head,slow=head;
            while(fast!=null&&fast.next!=null)
            {
                fast=fast.next.next;
                slow=slow.next;//fast 非空的时候 slow 肯定非空
            }
            ListNode half=reverList(slow);//后半段头结点
            ListNode pointer=head,pointer1=half;//pointer 为前半段上的指
针，pointer1 为后半段上的指针
            while(pointer!=half&&pointer1!=null)//列表长为奇数时，翻转队
列较剩余队列长 1。比较至剩余队列结束
            {
                if(pointer.val!=pointer1.val)
                    return false;
                pointer=pointer.next;
                pointer1=pointer1.next;
            }
            return true;
        }
        public static void main(String[] args){
            int[] nums={1,0,1};
            L234_PalindromeLinkedList l234=new
L234_PalindromeLinkedList();
            ListNode ln=l234.buildList(nums);
            System.out.println(l234.isPalindrome(ln));
        }
}
```

235. Lowest Common Ancestor of a Binary Search Tree

分析

二叉查找树（Binary Search Tree），（又：二叉搜索树，二叉排序树）它或者是一棵空树，或者是具有下列性质的二叉树： 若它的左子树不空，则左子树上所有结点的值均小于它的根结点的值； 若它的右子树不空，则右子树上所有结点的值均大于它的根结点的值； 它的左、右子树也分别为二叉排序树。

难度 易

来源

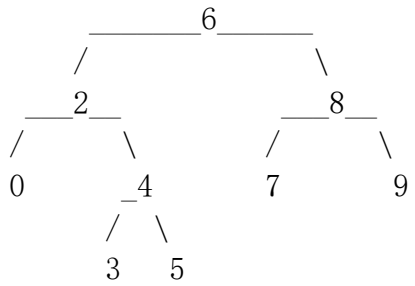
https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-search-tree/

题目

Given a binary search tree (BST), find the lowest common ancestor (LCA) of two given nodes in the BST.

According to the definition of LCA on Wikipedia: “The lowest common ancestor is defined between two nodes p and q as the lowest node in T that has both p and q as descendants (where we allow **a node to be a descendant of itself**).”

Given binary search tree: root = [6,2,8,0,4,7,9,null,null,3,5]



Example 1:

Input: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 8
Output: 6
Explanation: The LCA of nodes 2 and 8 is 6.

Example 2:

Input: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 4
Output: 2
Explanation: The LCA of nodes 2 and 4 is 2, since a node can be a descendant of itself
according to the LCA definition.

Note:

- All of the nodes' values will be unique.
- p and q are different and both values will exist in the BST.

解答

```
package LeetCode;

import java.util.LinkedList;
import java.util.Stack;

public class L235_LowestCommonAncestorOfABinarySearchTree {
    public TreeNode lowestCommonAncestor(TreeNode root, TreeNode p,
TreeNode q) {//返回最低公共父节点，不是公共序列
        if(p.val<root.val&&q.val<root.val){
            return lowestCommonAncestor(root.left,p,q);//在根节点左侧
        } else if(p.val>root.val&&q.val>root.val){
            return lowestCommonAncestor(root.right,p,q);//在根节点右
侧
        }
        else
            return root;//在根节点两侧
    }
    public static void main(String[] args){
        Integer[] nums={6,2,8,0,4,7,9,null,null,3,5};
        Tree tree=new Tree();
        TreeNode root=tree.buildTree(nums);
        //tree.printTree(root);
        L235_LowestCommonAncestorOfABinarySearchTree l235=new
L235_LowestCommonAncestorOfABinarySearchTree();
        TreeNode p=new TreeNode(2);
        TreeNode q=new TreeNode(4);
        System.out.println(l235.lowestCommonAncestor(root,p,q).val);
    }
}
```

237. Delete Node in a Linked List

分析

高赞说的没错，这题是有点傻逼呢，自己写 playground 没法写。Delete 的 node 和 head 的 list 里的 node 不是一个啊
难度 易
来源 https://leetcode.com/problems/delete-node-in-a-linked-list/

题目

Write a function to delete a node (except the tail) in a singly linked list, given only access to that node.

Given linked list -- head = [4,5,1,9], which looks like following:

```
    4 -> 5 -> 1 -> 9
```

Example 1:

```
Input: head = [4,5,1,9], node = 5
Output: [4,1,9]
Explanation: You are given the second node with value 5, the linked list
             should become 4 -> 1 -> 9 after calling your function.
```

Example 2:

```
Input: head = [4,5,1,9], node = 1
Output: [4,5,9]
Explanation: You are given the third node with value 1, the linked list
            should become 4 -> 5 -> 9 after calling your function.
```

Note:

- The linked list will have at least two elements.
- All of the nodes' values will be unique.
- The given node will not be the tail and it will always be a valid node of the linked list.
- Do not return anything from your function.

解答

```java
package LeetCode;

public class L237_DeleteNodeinaLinkedList {
    public static ListNode head;
    public void deleteNode(ListNode node) {//难点在于链表的获取？
        node.val=node.next.val;//不是最后一个 node 的用处。通过将值改为下一个的值，下一个节点改为下下一个节点实现删除，有点意思
        node.next=node.next.next;
    }
    public static void main(String[] args){
        int[] nums={4,5,1,9};
        ListNode pointer=ListNode.buildListNode(nums);
        L237_DeleteNodeinaLinkedList l237=new L237_DeleteNodeinaLinkedList();
        ListNode node=pointer.next;
        l237.deleteNode(node);
        while(pointer!=null){
            System.out.print(pointer.val+"\t");
            pointer=pointer.next;
        }
    }
}
```

240. Search a 2D Matrix II

分析

难度
来源

题目

Write an efficient algorithm that searches for a value in an *m* x *n* matrix. This matrix has the following properties:

- Integers in each row are sorted in ascending from left to right.
- Integers in each column are sorted in ascending from top to bottom.

Example:

Consider the following matrix:

```
[
  [1,   4,  7, 11, 15],
  [2,   5,  8, 12, 19],
  [3,   6,  9, 16, 22],
  [10, 13, 14, 17, 24],
  [18, 21, 23, 26, 30]
]
```

解答

https://leetcode.com/problems/search-a-2d-matrix-ii/discuss/66140/My-concise-O(m%2Bn)-Java-solution

```
package LeetCode;

public class L240_SearchA2DMatrixII {
    public boolean searchMatrix(int[][] matrix, int target) {
        if(matrix==null||matrix.length==0||matrix[0].length==0)
            return false;//空矩阵
        //重点：两个可移动方向，一个变大一个缩小。
        int row=0,col=matrix[0].length-1;
        while(col>=0&&row<matrix.length){
            if(matrix[row][col]==target)
```

```
                return true;
            else if(matrix[row][col]>target)
                col--;
            else
                row++;
        }
        return false;
    }
    public static void main(String[] args){
      /*  int[][] matrix={
                                {1,   4,   7,  11,  15},
                                {2,   5,   8,  12,  19},
                                {3,   6,   9,  16,  22},
                                {10,  13,  14,  17,  24},
                                {18,  21,  23,  26,  30}
                            };
        int target=10;*/
        int[][] matrix={
                {   1,   4,   7,   11,  15},
                {   2,   5,   8,   12,  19},
                {   3,   6,   9,   16,  22},
                {   10,  13,  14,  17,  24},
                {   18,  21,  23,  26,  30}
        };
        int target=5;
        L240_SearchA2DMatrixII 1240=new L240_SearchA2DMatrixII();

        System.out.println(1240.searchMatrix(matrix,target));
    }
}
```

242. Valid Anagram

分析

Java 中 Arrays 类的两个方法：deepEquals 和 equals

1、deepEquals 用于判定两个指定数组彼此是否深层相等，此方法适用于任意深度的嵌套数组。

2、equals 用于判定两个数组是否相等，如果两个数组以相同顺序包含相同元素，则返回 true，否则返回 false。

3、通过比对“代码一”和“代码二”我们可以得出这样一个结论：如果两个数组使用 equals 返回 true，则使用 deepEquals 也返回 true，也就是说在比较的两个数组均为一维数组的前提下，equals 和 deepEquals 的比较结果没有差别；

4、如果要比较多为数组，则需要使用 deepEquals 方法；

作者：不动声色的蜗牛
来源：CSDN
原文：https://blog.csdn.net/gaohuanjie/article/details/39157241
版权声明：本文为博主原创文章，转载请附上博文链接！

难度 易
来源
https://leetcode.com/problems/valid-anagram/

题目

Given two strings *s* and *t* , write a function to determine if *t* is an anagram of *s*.

Example 1:

Input: *s* = "anagram", *t* = "nagaram"
Output: true

Example 2:

Input: *s* = "rat", *t* = "car"
Output: false

Note:

You may assume the string contains only lowercase alphabets.

Follow up:

What if the inputs contain unicode characters? How would you adapt your solution to such case?

解答

```
package LeetCode;

import java.util.Arrays;

public class L242_ValidAnagram {
    /*public boolean isAnagram(String s, String t) {
        int[] charcount1=new int[26];//考虑大小写及特殊字符
        int[] charcount2=new int[26];

        for(int i=0;i<s.length();i++){
            charcount1[s.charAt(i)-'a']++;
        }
        for(int i=0;i<t.length();i++){
            charcount2[t.charAt(i)-'a']++;
        }
*//*         for(int i=0;i<26;i++)
            System.out.print(charcount1[i]+"\t");
        System.out.println();
        for(int i=0;i<26;i++)
            System.out.print(charcount2[i]+"\t");*//*
        if(Arrays.equals(charcount1,charcount2))
            return true;
        else
            return false;
    }*/
    public boolean isAnagram(String s, String t) {
        char[] s1=s.toCharArray();//考虑大小写及特殊字符
        char[] s2=t.toCharArray();
        Arrays.sort(s1);
        Arrays.sort(s2);
        if(Arrays.equals(s1,s2))
            return true;
        else
            return false;
    }
    public static void main(String[] args){
        String s="anagram";
```

```
        String t="nagaram";
        L242_ValidAnagram l242=new L242_ValidAnagram();
        System.out.println(l242.isAnagram(s,t));

    }

}
```

249. Group Shifted Strings

分析

难度

来源

https://leetcode.com/problems/group-shifted-strings/

（题目是收费的，从 49 题看推荐知道的，没看到题目内容，从别人那儿拷的。用例是抄的。不保证 ac）

题目

Given a string, we can "shift" each of its letter to its successive letter, for example: "abc" -> "bcd". We can keep "shifting" which forms the sequence:

"abc" -> "bcd" -> ... -> "xyz"

Given a list of strings which contains only lowercase alphabets, group all strings that belong to the same shifting sequence.

For example, given: ["abc", "bcd", "acef", "xyz", "az", "ba", "a", "z"],
A solution is:

```
[
  ["abc","bcd","xyz"],
  ["az","ba"],
  ["acef"],
  ["a","z"]
]
```

解答

```
package LeetCode;

import java.util.Arrays;
import java.util.HashMap;
import java.util.LinkedList;
```

```
import java.util.List;

public class L249_GroupShiftedStrings {
    public String getBase(String str){//将平移字符串的base定为首字母为a的字符串
        int distance=str.charAt(0)-'a';//平移的距离
        StringBuilder sb=new StringBuilder();
        sb.append('a');
        for(int i=1;i<str.length();i++){
            if(str.charAt(i)-'a'<distance)
                sb.append((char)(str.charAt(i)+26-distance));
            else
                sb.append((char) (str.charAt(i)-distance));
        }
        return sb.toString();
    }
    public List<List<String>> groupStrings(String[] strs) {
        List<List<String>> res=new LinkedList<List<String>>();
        HashMap<String,Integer> hm=new HashMap<String,Integer>();//前边为字符串，后边为字符串对应list的序号
        int listsnum=0;//list数目
        for(int i=0;i<strs.length;i++){
            String str=getBase(strs[i]);
            if(!hm.containsKey(str))//temp.toString不是temp里的字符组成字符串
            {
                hm.put(str,listsnum);
                List<String> list=new LinkedList<String>();
                list.add(strs[i]);
                res.add(list);
                listsnum++;
            }else{
                res.get(hm.get(str)).add(strs[i]);//找到对应的String list，往里添加string
            }
        }
        return res;
    }
    public static void main(String[] args){
        L249_GroupShiftedStrings l249=new L249_GroupShiftedStrings();
        //System.out.println(l249.getBase("cde"));
        String[] strs={"abc", "bcd", "acef", "xyz", "az", "ba", "a", "z"};
        System.out.println(l249.groupStrings(strs));
```

```
        }
    }
```

257. Binary Tree Paths

分析

深度优先

英文缩写为 DFS 即 Depth First Search.其过程简要来说是对每一个可能的分支路径深入到不能再深入为止，而且每个节点只能访问一次。

深度优先遍历各个节点，需要使用到栈（Stack）这种数据结构。

stack 的特点是是先进后出。整个遍历过程如下：

先往栈中压入右节点，再压左节点，这样出栈就是先左节点后右节点了。

题目

Given a binary tree, return all root-to-leaf paths.

Note: A leaf is a node with no children.

Example:

Input:

```
   1
 /   \
2     3
 \
  5
```

Output: ["1->2->5", "1->3"]

Explanation: All root-to-leaf paths are: 1->2->5, 1->3

解答

```java
package LeetCode;
import java.util.LinkedList;
import java.util.List;

public class L257_BinaryTreePaths {
   public List<String> binaryTreePaths(TreeNode root) {//使用递归
        List<String> paths= new LinkedList<String>();
        if(root==null)
            return paths;
        if(root.left==null&&root.right==null)//根节点为叶子节点
            paths.add(String.valueOf(root.val));
        for(String path:binaryTreePaths(root.left)){
            paths.add(root.val+"->"+path);
        }
        for(String path:binaryTreePaths(root.right)){
            paths.add(root.val+"->"+path);
        }
        return paths;
    }
    public static void main(String[] args){
        L257_BinaryTreePaths l257=new L257_BinaryTreePaths();
        Integer[] nums={37,-34,-48,null,-100,-
100,48,null,null,null,null,-54,null,-71,-22,null,null,null,8};
        //Expected ["37->-34->-100","37->-48->-100","37->-48->48->-
54->-71","37->-48->48->-54->-22->8"]
        Tree tree=new Tree();
        TreeNode root=tree.buildTree(nums);
        //tree.printTree(root);
        List<String> answer =l257.binaryTreePaths(root);
        System.out.println(answer);
    }
}
```

263. Ugly Number

DescriptionHintsSubmissionsDiscussSolution

Write a program to check whether a given number is an ugly number.

Ugly numbers are **positive numbers** whose prime factors only include 2, 3, 5.

Example 1:

Input: 6
Output: true
Explanation: 6 = 2 × 3

Example 2:

Input: 8
Output: true
Explanation: 8 = 2 × 2 × 2

Example 3:

Input: 14
Output: false
Explanation: 14 is not ugly since it includes another prime factor 7.

Note:

1. 1 is typically treated as an ugly number.
2. Input is within the 32-bit signed integer range: $[-2^{31}, 2^{31} - 1]$.

268. Missing Number

DescriptionHintsSubmissionsDiscussSolution

Given an array containing *n* distinct numbers taken from 0, 1, 2, ..., n, find the one that is missing from the array.

Example 1:

Input: [3,0,1]

Output: 2

Example 2:

Input: [9,6,4,2,3,5,7,0,1]
Output: 8

Note:

Your algorithm should run in linear runtime complexity. Could you implement it using only constant extra space complexity?

278. First Bad Version

DescriptionHintsSubmissionsDiscussSolution

You are a product manager and currently leading a team to develop a new product. Unfortunately, the latest version of your product fails the quality check. Since each version is developed based on the previous version, all the versions after a bad version are also bad.

Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad.

You are given an API bool isBadVersion(version) which will return whether version is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

Example:

Given n = 5, and version = 4 is the first bad version.

```
call isBadVersion(3) -> false
call isBadVersion(5)  -> true
call isBadVersion(4)  -> true
```

Then 4 is the first bad version.

279. Perfect Squares

分析

常规思路是使用动归解决（大家的讨论也都是围绕 DP 进行的），但是如果你了解一个定理：四平方和定理：任一个正整数都能拆成四个整数的平方和；
难度 中
来源
https://leetcode.com/problems/perfect-squares/

题目

Given a positive integer *n*, find the least number of perfect square numbers (for example, 1, 4, 9, 16, ...) which sum to *n*.

Example 1:

Input: *n* = 12
Output: 3
Explanation: 12 = 4 + 4 + 4.

Example 2:

Input: *n* = 13
Output: 2
Explanation: 13 = 4 + 9.

解答

https://leetcode.com/problems/perfect-squares/discuss/71605/Java-DP-Solution-with-explanation

```
package LeetCode;

public class L279_PerfectSquares {
    public int numSquares(int n) {
        int[] dp=new int[n+1];
        for(int i=1;i<=n;i++)//dp[i]表示到 i 为止的最少平方数的数目
            dp[i]=Integer.MAX_VALUE;
        for(int i=1;i<=n;i++){
            int sqrt=(int)Math.sqrt(i);
            if(sqrt*sqrt==i)
            {
                dp[i]=1;
                continue;//后边就不需要执行了
            }
```

```
            for(int j=1;j<=sqrt;j++){
                int diff=i-j*j;
                dp[i]=Math.min(dp[i],dp[diff]+1);
            }
        }
        return dp[n];
    }
    public static void main(String[] args){
        L279_PerfectSquares l279=new L279_PerfectSquares();
        int n=43;
        //Input
        //43
        //Output
        //4
        //Expected
        //3
        System.out.println(l279.numSquares(n));
    }
}
```

283. Move Zeroes

DescriptionHintsSubmissionsDiscussSolution

Given an array nums, write a function to move all 0's to the end of it while maintaining the relative order of the non-zero elements.

Example:

Input: [0,1,0,3,12]
Output: [1,3,12,0,0]

Note:

1. You must do this **in-place** without making a copy of the array.
2. Minimize the total number of operations.

287. Find the Duplicate Number

分析

难度 中
来源
https://leetcode.com/problems/find-the-duplicate-number/
博客
https://www.cnblogs.com/flowingfog/p/9929321.html
https://blog.csdn.net/flowingfog/article/details/83862028

题目

Given an array *nums* containing *n* + 1 integers where each integer is between 1 and *n* (inclusive), prove that at least one duplicate number must exist. Assume that there is only one duplicate number, find the duplicate one.

Example 1:

```
Input: [1,3,4,2,2]
Output: 2
```

Example 2:

```
Input: [3,1,3,4,2]
Output: 3
```

Note:

1. You **must not** modify the array (assume the array is read only).
2. You must use only constant, $O(1)$ extra space.
3. Your runtime complexity should be less than $O(n^2)$.
4. There is only one duplicate number in the array, but it could be repeated more than once.

解答

```
package LeetCode;

public class L287_FindTheDuplicateNumber {
    public int findDuplicate(int[] nums) {
        int slow=nums[0];
        int fast=nums[nums[0]];
```

```
        while(slow!=fast){
            slow=nums[slow];
            fast=nums[nums[fast]];
        }
        int slow2=0;
        while(slow2!=slow){
            slow=nums[slow];
            slow2=nums[slow2];
        }
        return slow;
    }
    public static void main(String[] args){
        L287_FindTheDuplicateNumber l287=new
L287_FindTheDuplicateNumber();
        int[] nums={3,1,3,4,2};
        System.out.println(l287.findDuplicate(nums));
    }
}
```

290. Word Pattern

DescriptionHintsSubmissionsDiscussSolution

Given a pattern and a string str, find if str follows the same pattern.

Here **follow** means a full match, such that there is a bijection between a letter in pattern and a **non-empty** word in str.

Example 1:

Input: pattern = "abba", str = "dog cat cat dog"
Output: true

Example 2:

Input:pattern = "abba", str = "dog cat cat fish"
Output: false

Example 3:

Input: pattern = "aaaa", str = "dog cat cat dog"
Output: false

Example 4:

Input: pattern = "abba", str = "dog dog dog dog"
Output: false

Notes:

You may assume pattern contains only lowercase letters, and str contains lowercase letters separated by a single space.

---

292. Nim Game

DescriptionHintsSubmissionsDiscussSolution

You are playing the following Nim Game with your friend: There is a heap of stones on the table, each time one of you take turns to remove 1 to 3 stones. The one who removes the last stone will be the winner. You will take the first turn to remove the stones.

Both of you are very clever and have optimal strategies for the game. Write a function to determine whether you can win the game given the number of stones in the heap.

Example:

Input: 4
Output: false
Explanation: If there are 4 stones in the heap, then you will never win the game;
No matter 1, 2, or 3 stones you remove, the last stone will always be
removed by your friend.

303. Range Sum Query - Immutable

DescriptionHintsSubmissionsDiscussSolution

Given an integer array *nums*, find the sum of the elements between indices *i* and *j* (*i* ≤ *j*), inclusive.

Example:

Given nums = [-2, 0, 3, -5, 2, -1]

sumRange(0, 2) -> 1
sumRange(2, 5) -> -1
sumRange(0, 5) -> -3

Note:

1. You may assume that the array does not change.
2. There are many calls to *sumRange* function.

326. Power of Three

DescriptionHintsSubmissionsDiscussSolution

Given an integer, write a function to determine if it is a power of three.

Example 1:

```
Input: 27
Output: true
```

Example 2:

```
Input: 0
Output: false
```

Example 3:

```
Input: 9
Output: true
```

Example 4:

```
Input: 45
Output: false
```

Follow up:

Could you do it without using any loop / recursion?

342. Power of Four

DescriptionHintsSubmissionsDiscussSolution

Given an integer (signed 32 bits), write a function to check whether it is a power of 4.

Example 1:

```
Input: 16
```

Output: true

Example 2:

Input: 5
Output: false

Follow up: Could you solve it without loops/recursion?

344. Reverse String

DescriptionHintsSubmissionsDiscussSolution

Write a function that takes a string as input and returns the string reversed.

Example 1:

Input: "hello"
Output: "olleh"

Example 2:

Input: "A man, a plan, a canal: Panama"
Output: "amanaP :lanac a ,nalp a ,nam A"

345. Reverse Vowels of a String

DescriptionHintsSubmissionsDiscussSolution

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".

350. Intersection of Two Arrays II

DescriptionHintsSubmissionsDiscussSolution

Given two arrays, write a function to compute their intersection.

Example 1:

```
Input: nums1 = [1,2,2,1], nums2 = [2,2]
Output: [2,2]
```

Example 2:

```
Input: nums1 = [4,9,5], nums2 = [9,4,9,8,4]
Output: [4,9]
```

Note:

- Each element in the result should appear as many times as it shows in both arrays.
- The result can be in any order.

Follow up:

- What if the given array is already sorted? How would you optimize your algorithm?
- What if *nums1*'s size is small compared to *nums2*'s size? Which algorithm is better?
- What if elements of *nums2* are stored on disk, and the memory is limited such that you cannot load all elements into the memory at once?

367. Valid Perfect Square

DescriptionHintsSubmissionsDiscussSolution

Given a positive integer *num*, write a function which returns True if *num* is a perfect square else False.

Note: Do not use any built-in library function such as sqrt.

Example 1:

Input: 16
Output: true

Example 2:

Input: 14
Output: false

371. Sum of Two Integers

DescriptionHintsSubmissionsDiscussSolution

Calculate the sum of two integers *a* and *b*, but you are **not allowed** to use the operator + and -.

Example 1:

Input: a = 1, b = 2
Output: 3

Example 2:

Input: a = -2, b = 3
Output: 1

374. Guess Number Higher or Lower

DescriptionHintsSubmissionsDiscussSolution

We are playing the Guess Game. The game is as follows:

I pick a number from **1** to ***n***. You have to guess which number I picked.

Every time you guess wrong, I'll tell you whether the number is higher or lower.

You call a pre-defined API guess(int num) which returns 3 possible results (-1, 1, or 0):

```
-1 : My number is lower
 1 : My number is higher
 0 : Congrats! You got it!
```

Example :

```
Input: n = 10, pick = 6
Output: 6
```

383. Ransom Note

DescriptionHintsSubmissionsDiscussSolution

Given an arbitrary ransom note string and another string containing letters from all the magazines, write a function that will return true if the ransom note can be constructed from the magazines ; otherwise, it will return false.

Each letter in the magazine string can only be used once in your ransom note.

Note:

You may assume that both strings contain only lowercase letters.

```
canConstruct("a", "b") -> false
canConstruct("aa", "ab") -> false
canConstruct("aa", "aab") -> true
```

387. First Unique Character in a String

DescriptionHintsSubmissionsDiscussSolution

Given a string, find the first non-repeating character in it and return it's index. If it doesn't exist, return -1.

Examples:

```
s = "leetcode"
return 0.

s = "loveleetcode",
return 2.
```

Note: You may assume the string contain only lowercase letters.

---

389. Find the Difference

DescriptionHintsSubmissionsDiscussSolution

Given two strings *s* and *t* which consist of only lowercase letters.

String *t* is generated by random shuffling string *s* and then add one more letter at a random position.

Find the letter that was added in *t*.

Example:

```
Input:
s = "abcd"
t = "abcde"

Output:
e

Explanation:
'e' is the letter that was added.
```

400. Nth Digit

DescriptionHintsSubmissionsDiscussSolution

Find the n^{th} digit of the infinite integer sequence 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...

Note:

n is positive and will fit within the range of a 32-bit signed integer ($n < 2^{31}$).

Example 1:

Input:
3

Output:
3

Example 2:

Input:
11

Output:
0

Explanation:
The 11th digit of the sequence 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ... is a 0, which is part of the number 10.

401. Binary Watch

DescriptionHintsSubmissionsDiscussSolution

A binary watch has 4 LEDs on the top which represent the **hours** (0-11), and the 6 LEDs on the bottom represent the **minutes** (0-59).

Each LED represents a zero or one, with the least significant bit on the right.

For example, the above binary watch reads "3:25".

Given a non-negative integer *n* which represents the number of LEDs that are currently on, return all possible times the watch could represent.

Example:

```
Input: n = 1
Return: ["1:00", "2:00", "4:00", "8:00", "0:01", "0:02", "0:04", "0:08", "0:16", "0:32"]
```

Note:

- The order of output does not matter.
- The hour must not contain a leading zero, for example "01:00" is not valid, it should be "1:00".
- The minute must be consist of two digits and may contain a leading zero, for example "10:2" is not valid, it should be "10:02".

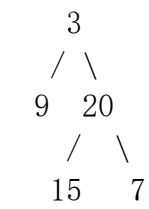
---

404. Sum of Left Leaves

DescriptionHintsSubmissionsDiscussSolution

Find the sum of all left leaves in a given binary tree.

Example:



There are two left leaves in the binary tree, with values 9 and 15 respectively. Return 24.

405. Convert a Number to Hexadecimal

DescriptionHintsSubmissionsDiscussSolution

Given an integer, write an algorithm to convert it to hexadecimal. For negative integer, two's complement method is used.

Note:

1. All letters in hexadecimal (a-f) must be in lowercase.
2. The hexadecimal string must not contain extra leading 0s. If the number is zero, it is represented by a single zero character '0'; otherwise, the first character in the hexadecimal string will not be the zero character.
3. The given number is guaranteed to fit within the range of a 32-bit signed integer.
4. You **must not use *any* method provided by the library** which converts/formats the number to hex directly.

Example 1:

```
Input:
26

Output:
"1a"
```

Example 2:

```
Input:
-1

Output:
"ffffffff"
```

409. Longest Palindrome

DescriptionHintsSubmissionsDiscussSolution

Given a string which consists of lowercase or uppercase letters, find the length of the longest palindromes that can be built with those letters.

This is case sensitive, for example "Aa" is not considered a palindrome here.

Note:

Assume the length of given string will not exceed 1,010.

Example:

Input:
"abccccdd"

Output:
7

Explanation:
One longest palindrome that can be built is "dccaccd", whose length is 7.

412. Fizz Buzz

DescriptionHintsSubmissionsDiscussSolution

Write a program that outputs the string representation of numbers from 1 to *n*.

But for multiples of three it should output “Fizz” instead of the number and for the multiples of five output “Buzz”. For numbers which are multiples of both three and five output “FizzBuzz”.

Example:

```
n = 15,

Return:
[
    "1",
    "2",
    "Fizz",
    "4",
    "Buzz",
    "Fizz",
    "7",
    "8",
    "Fizz",
    "Buzz",
    "11",
    "Fizz",
    "13",
    "14",
    "FizzBuzz"
]
```

414. Third Maximum Number

DescriptionHintsSubmissionsDiscussSolution

Given a **non-empty** array of integers, return the **third** maximum number in this array. If it does not exist, return the maximum number. The time complexity must be in O(n).

Example 1:

Input: [3, 2, 1]

Output: 1

Explanation: The third maximum is 1.

Example 2:

Input: [1, 2]

Output: 2

Explanation: The third maximum does not exist, so the maximum (2) is returned instead.

Example 3:

Input: [2, 2, 3, 1]

Output: 1

Explanation: Note that the third maximum here means the third maximum distinct number.
Both numbers with value 2 are both considered as second maximum.

415. Add Strings

DescriptionHintsSubmissionsDiscussSolution

Given two non-negative integers num1 and num2 represented as string, return the sum of num1 and num2.

Note:

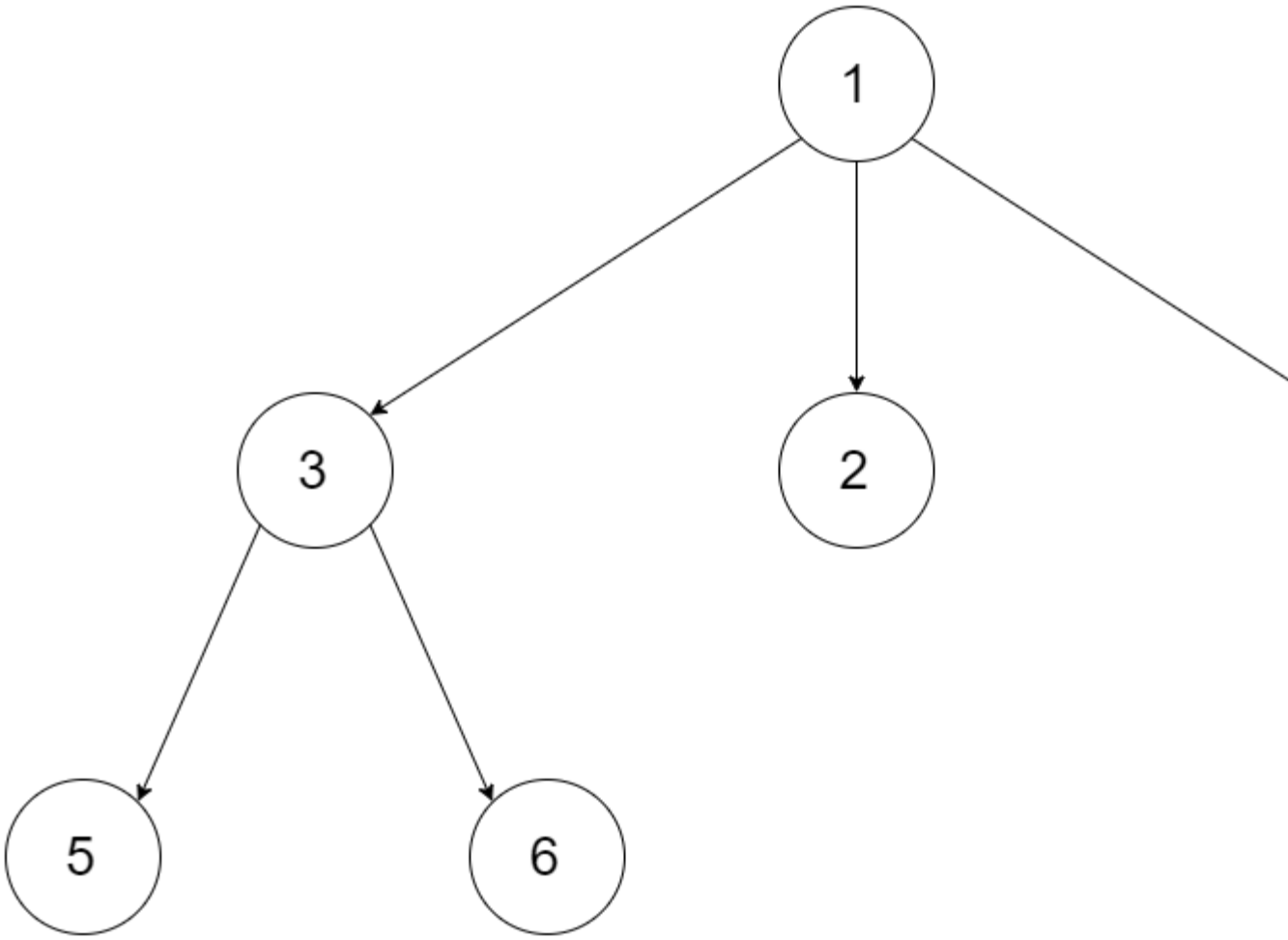
1. The length of both num1 and num2 is < 5100.
2. Both num1 and num2 contains only digits 0-9.
3. Both num1 and num2 does not contain any leading zero.
4. You **must not use any built-in BigInteger library** or **convert the inputs to integer** directly.

429. N-ary Tree Level Order Traversal

DescriptionHintsSubmissionsDiscussSolution

Given an n-ary tree, return the level order traversal of its nodes' values. (ie, from left to right, level by level).

For example, given a 3-ary tree:



We should return its level order traversal:

```
[
      [1],
      [3,2,4],
      [5,6]
]
```

Note:

1. The depth of the tree is at most 1000.
2. The total number of nodes is at most 5000.

434. Number of Segments in a String

DescriptionHintsSubmissionsDiscussSolution

Count the number of segments in a string, where a segment is defined to be a contiguous sequence of non-space characters.

Please note that the string does not contain any **non-printable** characters.

Example:

Input: "Hello, my name is John"
Output: 5

---

437. Path Sum III

DescriptionHintsSubmissionsDiscussSolution

You are given a binary tree in which each node contains an integer value.

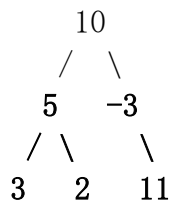
Find the number of paths that sum to a given value.

The path does not need to start or end at the root or a leaf, but it must go downwards (traveling only from parent nodes to child nodes).

The tree has no more than 1,000 nodes and the values are in the range -1,000,000 to 1,000,000.

Example:

root = [10,5,-3,3,2,null,11,3,-2,null,1], sum = 8



```
 / \   \
3  -2   1
```

Return 3. The paths that sum to 8 are:

1. 5 -> 3
2. 5 -> 2 -> 1
3. -3 -> 11

438. Find All Anagrams in a String

DescriptionHintsSubmissionsDiscussSolution

Given a string **s** and a **non-empty** string **p**, find all the start indices of **p**'s anagrams in **s**.

Strings consists of lowercase English letters only and the length of both strings **s** and **p** will not be larger than 20,100.

The order of output does not matter.

Example 1:

Input:
s: "cbaebabacd" p: "abc"

Output:
[0, 6]

Explanation:
The substring with start index = 0 is "cba", which is an anagram of "abc".
The substring with start index = 6 is "bac", which is an anagram of "abc".

Example 2:

Input:
s: "abab" p: "ab"

Output:
[0, 1, 2]

Explanation:
The substring with start index = 0 is "ab", which is an anagram of "ab".
The substring with start index = 1 is "ba", which is an anagram of "ab".
The substring with start index = 2 is "ab", which is an anagram of "ab".

441. Arranging Coins

DescriptionHintsSubmissionsDiscussSolution

You have a total of *n* coins that you want to form in a staircase shape, where every *k*-th row must have exactly *k* coins.

Given *n*, find the total number of **full** staircase rows that can be formed.

n is a non-negative integer and fits within the range of a 32-bit signed integer.

Example 1:

```
n = 5

The coins can form the following rows:
¤
¤ ¤
¤ ¤

Because the 3rd row is incomplete, we return 2.
```

Example 2:

```
n = 8

The coins can form the following rows:
¤
¤ ¤
¤ ¤ ¤
¤ ¤
```

Because the 4th row is incomplete, we return 3.

443. String Compression

DescriptionHintsSubmissionsDiscussSolution

Given an array of characters, compress it in-place.

The length after compression must always be smaller than or equal to the original array.

Every element of the array should be a **character** (not int) of length 1.

After you are done **modifying the input array** in-place, return the new length of the array.

Follow up:

Could you solve it using only O(1) extra space?

Example 1:

Input:
["a","a","b","b","c","c","c"]

Output:
Return 6, and the first 6 characters of the input array should be:
["a","2","b","2","c","3"]

Explanation:
"aa" is replaced by "a2". "bb" is replaced by "b2". "ccc" is replaced by "c3".

Example 2:

Input:
["a"]

Output:
Return 1, and the first 1 characters of the input array should be: ["a"]

Explanation:
Nothing is replaced.

Example 3:

Input:
["a","b","b","b","b","b","b","b","b","b","b","b","b"]

Output:
Return 4, and the first 4 characters of the input array should be: ["a","b","1","2"].

Explanation:
Since the character "a" does not repeat, it is not compressed.
"bbbbbbbbbbbb" is replaced by "b12".
Notice each digit has it's own entry in the array.

Note:

1. All characters have an ASCII value in [35, 126].
2. 1 <= len(chars) <= 1000.

447. Number of Boomerangs

DescriptionHintsSubmissionsDiscussSolution

Given *n* points in the plane that are all pairwise distinct, a "boomerang" is a tuple of points (i, j, k) such that the distance between i and j equals the distance between i and k (**the order of the tuple matters**).

Find the number of boomerangs. You may assume that *n* will be at most **500** and coordinates of points are all in the range **[-10000, 10000]** (inclusive).

Example:

Input:

[[0,0],[1,0],[2,0]]

Output:
2

Explanation:
The two boomerangs are **[[1,0],[0,0],[2,0]]** and **[[1,0],[2,0],[0,0]]**

448. Find All Numbers Disappeared in an Array

DescriptionHintsSubmissionsDiscussSolution

Given an array of integers where 1 ≤ a[i] ≤ *n* (*n* = size of array), some elements appear twice and others appear once.

Find all the elements of [1, *n*] inclusive that do not appear in this array.

Could you do it without extra space and in O(*n*) runtime? You may assume the returned list does not count as extra space.

Example:

Input:
[4,3,2,7,8,2,3,1]

Output:
[5,6]

453. Minimum Moves to Equal Array Elements

DescriptionHintsSubmissionsDiscussSolution

Given a **non-empty** integer array of size *n*, find the minimum number of moves required to make all array elements equal, where a move is incrementing *n* - 1 elements by 1.

Example:

Input:
[1,2,3]

Output:
3

Explanation:
Only three moves are needed (remember each move increments two elements):

[1,2,3] => [2,3,3] => [3,4,3] => [4,4,4]

---

455. Assign Cookies

DescriptionHintsSubmissionsDiscussSolution

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie. Each child i has a greed factor g_i, which is the minimum size of a cookie that the child will be content with; and each cookie j has a size s_j. If $s_j >= g_i$, we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

Note:

You may assume the greed factor is always positive.

You cannot assign more than one cookie to one child.

Example 1:

Input: [1,2,3], [1,1]

Output: 1

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.
You need to output 1.

Example 2:

Input: [1,2], [1,2,3]

Output: 2

Explanation: You have 2 children and 3 cookies. The greed factors of 2 children are 1, 2.
You have 3 cookies and their sizes are big enough to gratify all of the children,
You need to output 2.

458. Poor Pigs

DescriptionHintsSubmissionsDiscussSolution

There are 1000 buckets, one and only one of them contains poison, the rest are filled with water. They all look the same. If a pig drinks that poison it will die within 15 minutes. What is the minimum amount of pigs you need to figure out which bucket contains the poison within one hour.

Answer this question, and write an algorithm for the follow-up general case.

Follow-up:

If there are n buckets and a pig drinking poison will die within m minutes, how many pigs (x) you need to figure out the "poison" bucket within p minutes? There is exact one bucket with poison.

459. Repeated Substring Pattern

DescriptionHintsSubmissionsDiscussSolution

Given a non-empty string check if it can be constructed by taking a substring of it and appending multiple copies of the substring together. You may assume the given string consists of lowercase English letters only and its length will not exceed 10000.

Example 1:

Input: "abab"
Output: True
Explanation: It's the substring "ab" twice.

Example 2:

Input: "aba"
Output: False

Example 3:

Input: "abcabcabcabc"
Output: True
Explanation: It's the substring "abc" four times. (And the substring "abcabc" twice.)

461. Hamming Distance

DescriptionHintsSubmissionsDiscussSolution

The Hamming distance between two integers is the number of positions at which the corresponding bits are different.

Given two integers x and y, calculate the Hamming distance.

Note:

$0 \leqslant x, y < 2^{31}$.

Example:

Input: x = 1, y = 4

Output: 2

Explanation:
1 (0 0 0 1)
4 (0 1 0 0)
↑ ↑

The above arrows point to positions where the corresponding bits are different.

463. Island Perimeter

DescriptionHintsSubmissionsDiscussSolution

You are given a map in form of a two-dimensional integer grid where 1 represents land and 0 represents water.

Grid cells are connected horizontally/vertically (not diagonally). The grid is completely surrounded by water, and there is exactly one island (i.e., one or more connected land cells).

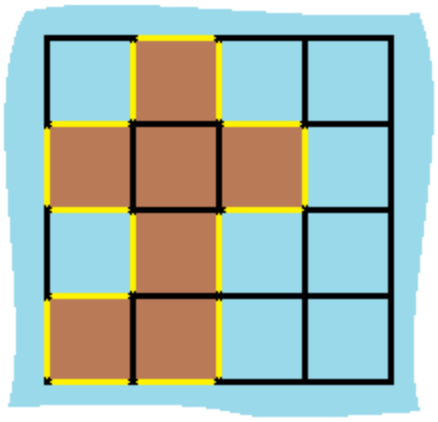
The island doesn't have "lakes" (water inside that isn't connected to the water around the island). One cell is a square with side length 1. The grid is rectangular, width and height don't exceed 100. Determine the perimeter of the island.

Example:

Input:
[[0,1,0,0],
[1,1,1,0],
[0,1,0,0],
[1,1,0,0]]

Output: 16

Explanation: The perimeter is the 16 yellow stripes in the image below:



475. Heaters

DescriptionHintsSubmissionsDiscussSolution

Winter is coming! Your first job during the contest is to design a standard heater with fixed warm radius to warm all the houses.

Now, you are given positions of houses and heaters on a horizontal line, find out minimum radius of heaters so that all houses could be covered by those heaters.

So, your input will be the positions of houses and heaters seperately, and your expected output will be the minimum radius standard of heaters.

Note:

1. Numbers of houses and heaters you are given are non-negative and will not exceed 25000.
2. Positions of houses and heaters you are given are non-negative and will not exceed 10^9.
3. As long as a house is in the heaters' warm radius range, it can be warmed.
4. All the heaters follow your radius standard and the warm radius will the same.

Example 1:

Input: [1,2,3],[2]
Output: 1
Explanation: The only heater was placed in the position 2, and if we use the radius 1 standard, then all the houses can be warmed.

Example 2:

Input: [1,2,3,4],[1,4]
Output: 1
Explanation: The two heater was placed in the position 1 and 4. We need to use radius 1 standard, then all the houses can be warmed.

---

476. Number Complement

DescriptionHintsSubmissionsDiscussSolution

Given a positive integer, output its complement number. The complement strategy is to flip the bits of its binary representation.

Note:

1. The given integer is guaranteed to fit within the range of a 32-bit signed integer.
2. You could assume no leading zero bit in the integer's binary representation.

Example 1:

Input: 5
Output: 2
Explanation: The binary representation of 5 is 101 (no leading zero bits), and its complement is 010. So you need to output 2.

Example 2:

Input: 1
Output: 0
Explanation: The binary representation of 1 is 1 (no leading zero bits), and its complement is 0. So you need to output 0.

---

479. Largest Palindrome Product

DescriptionHintsSubmissionsDiscussSolution

Find the largest palindrome made from the product of two n-digit numbers.

Since the result could be very large, you should return the largest palindrome mod 1337.

Example:

Input: 2

Output: 987

Explanation: 99 x 91 = 9009, 9009 % 1337 = 987

Note:

The range of n is [1,8].

482. License Key Formatting

DescriptionHintsSubmissionsDiscussSolution

You are given a license key represented as a string S which consists only alphanumeric character and dashes. The string is separated into N+1 groups by N dashes.

Given a number K, we would want to reformat the strings such that each group contains *exactly* K characters, except for the first group which could be shorter than K, but still must contain at least one character. Furthermore, there must be a dash inserted between two groups and all lowercase letters should be converted to uppercase.

Given a non-empty string S and a number K, format the string according to the rules described above.

Example 1:

Input: S = "5F3Z-2e-9-w", K = 4

Output: "5F3Z-2E9W"

Explanation: The string S has been split into two parts, each part has 4 characters.
Note that the two extra dashes are not needed and can be removed.

Example 2:

Input: S = "2-5g-3-J", K = 2

Output: "2-5G-3J"

Explanation: The string S has been split into three parts, each part has 2 characters except the first part as it could be shorter as mentioned above.

Note:

1. The length of string S will not exceed 12,000, and K is a positive integer.
2. String S consists only of alphanumerical characters (a-z and/or A-Z and/or 0-9) and dashes(-).
3. String S is non-empty.

485. Max Consecutive Ones

DescriptionHintsSubmissionsDiscussSolution

Given a binary array, find the maximum number of consecutive 1s in this array.

Example 1:

Input: [1,1,0,1,1,1]
Output: 3
Explanation: The first two digits or the last three digits are consecutive 1s.
    The maximum number of consecutive 1s is 3.

Note:

- The input array will only contain 0 and 1.
- The length of input array is a positive integer and will not exceed 10,000

492. Construct the Rectangle

DescriptionHintsSubmissionsDiscussSolution

For a web developer, it is very important to know how to design a web page's size. So, given a specific rectangular web page's area, your job by now is to design a rectangular web page, whose length L and width W satisfy the following requirements:

1. The area of the rectangular web page you designed must equal to the given target area.

2. The width W should not be larger than the length L, which means L >= W.

3. The difference between length L and width W should be as small as possible.
You need to output the length L and the width W of the web page you designed in sequence.

Example:

Input: 4
Output: [2, 2]
Explanation: The target area is 4, and all the possible ways to construct it are [1,4], [2,2], [4,1].
But according to requirement 2, [1,4] is illegal; according to requirement 3, [4,1] is not optimal compared to [2,2]. So the length L is 2, and the width W is 2.

Note:

1. The given area won't exceed 10,000,000 and is a positive integer
2. The web page's width and length you designed must be positive integers.

496. Next Greater Element I

DescriptionHintsSubmissionsDiscussSolution

You are given two arrays **(without duplicates)** nums1 and nums2 where nums1's elements are subset of nums2. Find all the next greater numbers for nums1's elements in the corresponding places of nums2.

The Next Greater Number of a number **x** in nums1 is the first greater number to its right in nums2. If it does not exist, output -1 for this number.

Example 1:

Input: **nums1** = [4,1,2], **nums2** = [1,3,4,2].
Output: [-1,3,-1]
Explanation:
    For number 4 in the first array, you cannot find the next greater number for it in the second array, so output -1.
    For number 1 in the first array, the next greater number for it in the second array is 3.
    For number 2 in the first array, there is no next greater number for it in the second array, so output -1.

Example 2:

Input: **nums1** = [2,4], **nums2** = [1,2,3,4].
Output: [3,-1]
Explanation:
    For number 2 in the first array, the next greater number for it in the second array is 3.
    For number 4 in the first array, there is no next greater number for it in the second array, so output -1.

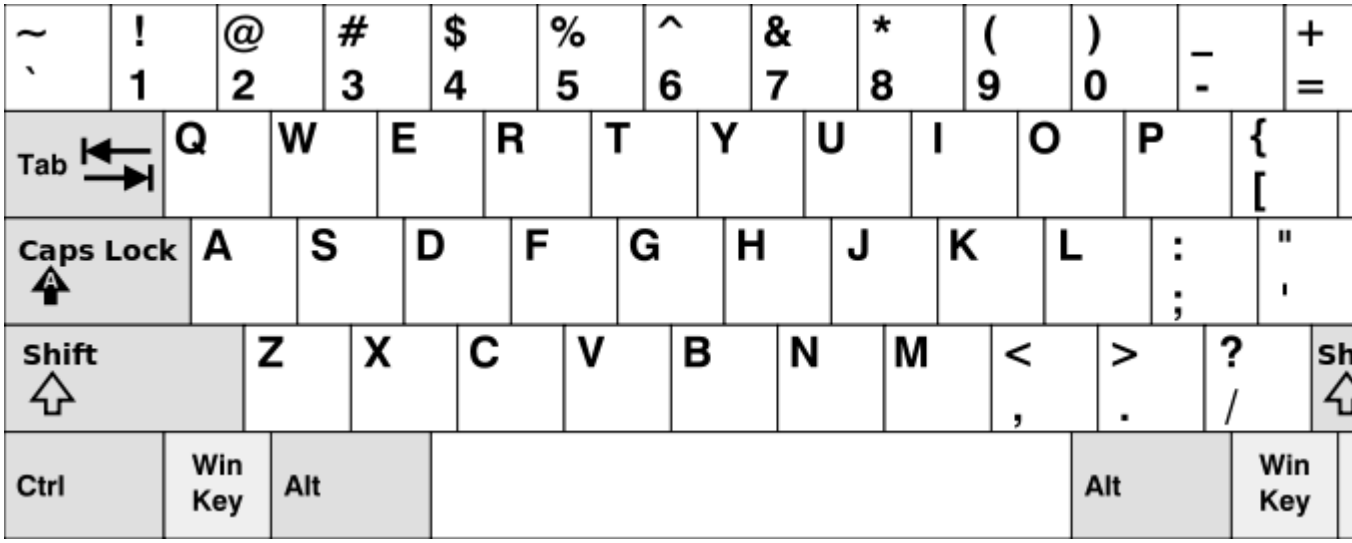
Note:

1. All elements in nums1 and nums2 are unique.
2. The length of both nums1 and nums2 would not exceed 1000.

500. Keyboard Row

DescriptionHintsSubmissionsDiscussSolution

Given a List of words, return the words that can be typed using letters of **alphabet** on only one row's of American keyboard like the image below.



Example:

```
Input: ["Hello", "Alaska", "Dad", "Peace"]
Output: ["Alaska", "Dad"]
```

Note:

1. You may use one character in the keyboard more than once.
2. You may assume the input string will only contain letters of alphabet.

---

501. Find Mode in Binary Search Tree

DescriptionHintsSubmissionsDiscussSolution

Given a binary search tree (BST) with duplicates, find all the mode(s) (the most frequently occurred element) in the given BST.

Assume a BST is defined as follows:

- The left subtree of a node contains only nodes with keys **less than or equal to** the node's key.
- The right subtree of a node contains only nodes with keys **greater than or equal to** the node's key.

- Both the left and right subtrees must also be binary search trees.

For example:

Given BST [1,null,2,2],

```
   1
    \
     2
    /
   2
```

return [2].

Note: If a tree has more than one mode, you can return them in any order.

Follow up: Could you do that without using any extra space? (Assume that the implicit stack space incurred due to recursion does not count).

---

504. Base 7

DescriptionHintsSubmissionsDiscussSolution

Given an integer, return its base 7 string representation.

Example 1:

Input: 100
Output: "202"

Example 2:

Input: -7
Output: "-10"

Note: The input will be in range of [-1e7, 1e7].

506. Relative Ranks

DescriptionHintsSubmissionsDiscussSolution

Given scores of N athletes, find their relative ranks and the people with the top three highest scores, who will be awarded medals: "Gold Medal", "Silver Medal" and "Bronze Medal".

Example 1:

Input: [5, 4, 3, 2, 1]
Output: ["Gold Medal", "Silver Medal", "Bronze Medal", "4", "5"]
Explanation: The first three athletes got the top three highest scores, so they got "Gold Medal", "Silver Medal" and "Bronze Medal". For the left two athletes, you just need to output their relative ranks according to their scores.

Note:

1. N is a positive integer and won't exceed 10,000.
2. All the scores of athletes are guaranteed to be unique.

---

507. Perfect Number

DescriptionHintsSubmissionsDiscussSolution

We define the Perfect Number is a **positive** integer that is equal to the sum of all its **positive** divisors except itself.

Now, given an **integer** n, write a function that returns true when it is a perfect number and false when it is not.

Example:

Input: 28
Output: True

Explanation: 28 = 1 + 2 + 4 + 7 + 14

Note: The input number **n** will not exceed 100,000,000. (1e8)

520. Detect Capital

DescriptionHintsSubmissionsDiscussSolution

Given a word, you need to judge whether the usage of capitals in it is right or not.

We define the usage of capitals in a word to be right when one of the following cases holds:

1. All letters in this word are capitals, like "USA".
2. All letters in this word are not capitals, like "leetcode".
3. Only the first letter in this word is capital if it has more than one letter, like "Google".

Otherwise, we define that this word doesn't use capitals in a right way.

Example 1:

Input: "USA"
Output: True

Example 2:

Input: "FlaG"
Output: False

Note: The input will be a non-empty word consisting of uppercase and lowercase latin letters.

521. Longest Uncommon Subsequence I

DescriptionHintsSubmissionsDiscussSolution

Given a group of two strings, you need to find the longest uncommon subsequence of this group of two strings. The longest uncommon subsequence is defined as the longest subsequence of one of these

strings and this subsequence should not be **any** subsequence of the other strings.

A **subsequence** is a sequence that can be derived from one sequence by deleting some characters without changing the order of the remaining elements. Trivially, any string is a subsequence of itself and an empty string is a subsequence of any string.

The input will be two strings, and the output needs to be the length of the longest uncommon subsequence. If the longest uncommon subsequence doesn't exist, return -1.

Example 1:

Input: "aba", "cdc"
Output: 3
Explanation: The longest uncommon subsequence is "aba" (or "cdc"), because "aba" is a subsequence of "aba",
but not a subsequence of any other strings in the group of two strings.

Note:

1. Both strings' lengths will not exceed 100.
2. Only letters from a ~ z will appear in input strings.

530. Minimum Absolute Difference in BST

DescriptionHintsSubmissionsDiscussSolution

Given a binary search tree with non-negative values, find the minimum absolute difference between values of any two nodes.

Example:

Input:

```
   1
    \
     3
    /
   2
```

Output:
1

Explanation:
The minimum absolute difference is 1, which is the difference between 2 and 1 (or between 2 and 3).

Note: There are at least two nodes in this BST.

532. K-diff Pairs in an Array

DescriptionHintsSubmissionsDiscussSolution

Given an array of integers and an integer **k**, you need to find the number of **unique** k-diff pairs in the array. Here a **k-diff** pair is defined as an integer pair (i, j), where **i** and **j** are both numbers in the array and their absolute difference is **k**.

Example 1:

Input: [3, 1, 4, 1, 5], k = 2
Output: 2
Explanation: There are two 2-diff pairs in the array, (1, 3) and (3, 5).
Although we have two 1s in the input, we should only return the number of **unique** pairs.

Example 2:

Input:[1, 2, 3, 4, 5], k = 1
Output: 4
Explanation: There are four 1-diff pairs in the array, (1, 2), (2, 3), (3, 4) and (4, 5).

Example 3:

Input: [1, 3, 1, 5, 4], k = 0
Output: 1
Explanation: There is one 0-diff pair in the array, (1, 1).

Note:

1. The pairs (i, j) and (j, i) count as the same pair.
2. The length of the array won't exceed 10,000.
3. All the integers in the given input belong to the range: [-1e7, 1e7].

538. Convert BST to Greater Tree

DescriptionHintsSubmissionsDiscussSolution

Given a Binary Search Tree (BST), convert it to a Greater Tree such that every key of the original BST is changed to the original key plus sum of all keys greater than the original key in BST.

Example:

```
Input: The root of a Binary Search Tree like this:
              5
            /   \
           2     13

Output: The root of a Greater Tree like this:
             18
            /   \
          20     13
```

---

541. Reverse String II

DescriptionHintsSubmissionsDiscussSolution

Given a string and an integer k, you need to reverse the first k characters for every 2k characters counting from the start of the string. If there are less than k characters left, reverse all of them. If there are less than 2k but greater than or equal to k characters, then reverse the first k characters and left the other as original.

Example:

```
Input: s = "abcdefg", k = 2
Output: "bacdfeg"
```

Restrictions:

1. The string consists of lower English letters only.

2. Length of the given string and k will in the range [1, 10000]

543. Diameter of Binary Tree

DescriptionHintsSubmissionsDiscussSolution

Given a binary tree, you need to compute the length of the diameter of the tree. The diameter of a binary tree is the length of the **longest** path between any two nodes in a tree. This path may or may not pass through the root.

Example:

Given a binary tree

```
          1
         / \
        2   3
       / \
      4   5
```

Return **3**, which is the length of the path [4,2,1,3] or [5,2,1,3].

Note: The length of path between two nodes is represented by the number of edges between them.

551. Student Attendance Record I

DescriptionHintsSubmissionsDiscussSolution

You are given a string representing an attendance record for a student. The record only contains the following three characters:

1. **'A'** : Absent.
2. **'L'** : Late.
3. **'P'** : Present.

A student could be rewarded if his attendance record doesn't contain **more than one 'A' (absent)** or **more than two continuous 'L' (late).**

You need to return whether the student could be rewarded according to his attendance record.

Example 1:

Input: "PPALLP"
Output: True

Example 2:

Input: "PPALLL"
Output: False

557. Reverse Words in a String III

DescriptionHintsSubmissionsDiscussSolution

Given a string, you need to reverse the order of characters in each word within a sentence while still preserving whitespace and initial word order.

Example 1:

Input: "Let's take LeetCode contest"
Output: "s'teL ekat edoCteeL tsetnoc"

Note: In the string, each word is separated by single space and there will not be any extra space in the string.

558. Quad Tree Intersection

DescriptionHintsSubmissionsDiscussSolution

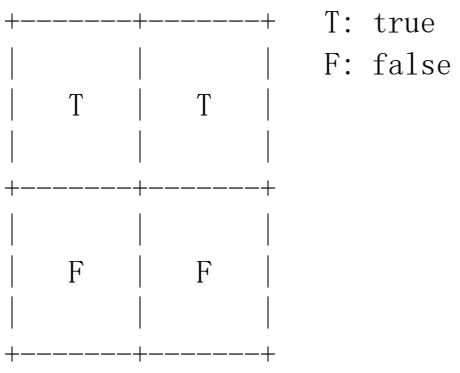
A quadtree is a tree data in which each internal node has exactly four children: topLeft, topRight, bottomLeft and bottomRight. Quad trees are often used to partition a two-dimensional space by recursively subdividing it into four quadrants or regions.

We want to store True/False information in our quad tree. The quad tree is used to represent a N * N boolean grid. For each node, it

will be subdivided into four children nodes **until the values in the region it represents are all the same.** Each node has another two boolean attributes : isLeaf and val. isLeaf is true if and only if the node is a leaf node. The val attribute for a leaf node contains the value of the region it represents.

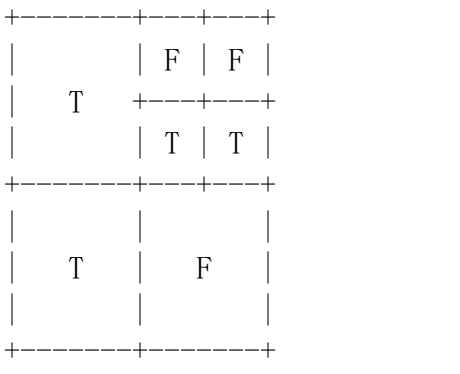
For example, below are two quad trees A and B:

A:



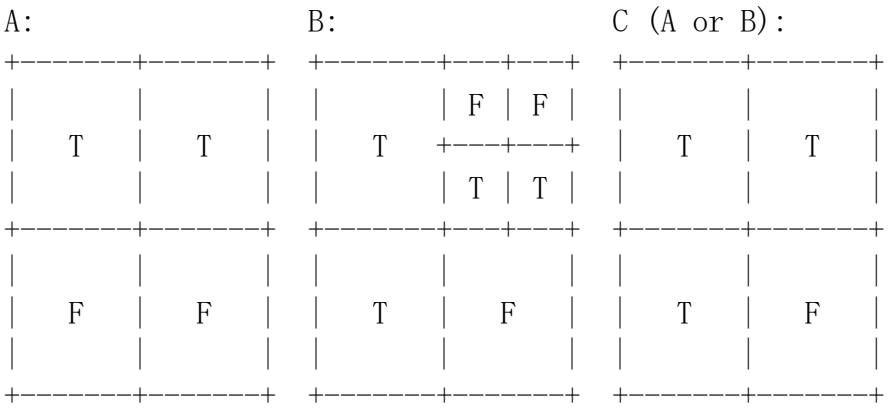
topLeft: T
topRight: T
bottomLeft: F
bottomRight: F

B:



topLeft: T
topRight:
    topLeft: F
    topRight: F
    bottomLeft: T
    bottomRight: T
bottomLeft: T
bottomRight: F

Your task is to implement a function that will take two quadtrees and return a quadtree that represents the logical OR (or union) of the two trees.



Note:

1. Both A and B represent grids of size N * N.
2. N is guaranteed to be a power of 2.
3. If you want to know more about the quad tree, you can refer to its wiki.
4. The logic OR operation is defined as this: "A or B" is true if A is true, or if B is true, or if both A and B are true.

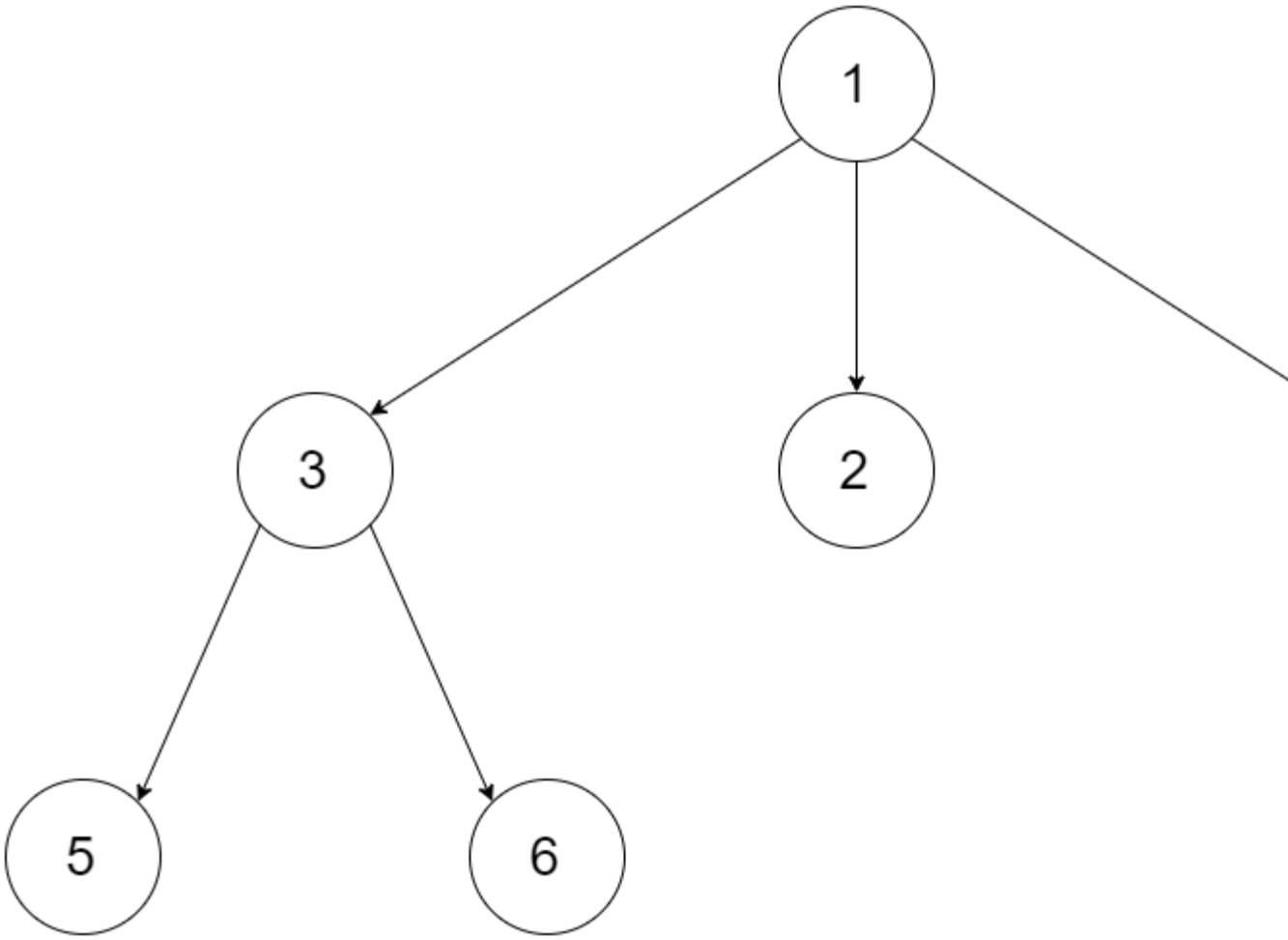
559. Maximum Depth of N-ary Tree

DescriptionHintsSubmissionsDiscussSolution

Given a n-ary tree, find its maximum depth.

The maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

For example, given a 3-ary tree:



We should return its max depth, which is 3.

Note:

1. The depth of the tree is at most 1000.
2. The total number of nodes is at most 5000.

561. Array Partition I

DescriptionHintsSubmissionsDiscussSolution

Given an array of **2n** integers, your task is to group these integers into **n** pairs of integer, say (a_1, b_1), (a_2, b_2), ..., (a_n, b_n) which makes sum of $\min(a_i, b_i)$ for all i from 1 to n as large as possible.

Example 1:

Input: [1,4,3,2]

Output: 4
Explanation: n is 2, and the maximum sum of pairs is 4 = min(1, 2) + min(3, 4).

Note:

1. **n** is a positive integer, which is in the range of [1, 10000].
2. All the integers in the array will be in the range of [-10000, 10000].

---

563. Binary Tree Tilt

DescriptionHintsSubmissionsDiscussSolution

Given a binary tree, return the tilt of the **whole tree**.

The tilt of a **tree node** is defined as the **absolute difference** between the sum of all left subtree node values and the sum of all right subtree node values. Null node has tilt 0.

The tilt of the **whole tree** is defined as the sum of all nodes' tilt.

Example:

Input:

```
         1
       /   \
      2     3
```

Output: 1
Explanation:
Tilt of node 2 : 0
Tilt of node 3 : 0
Tilt of node 1 : |2-3| = 1
Tilt of binary tree : 0 + 0 + 1 = 1

Note:

1. The sum of node values in any subtree won't exceed the range of 32-bit integer.
2. All the tilt values won't exceed the range of 32-bit integer.

566. Reshape the Matrix

DescriptionHintsSubmissionsDiscussSolution

In MATLAB, there is a very useful function called 'reshape', which can reshape a matrix into a new one with different size but keep its original data.

You're given a matrix represented by a two-dimensional array, and two **positive** integers **r** and **c** representing the **row** number and **column** number of the wanted reshaped matrix, respectively.

The reshaped matrix need to be filled with all the elements of the original matrix in the same **row-traversing** order as they were.

If the 'reshape' operation with given parameters is possible and legal, output the new reshaped matrix; Otherwise, output the original matrix.

Example 1:

Input:
nums =
[[1,2],
 [3,4]]
r = 1, c = 4
Output:
[[1,2,3,4]]
Explanation:
The **row-traversing** of nums is [1,2,3,4]. The new reshaped matrix is a 1 * 4 matrix, fill it row by row by using the previous list.

Example 2:

Input:
nums =
[[1,2],
 [3,4]]

r = 2, c = 4
Output:
[[1,2],
 [3,4]]
Explanation:
There is no way to reshape a 2 * 2 matrix to a 2 * 4 matrix. So output the original matrix.

Note:

1. The height and width of the given matrix is in range [1, 100].
2. The given r and c are all positive.

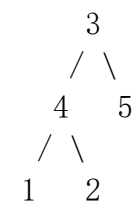
572. Subtree of Another Tree

DescriptionHintsSubmissionsDiscussSolution

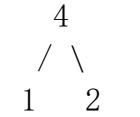
Given two non-empty binary trees **s** and **t**, check whether tree **t** has exactly the same structure and node values with a subtree of **s**. A subtree of **s** is a tree consists of a node in **s** and all of this node's descendants. The tree **s** could also be considered as a subtree of itself.

Example 1:

Given tree s:



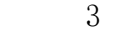
Given tree t:

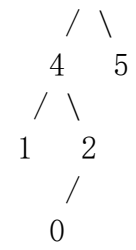


Return **true**, because t has the same structure and node values with a subtree of s.

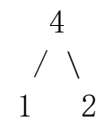
Example 2:

Given tree s:





Given tree t:



Return **false**.

575. Distribute Candies

DescriptionHintsSubmissionsDiscussSolution

Given an integer array with **even** length, where different numbers in this array represent different **kinds** of candies. Each number means one candy of the corresponding kind. You need to distribute these candies **equally** in number to brother and sister. Return the maximum number of **kinds** of candies the sister could gain.

Example 1:

```
Input: candies = [1,1,2,2,3,3]
Output: 3
Explanation:
There are three different kinds of candies (1, 2 and 3), and two
candies for each kind.
Optimal distribution: The sister has candies [1,2,3] and the brother
has candies [1,2,3], too.
The sister has three different kinds of candies.
```

Example 2:

```
Input: candies = [1,1,2,3]
Output: 2
Explanation: For example, the sister has candies [2,3] and the
brother has candies [1,1].
The sister has two different kinds of candies, the brother has only
one kind of candies.
```

Note:

1. The length of the given array is in range [2, 10,000], and will be even.
2. The number in given array is in range [-100,000, 100,000].

---

581. Shortest Unsorted Continuous Subarray

DescriptionHintsSubmissionsDiscussSolution

Given an integer array, you need to find one **continuous subarray** that if you only sort this subarray in ascending order, then the whole array will be sorted in ascending order, too.

You need to find the **shortest** such subarray and output its length.

Example 1:

Input: [2, 6, 4, 8, 10, 9, 15]
Output: 5
Explanation: You need to sort [6, 4, 8, 10, 9] in ascending order to make the whole array sorted in ascending order.

Note:

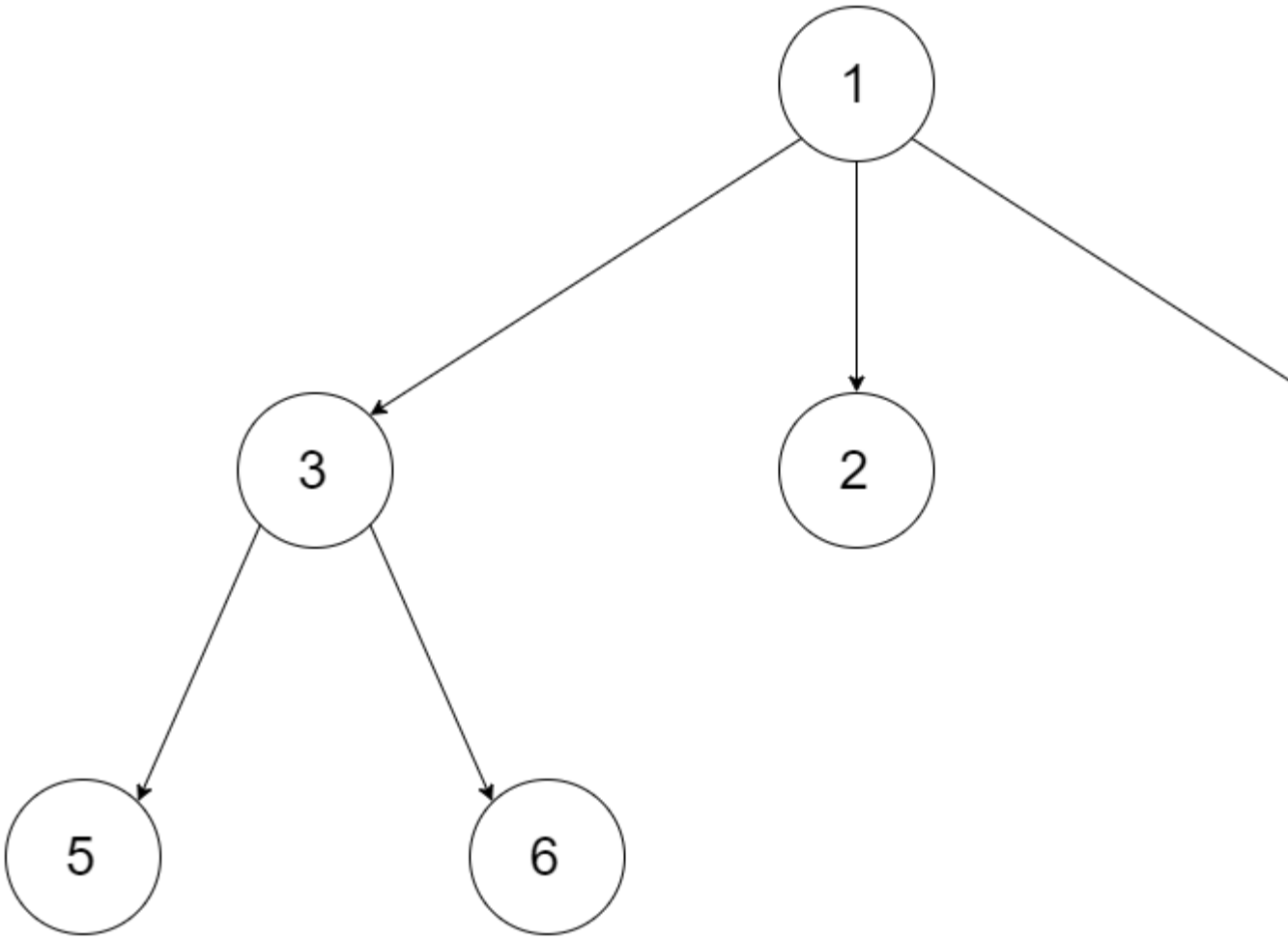
1. Then length of the input array is in range [1, 10,000].
2. The input array may contain duplicates, so ascending order here means **<=**.

589. N-ary Tree Preorder Traversal

DescriptionHintsSubmissionsDiscussSolution

Given an n-ary tree, return the *preorder* traversal of its nodes' values.

For example, given a 3-ary tree:



Return its preorder traversal as: [1,3,5,6,2,4].

Note:

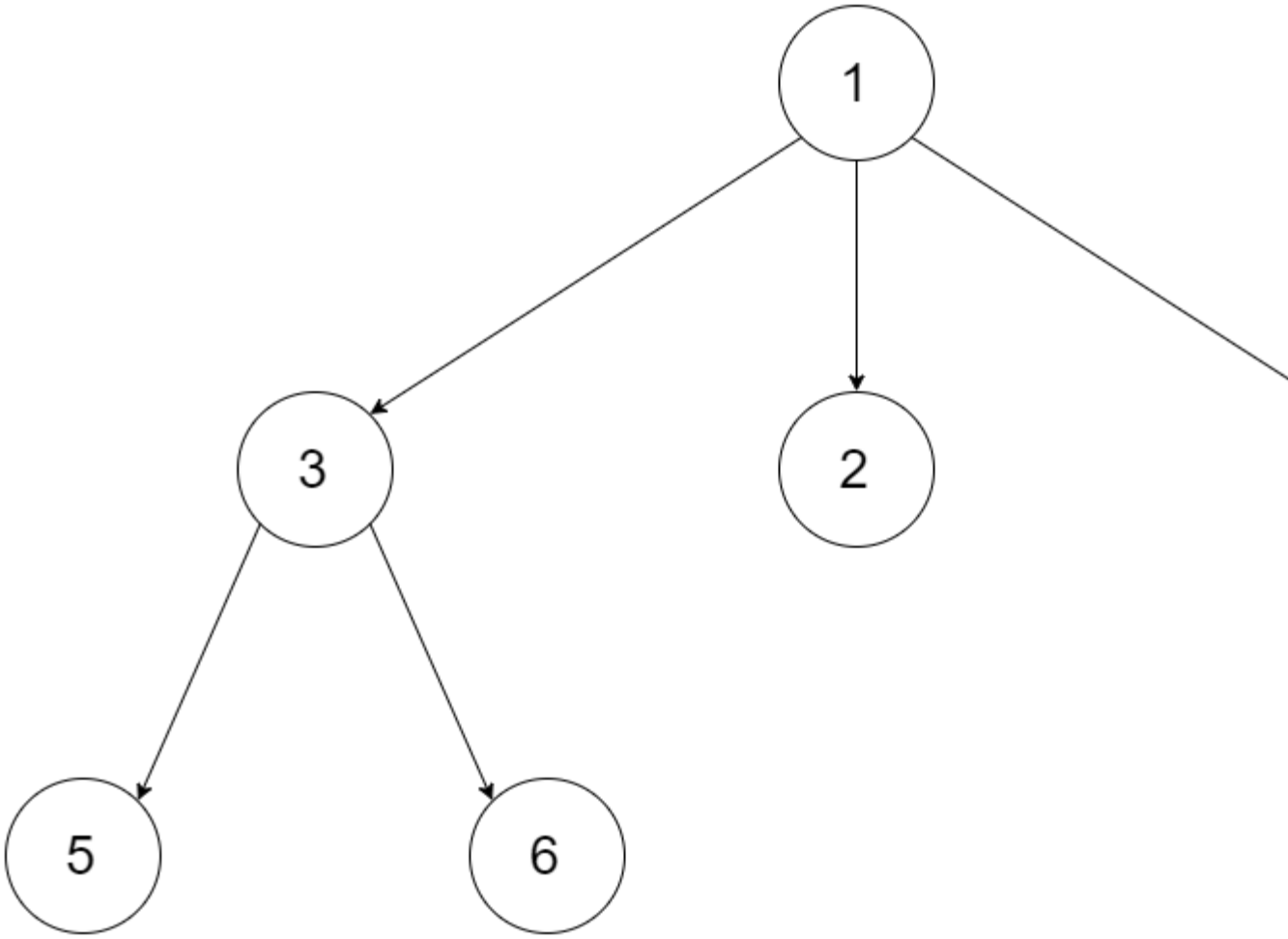
Recursive solution is trivial, could you do it iteratively?

590. N-ary Tree Postorder Traversal

DescriptionHintsSubmissionsDiscussSolution

Given an n-ary tree, return the *postorder* traversal of its nodes' values.

For example, given a 3-ary tree:



Return its postorder traversal as: [5,6,3,2,4,1].

Note:

Recursive solution is trivial, could you do it iteratively?

594. Longest Harmonious Subsequence

DescriptionHintsSubmissionsDiscussSolution

We define a harmonious array is an array where the difference between its maximum value and its minimum value is **exactly** 1.

Now, given an integer array, you need to find the length of its longest harmonious subsequence among all its possible subsequences.

Example 1:

Input: [1,3,2,2,5,2,3,7]
Output: 5
Explanation: The longest harmonious subsequence is [3,2,2,2,3].

Note: The length of the input array will not exceed 20,000.

595. Big Countries

DescriptionHintsSubmissionsDiscussSolution

There is a table World

```
+-----------------+------------+------------+--------------+---------
------+
| name            | continent  | area       | population   | gdp
|
+-----------------+------------+------------+--------------+---------
------+
| Afghanistan     | Asia       | 652230     | 25500100     | 20343000
|
| Albania         | Europe     | 28748      | 2831741      | 12960000
|
| Algeria         | Africa     | 2381741    | 37100000     |
188681000     |
| Andorra         | Europe     | 468        | 78115        | 3712000
|
| Angola          | Africa     | 1246700    | 20609294     |
100990000     |
+-----------------+------------+------------+--------------+---------
------+
```

A country is big if it has an area of bigger than 3 million square km or a population of more than 25 million.

Write a SQL solution to output big countries' name, population and area.

For example, according to the above table, we should output:

```
+-------------+------------+-------------+
| name        | population | area        |
+-------------+------------+-------------+
| Afghanistan | 25500100   | 652230      |
| Algeria     | 37100000   | 2381741     |
+-------------+------------+-------------+
```

596. Classes More Than 5 Students

DescriptionHintsSubmissionsDiscussSolution

There is a table courses with columns: **student** and **class**

Please list out all classes which have more than or equal to 5 students.

For example, the table:

```
+---------+------------+
| student | class      |
+---------+------------+
| A       | Math       |
| B       | English    |
| C       | Math       |
| D       | Biology    |
| E       | Math       |
| F       | Computer   |
| G       | Math       |
| H       | Math       |
| I       | Math       |
+---------+------------+
```

Should output:

```
+---------+
| class   |
+---------+
| Math    |
```

+---------+

Note:

The students should not be counted duplicate in each course.

598. Range Addition II

DescriptionHintsSubmissionsDiscussSolution

Given an m * n matrix **M** initialized with all **0**'s and several update operations.

Operations are represented by a 2D array, and each operation is represented by an array with two **positive** integers **a** and **b**, which means **M[i][j]** should be **added by one** for all **0 <= i < a** and **0 <= j < b**.

You need to count and return the number of maximum integers in the matrix after performing all the operations.

Example 1:

Input:
m = 3, n = 3
operations = [[2,2],[3,3]]
Output: 4
Explanation:
Initially, M =
[[0, 0, 0],
 [0, 0, 0],
 [0, 0, 0]]

After performing [2,2], M =
[[1, 1, 0],
 [1, 1, 0],
 [0, 0, 0]]

After performing [3,3], M =
[[2, 2, 1],
 [2, 2, 1],
 [1, 1, 1]]

So the maximum integer in M is 2, and there are four of it in M. So return 4.

Note:

1. The range of m and n is [1,40000].
2. The range of a is [1,m], and the range of b is [1,n].
3. The range of operations size won't exceed 10,000.

599. Minimum Index Sum of Two Lists

DescriptionHintsSubmissionsDiscussSolution

Suppose Andy and Doris want to choose a restaurant for dinner, and they both have a list of favorite restaurants represented by strings.

You need to help them find out their **common interest** with the **least list index sum**. If there is a choice tie between answers, output all of them with no order requirement. You could assume there always exists an answer.

Example 1:

Input:
["Shogun", "Tapioca Express", "Burger King", "KFC"]
["Piatti", "The Grill at Torrey Pines", "Hungry Hunter Steakhouse", "Shogun"]
Output: ["Shogun"]
Explanation: The only restaurant they both like is "Shogun".

Example 2:

Input:
["Shogun", "Tapioca Express", "Burger King", "KFC"]
["KFC", "Shogun", "Burger King"]
Output: ["Shogun"]
Explanation: The restaurant they both like and have the least index sum is "Shogun" with index sum 1 (0+1).

Note:

1. The length of both lists will be in the range of [1, 1000].

2. The length of strings in both lists will be in the range of [1, 30].
3. The index is starting from 0 to the list length minus 1.
4. No duplicates in both lists.

605. Can Place Flowers

DescriptionHintsSubmissionsDiscussSolution

Suppose you have a long flowerbed in which some of the plots are planted and some are not. However, flowers cannot be planted in adjacent plots - they would compete for water and both would die.

Given a flowerbed (represented as an array containing 0 and 1, where 0 means empty and 1 means not empty), and a number **n**, return if **n** new flowers can be planted in it without violating the no-adjacent-flowers rule.

Example 1:

```
Input: flowerbed = [1,0,0,0,1], n = 1
Output: True
```

Example 2:

```
Input: flowerbed = [1,0,0,0,1], n = 2
Output: False
```

Note:

1. The input array won't violate no-adjacent-flowers rule.
2. The input array size is in the range of [1, 20000].
3. **n** is a non-negative integer which won't exceed the input array size.

606. Construct String from Binary Tree

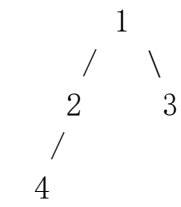
DescriptionHintsSubmissionsDiscussSolution

You need to construct a string consists of parenthesis and integers from a binary tree with the preorder traversing way.

The null node needs to be represented by empty parenthesis pair "()". And you need to omit all the empty parenthesis pairs that don't affect the one-to-one mapping relationship between the string and the original binary tree.

Example 1:

Input: Binary tree: [1,2,3,4]

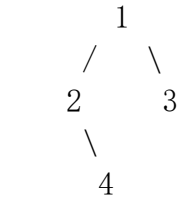


Output: "1(2(4))(3)"

Explanation: Originallay it needs to be "1(2(4())(3()())", but you need to omit all the unnecessary empty parenthesis pairs. And it will be "1(2(4))(3)".

Example 2:

Input: Binary tree: [1,2,3,null,4]



Output: "1(2()(4))(3)"

Explanation: Almost the same as the first example, except we can't omit the first parenthesis pair to break the one-to-one mapping relationship between the input and the output.

---

617. Merge Two Binary Trees

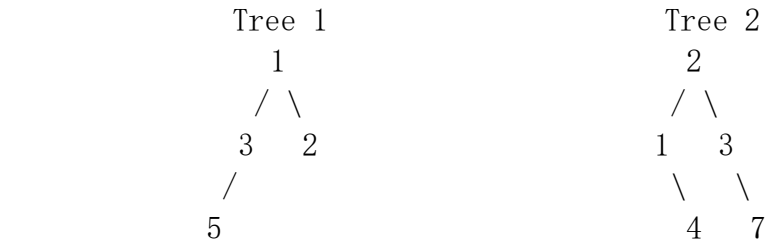
DescriptionHintsSubmissionsDiscussSolution

Given two binary trees and imagine that when you put one of them to cover the other, some nodes of the two trees are overlapped while the others are not.

You need to merge them into a new binary tree. The merge rule is that if two nodes overlap, then sum node values up as the new value of the merged node. Otherwise, the NOT null node will be used as the node of new tree.

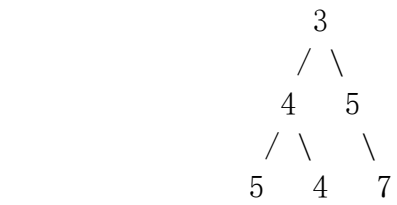
Example 1:

Input:



Output:

Merged tree:



Note: The merging process must start from the root nodes of both trees.

620. Not Boring Movies

DescriptionHintsSubmissionsDiscussSolution

X city opened a new cinema, many people would like to go to this cinema. The cinema also gives out a poster indicating the movies' ratings and descriptions.

Please write a SQL query to output movies with an odd numbered ID and a description that is not 'boring'. Order the result by rating.

For example, table cinema:

```
+---------+-----------+--------------+-----------+
```

id	movie	description	rating
1	War	great 3D	8.9
2	Science	fiction	8.5
3	irish	boring	6.2
4	Ice song	Fantacy	8.6
5	House card	Interesting	9.1

For the example above, the output should be:

id	movie	description	rating
5	House card	Interesting	9.1
1	War	great 3D	8.9

627. Swap Salary

DescriptionHintsSubmissionsDiscussSolution

Given a table `salary`, such as the one below, that has m=male and f=female values. Swap all f and m values (i.e., change all f values to m and vice versa) with a single update query and no intermediate temp table.

For example:

id	name	sex	salary
1	A	m	2500
2	B	f	1500
3	C	m	5500
4	D	f	500

After running your query, the above salary table should have the following rows:

id	name	sex	salary
1	A	f	2500
2	B	m	1500
3	C	f	5500
4	D	m	500

628. Maximum Product of Three Numbers

DescriptionHintsSubmissionsDiscussSolution

Given an integer array, find three numbers whose product is maximum and output the maximum product.

Example 1:

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

Example 2:

```
Input: [1,2,3,4]
Output: 24
```

Note:

1. The length of the given array will be in range [3,10^4] and all elements are in the range [-1000, 1000].
2. Multiplication of any three numbers in the input won't exceed the range of 32-bit signed integer.

633. Sum of Square Numbers

DescriptionHintsSubmissionsDiscussSolution

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

Example 1:

```
Input: 5
Output: True
Explanation: 1 * 1 + 2 * 2 = 5
```

Example 2:

```
Input: 3
Output: False
```

637. Average of Levels in Binary Tree

DescriptionHintsSubmissionsDiscussSolution

Given a non-empty binary tree, return the average value of the nodes on each level in the form of an array.

Example 1:

Input:

```
    3
   / \
  9  20
    /  \
   15   7
```

Output: [3, 14.5, 11]
Explanation:
The average value of nodes on level 0 is 3, on level 1 is 14.5, and on level 2 is 11. Hence return [3, 14.5, 11].

Note:

1. The range of node's value is in the range of 32-bit signed integer.

643. Maximum Average Subarray I

DescriptionHintsSubmissionsDiscussSolution

Given an array consisting of n integers, find the contiguous subarray of given length k that has the maximum average value. And you need to output the maximum average value.

Example 1:

Input: [1,12,-5,-6,50,3], k = 4
Output: 12.75
Explanation: Maximum average is (12-5-6+50)/4 = 51/4 = 12.75

Note:

1. 1 <= k <= n <= 30,000.

2. Elements of the given array will be in the range [-10,000, 10,000].

---

645. Set Mismatch

DescriptionHintsSubmissionsDiscussSolution

The set S originally contains numbers from 1 to n. But unfortunately, due to the data error, one of the numbers in the set got duplicated to another number in the set, which results in repetition of one number and loss of another number.

Given an array nums representing the data status of this set after the error. Your task is to firstly find the number occurs twice and then find the number that is missing. Return them in the form of an array.

Example 1:

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

Note:

1. The given array size will in the range [2, 10000].
2. The given array's numbers won't have any order.

---

653. Two Sum IV - Input is a BST

DescriptionHintsSubmissionsDiscussSolution

Given a Binary Search Tree and a target number, return true if there exist two elements in the BST such that their sum is equal to the given target.

Example 1:

Input:

```
    5
   / \
  3   6
 / \   \
2   4   7
```

Target = 9

Output: True

Example 2:

Input:

```
    5
   / \
  3   6
 / \   \
2   4   7
```

Target = 28

Output: False

657. Robot Return to Origin

DescriptionHintsSubmissionsDiscussSolution

There is a robot starting at position (0, 0), the origin, on a 2D plane. Given a sequence of its moves, judge if this robot **ends up at (0, 0)** after it completes its moves.

The move sequence is represented by a string, and the character moves[i] represents its ith move. Valid moves are R (right), L (left), U (up), and D (down). If the robot returns to the origin after it finishes all of its moves, return true. Otherwise, return false.

Note: The way that the robot is "facing" is irrelevant. "R" will always make the robot move to the right once, "L" will always make it move left, etc. Also, assume that the magnitude of the robot's movement is the same for each move.

Example 1:

Input: "UD"
Output: true
Explanation: The robot moves up once, and then down once. All moves have the same magnitude, so it ended up at the origin where it started. Therefore, we return true.

Example 2:

Input: "LL"
Output: false
Explanation: The robot moves left twice. It ends up two "moves" to the left of the origin. We return false because it is not at the origin at the end of its moves.

---

661. Image Smoother

DescriptionHintsSubmissionsDiscussSolution

Given a 2D integer matrix M representing the gray scale of an image, you need to design a smoother to make the gray scale of each cell becomes the average gray scale (rounding down) of all the 8 surrounding cells and itself. If a cell has less than 8 surrounding cells, then use as many as you can.

Example 1:

Input:
[[1,1,1],
 [1,0,1],
 [1,1,1]]
Output:
[[0, 0, 0],
 [0, 0, 0],
 [0, 0, 0]]
Explanation:
For the point (0,0), (0,2), (2,0), (2,2): floor(3/4) = floor(0.75) = 0

For the point (0,1), (1,0), (1,2), (2,1): floor(5/6) = floor(0.83333333) = 0
For the point (1,1): floor(8/9) = floor(0.88888889) = 0

Note:

1. The value in the given matrix is in the range of [0, 255].
2. The length and width of the given matrix are in the range of [1, 150].

664. Strange Printer

DescriptionHintsSubmissionsDiscussSolution

There is a strange printer with the following two special requirements:

1. The printer can only print a sequence of the same character each time.
2. At each turn, the printer can print new characters starting from and ending at any places, and will cover the original existing characters.

Given a string consists of lower English letters only, your job is to count the minimum number of turns the printer needed in order to print it.

Example 1:

Input: "aaabbb"
Output: 2
Explanation: Print "aaa" first and then print "bbb".

Example 2:

Input: "aba"
Output: 2
Explanation: Print "aaa" first and then print "b" from the second place of the string, which will cover the existing character 'a'.

Hint: Length of the given string will not exceed 100.

665. Non-decreasing Array

DescriptionHintsSubmissionsDiscussSolution

Given an array with n integers, your task is to check if it could become non-decreasing by modifying **at most** 1 element.

We define an array is non-decreasing if array[i] <= array[i + 1] holds for every i (1 <= i < n).

Example 1:

Input: [4,2,3]
Output: True
Explanation: You could modify the first 4 to 1 to get a non-decreasing array.

Example 2:

Input: [4,2,1]
Output: False
Explanation: You can't get a non-decreasing array by modify at most one element.

Note: The n belongs to [1, 10,000].

669. Trim a Binary Search Tree

DescriptionHintsSubmissionsDiscussSolution

Given a binary search tree and the lowest and highest boundaries as L and R, trim the tree so that all its elements lies in [L, R] (R >= L). You might need to change the root of the tree, so the result should return the new root of the trimmed binary search tree.

Example 1:

Input:
```
    1
   / \
```

```
  0   2

  L = 1
  R = 2
```

Output:

```
    1
      \
       2
```

Example 2:

Input:

```
    3
   / \
  0   4
   \
    2
   /
  1

  L = 1
  R = 3
```

Output:

```
      3
     /
   2
  /
 1
```

671. Second Minimum Node In a Binary Tree

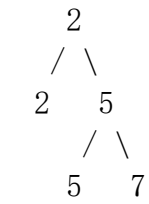
DescriptionHintsSubmissionsDiscussSolution

Given a non-empty special binary tree consisting of nodes with the non-negative value, where each node in this tree has exactly two or zero sub-node. If the node has two sub-nodes, then this node's value is the smaller value among its two sub-nodes.

Given such a binary tree, you need to output the **second minimum** value in the set made of all the nodes' value in the whole tree.

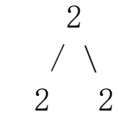
If no such second minimum value exists, output -1 instead.

Example 1:

Input:


Output: 5
Explanation: The smallest value is 2, the second smallest value is 5.

Example 2:

Input:


Output: -1
Explanation: The smallest value is 2, but there isn't any second smallest value.

674. Longest Continuous Increasing Subsequence

DescriptionHintsSubmissionsDiscussSolution

Given an unsorted array of integers, find the length of longest continuous increasing subsequence (subarray).

Example 1:

Input: [1,3,5,4,7]
Output: 3
Explanation: The longest continuous increasing subsequence is [1,3,5], its length is 3.
Even though [1,3,5,7] is also an increasing subsequence, it's not a continuous one where 5 and 7 are separated by 4.

Example 2:

```
Input: [2,2,2,2,2]
Output: 1
Explanation: The longest continuous increasing subsequence is [2],
its length is 1.
```

Note: Length of the array will not exceed 10,000.

680. Valid Palindrome II

DescriptionHintsSubmissionsDiscussSolution

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:

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

682. Baseball Game

DescriptionHintsSubmissionsDiscussSolution

You're now a baseball game point recorder.

Given a list of strings, each string can be one of the 4 following types:

1. Integer (one round's score): Directly represents the number of points you get in this round.
2. "+" (one round's score): Represents that the points you get in this round are the sum of the last two valid round's points.
3. "D" (one round's score): Represents that the points you get in this round are the doubled data of the last valid round's points.
4. "C" (an operation, which isn't a round's score): Represents the last valid round's points you get were invalid and should be removed.

Each round's operation is permanent and could have an impact on the round before and the round after.

You need to return the sum of the points you could get in all the rounds.

Example 1:

Input: ["5","2","C","D","+"]
Output: 30
Explanation:
Round 1: You could get 5 points. The sum is: 5.
Round 2: You could get 2 points. The sum is: 7.
Operation 1: The round 2's data was invalid. The sum is: 5.
Round 3: You could get 10 points (the round 2's data has been removed). The sum is: 15.
Round 4: You could get 5 + 10 = 15 points. The sum is: 30.

Example 2:

Input: ["5","-2","4","C","D","9","+","+"]
Output: 27
Explanation:
Round 1: You could get 5 points. The sum is: 5.
Round 2: You could get -2 points. The sum is: 3.
Round 3: You could get 4 points. The sum is: 7.
Operation 1: The round 3's data is invalid. The sum is: 3.
Round 4: You could get -4 points (the round 3's data has been removed). The sum is: -1.
Round 5: You could get 9 points. The sum is: 8.
Round 6: You could get -4 + 9 = 5 points. The sum is 13.
Round 7: You could get 9 + 5 = 14 points. The sum is 27.

Note:

- The size of the input list will be between 1 and 1000.

- Every integer represented in the list will be between -30000 and 30000.

686. Repeated String Match

DescriptionHintsSubmissionsDiscussSolution

Given two strings A and B, find the minimum number of times A has to be repeated such that B is a substring of it. If no such solution, return -1.

For example, with A = "abcd" and B = "cdabcdab".

Return 3, because by repeating A three times (“abcdabcdabcd”), B is a substring of it; and B is not a substring of A repeated two times ("abcdabcd").

Note:

The length of A and B will be between 1 and 10000.

687. Longest Univalue Path

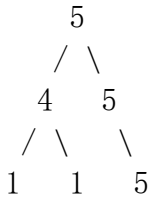
DescriptionHintsSubmissionsDiscussSolution

Given a binary tree, find the length of the longest path where each node in the path has the same value. This path may or may not pass through the root.

Note: The length of path between two nodes is represented by the number of edges between them.

Example 1:

Input:

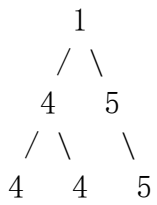


Output:

2

Example 2:

Input:



Output:

2

Note: The given binary tree has not more than 10000 nodes. The height of the tree is not more than 1000.

690. Employee Importance

DescriptionHintsSubmissionsDiscussSolution

You are given a data structure of employee information, which includes the employee's **unique id**, his **importance value** and his **direct** subordinates' id.

For example, employee 1 is the leader of employee 2, and employee 2 is the leader of employee 3. They have importance value 15, 10 and 5, respectively. Then employee 1 has a data structure like [1, 15, [2]], and employee 2 has [2, 10, [3]], and employee 3 has [3, 5, []]. Note that although employee 3 is also a subordinate of employee 1, the relationship is **not direct**.

Now given the employee information of a company, and an employee id, you need to return the total importance value of this employee and all his subordinates.

Example 1:

Input: [[1, 5, [2, 3]], [2, 3, []], [3, 3, []]], 1

Output: 11
Explanation:
Employee 1 has importance value 5, and he has two direct subordinates: employee 2 and employee 3. They both have importance value 3. So the total importance value of employee 1 is 5 + 3 + 3 = 11.

Note:

1. One employee has at most one **direct** leader and may have several subordinates.
2. The maximum number of employees won't exceed 2000.

695. Max Area of Island

DescriptionHintsSubmissionsDiscussSolution

Given a non-empty 2D array grid of 0's and 1's, an **island** is a group of 1's (representing land) connected 4-directionally (horizontal or vertical.) You may assume all four edges of the grid are surrounded by water.

Find the maximum area of an island in the given 2D array. (If there is no island, the maximum area is 0.)

Example 1:

```
[[0,0,1,0,0,0,0,1,0,0,0,0,0],
 [0,0,0,0,0,0,0,1,1,1,0,0,0],
 [0,1,1,0,1,0,0,0,0,0,0,0,0],
 [0,1,0,0,1,1,0,0,1,0,1,0,0],
 [0,1,0,0,1,1,0,0,1,1,1,0,0],
 [0,0,0,0,0,0,0,0,0,0,1,0,0],
 [0,0,0,0,0,0,0,1,1,1,0,0,0],
 [0,0,0,0,0,0,0,1,1,0,0,0,0]]
```

Given the above grid, return 6. Note the answer is not 11, because the island must be connected 4-directionally.

Example 2:

```
[[0,0,0,0,0,0,0,0]]
```

Given the above grid, return 0.

Note: The length of each dimension in the given grid does not exceed 50.

696. Count Binary Substrings

DescriptionHintsSubmissionsDiscussSolution

Give a string s, count the number of non-empty (contiguous) substrings that have the same number of 0's and 1's, and all the 0's and all the 1's in these substrings are grouped consecutively.

Substrings that occur multiple times are counted the number of times they occur.

Example 1:

Input: "00110011"
Output: 6
Explanation: There are 6 substrings that have equal number of consecutive 1's and 0's: "0011", "01", "1100", "10", "0011", and "01".

Notice that some of these substrings repeat and are counted the number of times they occur.

Also, "00110011" is not a valid substring because **all** the 0's (and 1's) are not grouped together.

Example 2:

Input: "10101"
Output: 4
Explanation: There are 4 substrings: "10", "01", "10", "01" that have equal number of consecutive 1's and 0's.

Note:

- s.length will be between 1 and 50,000.
- s will only consist of "0" or "1" characters.

697. Degree of an Array

DescriptionHintsSubmissionsDiscussSolution

Given a non-empty array of non-negative integers nums, the **degree** of this array is defined as the maximum frequency of any one of its elements.

Your task is to find the smallest possible length of a (contiguous) subarray of nums, that has the same degree as nums.

Example 1:

Input: [1, 2, 2, 3, 1]
Output: 2
Explanation:
The input array has a degree of 2 because both elements 1 and 2 appear twice.
Of the subarrays that have the same degree:
[1, 2, 2, 3, 1], [1, 2, 2, 3], [2, 2, 3, 1], [1, 2, 2], [2, 2, 3], [2, 2]
The shortest length is 2. So return 2.

Example 2:

Input: [1,2,2,3,1,4,2]
Output: 6

Note:

- nums.length will be between 1 and 50,000.
- nums[i] will be an integer between 0 and 49,999.

701-800

700. Search in a Binary Search Tree

分析

难度 易
来源
https://leetcode.com/problems/search-in-a-binary-search-tree/

题目

Given the root node of a binary search tree (BST) and a value. You need to find the node in the BST that the node's value equals the given value. Return the subtree rooted with that node. If such node doesn't exist, you should return NULL.

For example,

Given the tree:

```
        4
       / \
      2   7
     / \
    1   3
```

And the value to search: 2

You should return this subtree:

```
      2
     / \
    1   3
```

In the example above, if we want to search the value 5, since there is no node with value 5, we should return NULL.

Note that an empty tree is represented by NULL, therefore you would see the expected output (serialized tree format) as [], not null.

解答

Runtime: 2 ms, faster than 92.56% of Java online submissions for Search in a Binary Search Tree.
Memory Usage: 26.8 MB, less than 84.57% of Java online submissions for Search in a Binary Search Tree.

```
package LeetCode;

import java.util.LinkedList;

public class L700_SearchInABinarySearchTree {
   public TreeNode searchBST(TreeNode root, int val) {
        if(root==null||root.val==val)
            return root;
        LinkedList<TreeNode> stack=new LinkedList<TreeNode>();
        stack.addLast(root);
        while(!stack.isEmpty()){
            TreeNode tn=stack.removeLast();
            if(tn==null||tn.val==val)
                return tn;
            else if(tn.val>val)
                stack.addLast(tn.left);
            else
                stack.addLast(tn.right);
        }
        return null;
    }
   /*public TreeNode searchBST(TreeNode root, int val) {//是二叉搜索树，有序的
        if(root==null||root.val==val)
            return root;
        else if(root.val<val)
            return searchBST(root.right,val);
        else
            return searchBST(root.left,val);
   }*/

    public static void main(String[] args){
        L700_SearchInABinarySearchTree l700=new L700_SearchInABinarySearchTree();
        Integer[] nums={4,2,7,1,3};
        Tree tree=new Tree();
        TreeNode root=tree.buildTree(nums);
        tree.printTree(root);
        //TreeNode tn =l700.searchBST(root,5);
        TreeNode tn =l700.searchBST(root,2);
```

```
            tree.printTree(tn);
        }
    }
```

701. Insert into a Binary Search Tree

分析

难度 中
来源
https://leetcode.com/problems/insert-into-a-binary-search-tree/

题目

Medium

Given the root node of a binary search tree (BST) and a value to be inserted into the tree, insert the value into the BST. Return the root node of the BST after the insertion. It is guaranteed that the new value does not exist in the original BST.

Note that there may exist multiple valid ways for the insertion, as long as the tree remains a BST after insertion. You can return any of them.

For example,

Given the tree:

```
        4
       / \
      2   7
     / \
    1   3
```

And the value to insert: 5

You can return this binary search tree:

```
         4
       /   \
      2     7
     / \   /
    1   3 5
```

This tree is also valid:

```
         5
       /   \
      2     7
     / \
```

```
    1    3
          \
           4
```

解答

Runtime: 2 ms, faster than 87.98% of Java online submissions for Insert into a Binary Search Tree.
Memory Usage: 25.8 MB, less than 87.04% of Java online submissions for Insert into a Binary Search Tree.

```
package LeetCode;

public class L701_InsertIntoABinarySearchTree {
   public TreeNode insertIntoBST(TreeNode root, int val) {//是二叉搜索树，有序的
       /* if(root==null)
            root=new TreeNode(val);*/
         if(val<root.val){
            if(root.left==null)
                 root.left=new TreeNode(val);
            else
                 insertIntoBST(root.left,val);
         }else{
            if(root.right==null)
                root.right=new TreeNode(val);
            else
                insertIntoBST(root.right,val);
         }
         return root;
   }
    public static void main(String[] args){
        L701_InsertIntoABinarySearchTree l701=new L701_InsertIntoABinarySearchTree();
        //Integer[] nums={4,2,7,1,3};
        //[61,46,66,43,null,null,null,39,null,null,null]
        //88
        Integer[] nums={61,46,66,43,null,null,null,39,null,null,null};
        Tree tree=new Tree();
        TreeNode root=tree.buildTree(nums);
        tree.printTree(root);
        TreeNode tn =l701.insertIntoBST(root,88);
        tree.printTree(tn);
```

```
        }
    }
```

703. Kth Largest Element in a Stream

DescriptionHintsSubmissionsDiscussSolution

Design a class to find the **k**th largest element in a stream. Note that it is the kth largest element in the sorted order, not the kth distinct element.

Your KthLargest class will have a constructor which accepts an integer k and an integer array nums, which contains initial elements from the stream. For each call to the method KthLargest.add, return the element representing the kth largest element in the stream.

Example:

```
int k = 3;
int[] arr = [4,5,8,2];
KthLargest kthLargest = new KthLargest(3, arr);
kthLargest.add(3);   // returns 4
kthLargest.add(5);   // returns 5
kthLargest.add(10);  // returns 5
kthLargest.add(9);   // returns 8
kthLargest.add(4);   // returns 8
```

Note:

You may assume that nums' length ≥ k-1 and k ≥ 1.

704. Binary Search

DescriptionHintsSubmissionsDiscussSolution

Given a **sorted** (in ascending order) integer array nums of n elements and a target value, write a function to search target in nums. If target exists, then return its index, otherwise return -1.

Example 1:

Input: nums = [-1,0,3,5,9,12], target = 9
Output: 4

Explanation: 9 exists in nums and its index is 4

Example 2:

```
Input: nums = [-1,0,3,5,9,12], target = 2
Output: -1
Explanation: 2 does not exist in nums so return -1
```

Note:

1. You may assume that all elements in nums are unique.
2. n will be in the range [1, 10000].
3. The value of each element in nums will be in the range [-9999, 9999].

706. Design HashMap

DescriptionHintsSubmissionsDiscussSolution

Design a HashMap without using any built-in hash table libraries.

To be specific, your design should include these functions:

- put(key, value) : Insert a (key, value) pair into the HashMap. If the value already exists in the HashMap, update the value.
- get(key): Returns the value to which the specified key is mapped, or -1 if this map contains no mapping for the key.
- remove(key) : Remove the mapping for the value key if this map contains the mapping for the key.

Example:

```
MyHashMap hashMap = new MyHashMap();
hashMap.put(1, 1);
hashMap.put(2, 2);
hashMap.get(1);                  // returns 1
hashMap.get(3);                  // returns -1 (not found)
hashMap.put(2, 1);               // update the existing value
hashMap.get(2);                  // returns 1
hashMap.remove(2);               // remove the mapping for 2
```

```
hashMap.get(2);                    // returns -1 (not found)
```

Note:

- All keys and values will be in the range of [0, 1000000].
- The number of operations will be in the range of [1, 10000].
- Please do not use the built-in HashMap library.

---

707. Design Linked List

DescriptionHintsSubmissionsDiscussSolution

Design your implementation of the linked list. You can choose to use the singly linked list or the doubly linked list. A node in a singly linked list should have two attributes: val and next. val is the value of the current node, and next is a pointer/reference to the next node. If you want to use the doubly linked list, you will need one more attribute prev to indicate the previous node in the linked list. Assume all nodes in the linked list are 0-indexed.

Implement these functions in your linked list class:

- get(index) : Get the value of the index-th node in the linked list. If the index is invalid, return -1.
- addAtHead(val) : Add a node of value val before the first element of the linked list. After the insertion, the new node will be the first node of the linked list.
- addAtTail(val) : Append a node of value val to the last element of the linked list.
- addAtIndex(index, val) : Add a node of value val before the index-th node in the linked list. If index equals to the length of linked list, the node will be appended to the end of linked list. If index is greater than the length, the node will not be inserted.
- deleteAtIndex(index) : Delete the index-th node in the linked list, if the index is valid.

Example:

```
MyLinkedList linkedList = new MyLinkedList();
linkedList.addAtHead(1);
linkedList.addAtTail(3);
linkedList.addAtIndex(1, 2);  // linked list becomes 1->2->3
linkedList.get(1);            // returns 2
```

```
linkedList.deleteAtIndex(1);  // now the linked list is 1->3
linkedList.get(1);            // returns 3
```

Note:

- All values will be in the range of [1, 1000].
- The number of operations will be in the range of [1, 1000].
- Please do not use the built-in LinkedList library.

---

709. To Lower Case

DescriptionHintsSubmissionsDiscussSolution

Implement function ToLowerCase() that has a string parameter str, and returns the same string in lowercase.

Example 1:

```
Input: "Hello"
Output: "hello"
```

Example 2:

```
Input: "here"
Output: "here"
```

Example 3:

```
Input: "LOVELY"
Output: "lovely"
```

717. 1-bit and 2-bit Characters

DescriptionHintsSubmissionsDiscussSolution

We have two special characters. The first character can be represented by one bit 0. The second character can be represented by two bits (10 or 11).

Now given a string represented by several bits. Return whether the last character must be a one-bit character or not. The given string will always end with a zero.

Example 1:

Input:
bits = [1, 0, 0]
Output: True
Explanation:
The only way to decode it is two-bit character and one-bit character. So the last character is one-bit character.

Example 2:

Input:
bits = [1, 1, 1, 0]
Output: False
Explanation:
The only way to decode it is two-bit character and two-bit character. So the last character is NOT one-bit character.

Note:

- 1 <= len(bits) <= 1000.
- bits[i] is always 0 or 1.

720. Longest Word in Dictionary

DescriptionHintsSubmissionsDiscussSolution

Given a list of strings words representing an English Dictionary, find the longest word in words that can be built one character at a time by other words in words. If there is more than one possible answer, return the longest word with the smallest lexicographical order.

If there is no answer, return the empty string.

Example 1:

Input:
words = ["w","wo","wor","worl", "world"]
Output: "world"

Explanation:
The word "world" can be built one character at a time by "w", "wo", "wor", and "worl".

Example 2:

Input:
words = ["a", "banana", "app", "appl", "ap", "apply", "apple"]
Output: "apple"
Explanation:
Both "apply" and "apple" can be built from other words in the dictionary. However, "apple" is lexicographically smaller than "apply".

Note:

- All the strings in the input will only contain lowercase letters.
- The length of `words` will be in the range `[1, 1000]`.
- The length of `words[i]` will be in the range `[1, 30]`.

724. Find Pivot Index

DescriptionHintsSubmissionsDiscussSolution

Given an array of integers nums, write a method that returns the "pivot" index of this array.

We define the pivot index as the index where the sum of the numbers to the left of the index is equal to the sum of the numbers to the right of the index.

If no such index exists, we should return -1. If there are multiple pivot indexes, you should return the left-most pivot index.

Example 1:

Input:
nums = [1, 7, 3, 6, 5, 6]
Output: 3
Explanation:
The sum of the numbers to the left of index 3 (nums[3] = 6) is equal to the sum of numbers to the right of index 3.
Also, 3 is the first index where this occurs.

Example 2:

Input:
nums = [1, 2, 3]
Output: -1
Explanation:
There is no index that satisfies the conditions in the problem statement.

Note:

- The length of nums will be in the range [0, 10000].
- Each element nums[i] will be an integer in the range [-1000, 1000].

728. Self Dividing Numbers

DescriptionHintsSubmissionsDiscussSolution

A *self-dividing number* is a number that is divisible by every digit it contains.

For example, 128 is a self-dividing number because 128 % 1 == 0, 128 % 2 == 0, and 128 % 8 == 0.

Also, a self-dividing number is not allowed to contain the digit zero.

Given a lower and upper number bound, output a list of every possible self dividing number, including the bounds if possible.

Example 1:

Input:
left = 1, right = 22
Output: [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 15, 22]

Note:

- The boundaries of each input argument are 1 <= left <= right <= 10000.

---

733. Flood Fill

DescriptionHintsSubmissionsDiscussSolution

An image is represented by a 2-D array of integers, each integer representing the pixel value of the image (from 0 to 65535).

Given a coordinate (sr, sc) representing the starting pixel (row and column) of the flood fill, and a pixel value newColor, "flood fill" the image.

To perform a "flood fill", consider the starting pixel, plus any pixels connected 4-directionally to the starting pixel of the same color as the starting pixel, plus any pixels connected 4-directionally to those pixels (also with the same color as the starting pixel), and so on. Replace the color of all of the aforementioned pixels with the newColor.

At the end, return the modified image.

Example 1:

Input:
image = [[1,1,1],[1,1,0],[1,0,1]]
sr = 1, sc = 1, newColor = 2
Output: [[2,2,2],[2,2,0],[2,0,1]]
Explanation:
From the center of the image (with position (sr, sc) = (1, 1)), all pixels connected
by a path of the same color as the starting pixel are colored with the new color.
Note the bottom corner is not colored 2, because it is not 4-directionally connected
to the starting pixel.

Note:

- The length of `image` and `image[0]` will be in the range [1, 50].
- The given starting pixel will satisfy `0 <= sr < image.length` and `0 <= sc < image[0].length`.
- The value of each color in `image[i][j]` and `newColor` will be an integer in [0, 65535].

743. Network Delay Time

DescriptionHintsSubmissionsDiscussSolution

There are N network nodes, labelled 1 to N.

Given times, a list of travel times as **directed** edges times[i] = (u, v, w), where u is the source node, v is the target node, and w is the time it takes for a signal to travel from source to target.

Now, we send a signal from a certain node K. How long will it take for all nodes to receive the signal? If it is impossible, return -1.

Note:

1. N will be in the range [1, 100].
2. K will be in the range [1, N].
3. The length of times will be in the range [1, 6000].
4. All edges times[i] = (u, v, w) will have 1 <= u, v <= N and 1 <= w <= 100.

744. Find Smallest Letter Greater Than Target

DescriptionHintsSubmissionsDiscussSolution

Given a list of sorted characters letters containing only lowercase letters, and given a target letter target, find the smallest element in the list that is larger than the given target.

Letters also wrap around. For example, if the target is target = 'z' and letters = ['a', 'b'], the answer is 'a'.

Examples:

Input:
letters = ["c", "f", "j"]
target = "a"
Output: "c"

Input:
letters = ["c", "f", "j"]
target = "c"

Output: "f"

Input:
letters = ["c", "f", "j"]
target = "d"
Output: "f"

Input:
letters = ["c", "f", "j"]
target = "g"
Output: "j"

Input:
letters = ["c", "f", "j"]
target = "j"
Output: "c"

Input:
letters = ["c", "f", "j"]
target = "k"
Output: "c"

Note:

1. letters has a length in range [2, 10000].
2. letters consists of lowercase letters, and contains at least 2 unique letters.
3. target is a lowercase letter.

746. Min Cost Climbing Stairs

DescriptionHintsSubmissionsDiscussSolution

On a staircase, the i-th step has some non-negative cost cost[i] assigned (0 indexed).

Once you pay the cost, you can either climb one or two steps. You need to find minimum cost to reach the top of the floor, and you can either start from the step with index 0, or the step with index 1.

Example 1:

Input: cost = [10, 15, 20]
Output: 15

Explanation: Cheapest is start on cost[1], pay that cost and go to the top.

Example 2:

Input: cost = [1, 100, 1, 1, 1, 100, 1, 1, 100, 1]
Output: 6
Explanation: Cheapest is start on cost[0], and only step on 1s, skipping cost[3].

Note:

1. cost will have a length in the range [2, 1000].
2. Every cost[i] will be an integer in the range [0, 999].

747. Largest Number At Least Twice of Others

DescriptionHintsSubmissionsDiscussSolution

In a given integer array nums, there is always exactly one largest element.

Find whether the largest element in the array is at least twice as much as every other number in the array.

If it is, return the **index** of the largest element, otherwise return -1.

Example 1:

Input: nums = [3, 6, 1, 0]
Output: 1
Explanation: 6 is the largest integer, and for every other number in the array x,
6 is more than twice as big as x. The index of value 6 is 1, so we return 1.

Example 2:

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

Explanation: 4 isn't at least as big as twice the value of 3, so we return -1.

Note:

1. nums will have a length in the range [1, 50].
2. Every nums[i] will be an integer in the range [0, 99].

748. Shortest Completing Word

DescriptionHintsSubmissionsDiscussSolution

Find the minimum length word from a given dictionary words, which has all the letters from the string licensePlate. Such a word is said to *complete* the given string licensePlate

Here, for letters we ignore case. For example, "P" on the licensePlate still matches "p" on the word.

It is guaranteed an answer exists. If there are multiple answers, return the one that occurs first in the array.

The license plate might have the same letter occurring multiple times. For example, given a licensePlate of "PP", the word "pair" does not complete the licensePlate, but the word "supper" does.

Example 1:

Input: licensePlate = "1s3 PSt", words = ["step", "steps", "stripe", "stepple"]
Output: "steps"
Explanation: The smallest length word that contains the letters "S", "P", "S", and "T".
Note that the answer is not "step", because the letter "s" must occur in the word twice.
Also note that we ignored case for the purposes of comparing whether a letter exists in the word.

Example 2:

Input: licensePlate = "1s3 456", words = ["looks", "pest", "stew", "show"]

Output: "pest"
Explanation: There are 3 smallest length words that contains the letters "s".
We return the one that occurred first.

Note:

1. licensePlate will be a string with length in range [1, 7].
2. licensePlate will contain digits, spaces, or letters (uppercase or lowercase).
3. words will have a length in the range [10, 1000].
4. Every words[i] will consist of lowercase letters, and have length in range [1, 15].

762. Prime Number of Set Bits in Binary Representation

DescriptionHintsSubmissionsDiscussSolution

Given two integers L and R, find the count of numbers in the range [L, R] (inclusive) having a prime number of set bits in their binary representation.

(Recall that the number of set bits an integer has is the number of 1s present when written in binary. For example, 21 written in binary is 10101 which has 3 set bits. Also, 1 is not a prime.)

Example 1:

Input: L = 6, R = 10
Output: 4
Explanation:
6 -> 110 (2 set bits, 2 is prime)
7 -> 111 (3 set bits, 3 is prime)
9 -> 1001 (2 set bits , 2 is prime)
10->1010 (2 set bits , 2 is prime)

Example 2:

Input: L = 10, R = 15
Output: 5
Explanation:
10 -> 1010 (2 set bits, 2 is prime)
11 -> 1011 (3 set bits, 3 is prime)
12 -> 1100 (2 set bits, 2 is prime)
13 -> 1101 (3 set bits, 3 is prime)

14 -> 1110 (3 set bits, 3 is prime)
15 -> 1111 (4 set bits, 4 is not prime)

Note:

1. L, R will be integers L <= R in the range [1, 10^6].
2. R - L will be at most 10000.

766. Toeplitz Matrix

DescriptionHintsSubmissionsDiscussSolution

A matrix is *Toeplitz* if every diagonal from top-left to bottom-right has the same element.

Now given an M x N matrix, return True if and only if the matrix is *Toeplitz*.

Example 1:

Input:
```
matrix = [
  [1,2,3,4],
  [5,1,2,3],
  [9,5,1,2]
]
```
Output: True
Explanation:
In the above grid, the diagonals are:
"[9]", "[5, 5]", "[1, 1, 1]", "[2, 2, 2]", "[3, 3]", "[4]".
In each diagonal all elements are the same, so the answer is True.

Example 2:

Input:
```
matrix = [
  [1,2],
  [2,2]
]
```
Output: False
Explanation:

The diagonal "[1, 2]" has different elements.

Note:

1. matrix will be a 2D array of integers.
2. matrix will have a number of rows and columns in range [1, 20].
3. matrix[i][j] will be integers in range [0, 99].

Follow up:

1. What if the matrix is stored on disk, and the memory is limited such that you can only load at most one row of the matrix into the memory at once?
2. What if the matrix is so large that you can only load up a partial row into the memory at once?

771. Jewels and Stones

DescriptionHintsSubmissionsDiscussSolution

You're given strings J representing the types of stones that are jewels, and S representing the stones you have. Each character in S is a type of stone you have. You want to know how many of the stones you have are also jewels.

The letters in J are guaranteed distinct, and all characters in J and S are letters. Letters are case sensitive, so "a" is considered a different type of stone from "A".

Example 1:

```
Input: J = "aA", S = "aAAbbbb"
Output: 3
```

Example 2:

```
Input: J = "z", S = "ZZ"
Output: 0
```

Note:

- S and J will consist of letters and have length at most 50.
- The characters in J are distinct.

783. Minimum Distance Between BST Nodes

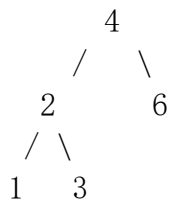
DescriptionHintsSubmissionsDiscussSolution

Given a Binary Search Tree (BST) with the root node root, return the minimum difference between the values of any two different nodes in the tree.

Example :

Input: root = [4,2,6,1,3,null,null]
Output: 1
Explanation:
Note that root is a TreeNode object, not an array.

The given tree [4,2,6,1,3,null,null] is represented by the following diagram:



while the minimum difference in this tree is 1, it occurs between node 1 and node 2, also between node 3 and node 2.

Note:

1. The size of the BST will be between 2 and 100.
2. The BST is always valid, each node's value is an integer, and each node's value is different.

784. Letter Case Permutation

DescriptionHintsSubmissionsDiscussSolution

Given a string S, we can transform every letter individually to be lowercase or uppercase to create another string. Return a list of all possible strings we could create.

Examples:
Input: S = "a1b2"
Output: ["a1b2", "a1B2", "A1b2", "A1B2"]

Input: S = "3z4"
Output: ["3z4", "3Z4"]

Input: S = "12345"
Output: ["12345"]

Note:

- S will be a string with length between 1 and 12.
- S will consist only of letters or digits.

788. Rotated Digits

DescriptionHintsSubmissionsDiscussSolution

X is a good number if after rotating each digit individually by 180 degrees, we get a valid number that is different from X. Each digit must be rotated - we cannot choose to leave it alone.

A number is valid if each digit remains a digit after rotation. 0, 1, and 8 rotate to themselves; 2 and 5 rotate to each other; 6 and 9 rotate to each other, and the rest of the numbers do not rotate to any other number and become invalid.

Now given a positive number N, how many numbers X from 1 to N are good?

Example:
Input: 10
Output: 4
Explanation:
There are four good numbers in the range [1, 10] : 2, 5, 6, 9.
Note that 1 and 10 are not good numbers, since they remain unchanged after rotating.

Note:

- N will be in range [1, 10000].

796. Rotate String

DescriptionHintsSubmissionsDiscussSolution

We are given two strings, A and B.

A *shift on A* consists of taking string A and moving the leftmost character to the rightmost position. For example, if A = 'abcde', then it will be 'bcdea' after one shift on A. Return True if and only if A can become B after some number of shifts on A.

Example 1:
Input: A = 'abcde', B = 'cdeab'
Output: true

Example 2:
Input: A = 'abcde', B = 'abced'
Output: false

Note:

- A and B will have length at most 100.

804. Unique Morse Code Words

DescriptionHintsSubmissionsDiscussSolution

International Morse Code defines a standard encoding where each letter is mapped to a series of dots and dashes, as follows: "a" maps to ".-", "b" maps to "-...", "c" maps to "-.-.", and so on.

For convenience, the full table for the 26 letters of the English alphabet is given below:

[".-","-...","-.-.","-..",".","..-.","--.","....","..",".---","-.-","-..","--","-.","---",".--.","--.-",".-.","...","-","..-","...-",".--","-..-","-.--","--.."]

Now, given a list of words, each word can be written as a concatenation of the Morse code of each letter. For example, "cab" can be written as "-.-.-....-", (which is the concatenation "-.-." + "-..." + ".-"). We'll call such a concatenation, the transformation of a word.

Return the number of different transformations among all words we have.

Example:
Input: words = ["gin", "zen", "gig", "msg"]
Output: 2
Explanation:
The transformation of each word is:
"gin" -> "--...-."
"zen" -> "--...-."
"gig" -> "--...--."
"msg" -> "--...--."

There are 2 different transformations, "--...-." and "--...--.".

Note:

- The length of words will be at most 100.
- Each words[i] will have length in range [1, 12].
- words[i] will only consist of lowercase letters.

---

806. Number of Lines To Write String

DescriptionHintsSubmissionsDiscussSolution

We are to write the letters of a given string S, from left to right into lines. Each line has maximum width 100 units, and if writing a letter would cause the width of the line to exceed 100 units, it is written on the next line. We are given an array widths, an array where widths[0] is the width of 'a', widths[1] is the width of 'b', ..., and widths[25] is the width of 'z'.

Now answer two questions: how many lines have at least one character from S, and what is the width used by the last such line? Return your answer as an integer list of length 2.

Example :
Input:
widths =
[10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10]
S = "abcdefghijklmnopqrstuvwxyz"
Output: [3, 60]
Explanation:
All letters have the same length of 10. To write all 26 letters,
we need two full lines and one line with 60 units.
Example :
Input:
widths =
[4, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10]
S = "bbbcccdddaaa"
Output: [2, 4]
Explanation:
All letters except 'a' have the same length of 10, and
"bbbcccdddaa" will cover 9 * 10 + 2 * 4 = 98 units.
For the last 'a', it is written on the second line because
there is only 2 units left in the first line.
So the answer is 2 lines, plus 4 units in the second line.

Note:

- The length of S will be in the range [1, 1000].
- S will only contain lowercase letters.
- widths is an array of length 26.
- widths[i] will be in the range of [2, 10].

811. Subdomain Visit Count

DescriptionHintsSubmissionsDiscussSolution

A website domain like "discuss.leetcode.com" consists of various subdomains. At the top level, we have "com", at the next level, we have "leetcode.com", and at the lowest level, "discuss.leetcode.com". When we visit a domain like "discuss.leetcode.com", we will also visit the parent domains "leetcode.com" and "com" implicitly.

Now, call a "count-paired domain" to be a count (representing the number of visits this domain received), followed by a space, followed by the address. An example of a count-paired domain might be "9001 discuss.leetcode.com".

We are given a list cpdomains of count-paired domains. We would like a list of count-paired domains, (in the same format as the input, and in any order), that explicitly counts the number of visits to each subdomain.

Example 1:
Input:
["9001 discuss.leetcode.com"]
Output:
["9001 discuss.leetcode.com", "9001 leetcode.com", "9001 com"]
Explanation:
We only have one website domain: "discuss.leetcode.com". As discussed above, the subdomain "leetcode.com" and "com" will also be visited. So they will all be visited 9001 times.

Example 2:
Input:
["900 google.mail.com", "50 yahoo.com", "1 intel.mail.com", "5 wiki.org"]
Output:
["901 mail.com","50 yahoo.com","900 google.mail.com","5 wiki.org","5 org","1 intel.mail.com","951 com"]
Explanation:
We will visit "google.mail.com" 900 times, "yahoo.com" 50 times, "intel.mail.com" once and "wiki.org" 5 times. For the subdomains, we will visit "mail.com" 900 + 1 = 901 times, "com" 900 + 50 + 1 = 951 times, and "org" 5 times.

Notes:

- The length of cpdomains will not exceed 100.
- The length of each domain name will not exceed 100.
- Each address will have either 1 or 2 "." characters.

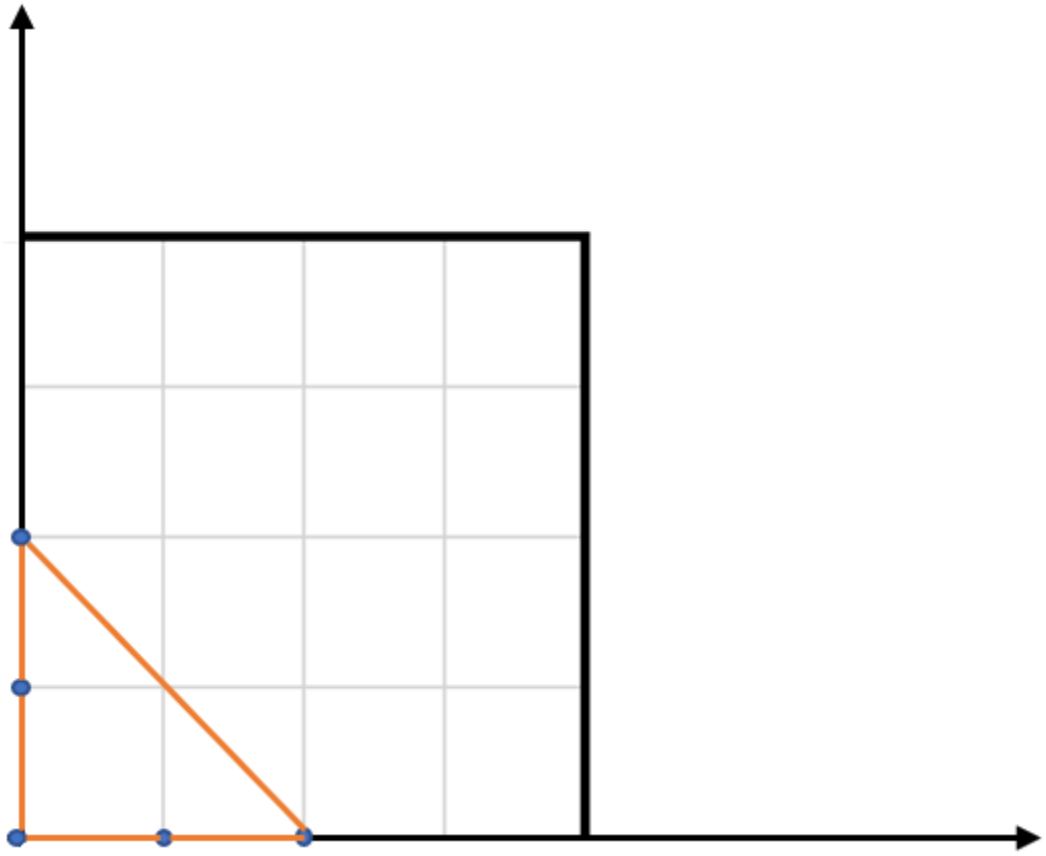
- The input count in any count-paired domain will not exceed 10000.
- The answer output can be returned in any order.

812. Largest Triangle Area

DescriptionHintsSubmissionsDiscussSolution

You have a list of points in the plane. Return the area of the largest triangle that can be formed by any 3 of the points.

Example:
Input: points = [[0,0],[0,1],[1,0],[0,2],[2,0]]
Output: 2
Explanation:
The five points are show in the figure below. The red triangle is the largest.



Notes:

- 3 <= points.length <= 50.
- No points will be duplicated.
- -50 <= points[i][j] <= 50.
- Answers within 10^-6 of the true value will be accepted as correct.

819. Most Common Word

DescriptionHintsSubmissionsDiscussSolution

Given a paragraph and a list of banned words, return the most frequent word that is not in the list of banned words. It is guaranteed there is at least one word that isn't banned, and that the answer is unique.

Words in the list of banned words are given in lowercase, and free of punctuation. Words in the paragraph are not case sensitive. The answer is in lowercase.

Example:
Input:
paragraph = "Bob hit a ball, the hit BALL flew far after it was hit."
banned = ["hit"]
Output: "ball"
Explanation:
"hit" occurs 3 times, but it is a banned word.
"ball" occurs twice (and no other word does), so it is the most frequent non-banned word in the paragraph.
Note that words in the paragraph are not case sensitive,
that punctuation is ignored (even if adjacent to words, such as "ball,"),
and that "hit" isn't the answer even though it occurs more because it is banned.

Note:

- 1 <= paragraph.length <= 1000.
- 1 <= banned.length <= 100.
- 1 <= banned[i].length <= 10.
- The answer is unique, and written in lowercase (even if its occurrences in paragraph may have uppercase symbols, and even if it is a proper noun.)

- paragraph only consists of letters, spaces, or the punctuation symbols !?', ;.
- Different words in paragraph are always separated by a space.
- There are no hyphens or hyphenated words.
- Words only consist of letters, never apostrophes or other punctuation symbols.

821. Shortest Distance to a Character

DescriptionHintsSubmissionsDiscussSolution

Given a string S and a character C, return an array of integers representing the shortest distance from the character C in the string.

Example 1:

Input: S = "loveleetcode", C = 'e'
Output: [3, 2, 1, 0, 1, 0, 0, 1, 2, 2, 1, 0]

Note:

1. S string length is in [1, 10000].
2. C is a single character, and guaranteed to be in string S.
3. All letters in S and C are lowercase.

---

824. Goat Latin

DescriptionHintsSubmissionsDiscussSolution

A sentence S is given, composed of words separated by spaces. Each word consists of lowercase and uppercase letters only.

We would like to convert the sentence to "*Goat Latin*" (a made-up language similar to Pig Latin.)

The rules of Goat Latin are as follows:

- If a word begins with a vowel (a, e, i, o, or u), append "ma" to the end of the word.
- For example, the word 'apple' becomes 'applema'.
-
- If a word begins with a consonant (i.e. not a vowel), remove the first letter and append it to the end, then add "ma".
- For example, the word "goat" becomes "oatgma".
-
- Add one letter 'a' to the end of each word per its word index in the sentence, starting with 1.
- For example, the first word gets "a" added to the end, the second word gets "aa" added to the end and so on.

Return the final sentence representing the conversion from S to Goat Latin.

Example 1:

Input: "I speak Goat Latin"
Output: "Imaa peaksmaaa oatGmaaaa atinLmaaaaa"

Example 2:

Input: "The quick brown fox jumped over the lazy dog"
Output: "heTmaa uickqmaaa rownbmaaaa oxfmaaaaa umpedjmaaaaaa overmaaaaaaa hetmaaaaaaaa azylmaaaaaaaaa ogdmaaaaaaaaaa"

Notes:

- S contains only uppercase, lowercase and spaces. Exactly one space between each word.
- 1 <= S.length <= 150.

830. Positions of Large Groups

DescriptionHintsSubmissionsDiscussSolution

In a string S of lowercase letters, these letters form consecutive groups of the same character.

For example, a string like S = "abbxxxxzyy" has the groups "a", "bb", "xxxx", "z" and "yy".

Call a group *large* if it has 3 or more characters. We would like the starting and ending positions of every large group.

The final answer should be in lexicographic order.

Example 1:

Input: "abbxxxxzzy"
Output: [[3,6]]
Explanation: "xxxx" is the single large group with starting 3 and ending positions 6.

Example 2:

Input: "abc"
Output: []
Explanation: We have "a","b" and "c" but no large group.

Example 3:

Input: "abcdddeeeeaabbbcd"
Output: [[3,5],[6,9],[12,14]]

Note: 1 <= S.length <= 1000

832. Flipping an Image

DescriptionHintsSubmissionsDiscussSolution

Given a binary matrix A, we want to flip the image horizontally, then invert it, and return the resulting image.

To flip an image horizontally means that each row of the image is reversed. For example, flipping [1, 1, 0] horizontally results in [0, 1, 1].

To invert an image means that each 0 is replaced by 1, and each 1 is replaced by 0. For example, inverting [0, 1, 1] results in [1, 0, 0].

Example 1:

Input: [[1,1,0],[1,0,1],[0,0,0]]
Output: [[1,0,0],[0,1,0],[1,1,1]]
Explanation: First reverse each row: [[0,1,1],[1,0,1],[0,0,0]].
Then, invert the image: [[1,0,0],[0,1,0],[1,1,1]]

Example 2:

Input: [[1,1,0,0],[1,0,0,1],[0,1,1,1],[1,0,1,0]]
Output: [[1,1,0,0],[0,1,1,0],[0,0,0,1],[1,0,1,0]]
Explanation: First reverse each row:
[[0,0,1,1],[1,0,0,1],[1,1,1,0],[0,1,0,1]].
Then invert the image: [[1,1,0,0],[0,1,1,0],[0,0,0,1],[1,0,1,0]]

Notes:

- 1 <= A.length = A[0].length <= 20
- 0 <= A[i][j] <= 1

---

836. Rectangle Overlap

DescriptionHintsSubmissionsDiscussSolution

A rectangle is represented as a list [x1, y1, x2, y2], where (x1, y1) are the coordinates of its bottom-left corner, and (x2, y2) are the coordinates of its top-right corner.

Two rectangles overlap if the area of their intersection is positive. To be clear, two rectangles that only touch at the corner or edges do not overlap.

Given two (axis-aligned) rectangles, return whether they overlap.

Example 1:

Input: rec1 = [0,0,2,2], rec2 = [1,1,3,3]

Output: true

Example 2:

Input: rec1 = [0,0,1,1], rec2 = [1,0,2,1]
Output: false

Notes:

1. Both rectangles `rec1` and `rec2` are lists of 4 integers.
2. All coordinates in rectangles will be between `-10^9` and `10^9`.

840. Magic Squares In Grid

DescriptionHintsSubmissionsDiscussSolution

A 3 x 3 magic square is a 3 x 3 grid filled with distinct numbers **from 1 to 9** such that each row, column, and both diagonals all have the same sum.

Given an grid of integers, how many 3 x 3 "magic square" subgrids are there? (Each subgrid is contiguous).

Example 1:

Input: [[4,3,8,4],
[9,5,1,9],
[2,7,6,2]]
Output: 1
Explanation:
The following subgrid is a 3 x 3 magic square:
438
951
276

while this one is not:
384
519
762

In total, there is only one magic square inside the given grid.

Note:

1. 1 <= grid.length <= 10
2. 1 <= grid[0].length <= 10
3. 0 <= grid[i][j] <= 15

---

844. Backspace String Compare

DescriptionHintsSubmissionsDiscussSolution

Given two strings S and T, return if they are equal when both are typed into empty text editors. # means a backspace character.

Example 1:

Input: S = "ab#c", T = "ad#c"
Output: true
Explanation: Both S and T become "ac".

Example 2:

Input: S = "ab##", T = "c#d#"
Output: true
Explanation: Both S and T become "".

Example 3:

Input: S = "a##c", T = "#a#c"
Output: true
Explanation: Both S and T become "c".

Example 4:

Input: S = "a#c", T = "b"
Output: false
Explanation: S becomes "c" while T becomes "b".

Note:

1. 1 <= S.length <= 200
2. 1 <= T.length <= 200

3. S and T only contain lowercase letters and '#' characters.

Follow up:

- Can you solve it in O(N) time and O(1) space?

849. Maximize Distance to Closest Person

DescriptionHintsSubmissionsDiscussSolution

In a row of seats, 1 represents a person sitting in that seat, and 0 represents that the seat is empty.

There is at least one empty seat, and at least one person sitting.

Alex wants to sit in the seat such that the distance between him and the closest person to him is maximized.

Return that maximum distance to closest person.

Example 1:

Input: [1,0,0,0,1,0,1]
Output: 2
Explanation:
If Alex sits in the second open seat (seats[2]), then the closest person has distance 2.
If Alex sits in any other open seat, the closest person has distance 1.
Thus, the maximum distance to the closest person is 2.

Example 2:

Input: [1,0,0,0]
Output: 3
Explanation:
If Alex sits in the last seat, the closest person is 3 seats away.
This is the maximum distance possible, so the answer is 3.

Note:

1. 1 <= seats.length <= 20000
2. seats contains only 0s or 1s, at least one 0, and at least one 1.

852. Peak Index in a Mountain Array

DescriptionHintsSubmissionsDiscussSolution

Let's call an array A a *mountain* if the following properties hold:

- A.length >= 3
- There exists some 0 < i < A.length - 1 such that A[0] < A[1] < ... A[i-1] < A[i] > A[i+1] > ... > A[A.length - 1]

Given an array that is definitely a mountain, return any i such that A[0] < A[1] < ... A[i-1] < A[i] > A[i+1] > ... > A[A.length - 1].

Example 1:

Input: [0,1,0]
Output: 1

Example 2:

Input: [0,2,1,0]
Output: 1

Note:

1. 3 <= A.length <= 10000
2. 0 <= A[i] <= 10^6
3. A is a mountain, as defined above.

859. Buddy Strings

DescriptionHintsSubmissionsDiscussSolution

Given two strings A and B of lowercase letters, return true if and only if we can swap two letters in A so that the result equals B.

Example 1:

```
Input: A = "ab", B = "ba"
Output: true
```

Example 2:

```
Input: A = "ab", B = "ab"
Output: false
```

Example 3:

```
Input: A = "aa", B = "aa"
Output: true
```

Example 4:

```
Input: A = "aaaaaaabc", B = "aaaaaaacb"
Output: true
```

Example 5:

```
Input: A = "", B = "aa"
Output: false
```

Note:

1. 0 <= A.length <= 20000
2. 0 <= B.length <= 20000
3. A and B consist only of lowercase letters.

860. Lemonade Change

DescriptionHintsSubmissionsDiscussSolution

At a lemonade stand, each lemonade costs $5.

Customers are standing in a queue to buy from you, and order one at a time (in the order specified by bills).

Each customer will only buy one lemonade and pay with either a $5, $10, or $20 bill. You must provide the correct change to each customer, so that the net transaction is that the customer pays $5.

Note that you don't have any change in hand at first.

Return true if and only if you can provide every customer with correct change.

Example 1:

Input: [5,5,5,10,20]
Output: true
Explanation:
From the first 3 customers, we collect three $5 bills in order.
From the fourth customer, we collect a $10 bill and give back a $5.
From the fifth customer, we give a $10 bill and a $5 bill.
Since all customers got correct change, we output true.

Example 2:

Input: [5,5,10]
Output: true

Example 3:

Input: [10,10]
Output: false

Example 4:

Input: [5,5,10,10,20]
Output: false
Explanation:
From the first two customers in order, we collect two $5 bills.
For the next two customers in order, we collect a $10 bill and give back a $5 bill.
For the last customer, we can't give change of $15 back because we only have two $10 bills.
Since not every customer received correct change, the answer is false.

Note:

- 0 <= bills.length <= 10000
- bills[i] will be either 5, 10, or 20.

867. Transpose Matrix

DescriptionHintsSubmissionsDiscussSolution

Given a matrix A, return the transpose of A.

The transpose of a matrix is the matrix flipped over it's main diagonal, switching the row and column indices of the matrix.

Example 1:

Input: [[1,2,3],[4,5,6],[7,8,9]]
Output: [[1,4,7],[2,5,8],[3,6,9]]

Example 2:

Input: [[1,2,3],[4,5,6]]
Output: [[1,4],[2,5],[3,6]]

Note:

1. 1 <= A.length <= 1000
2. 1 <= A[0].length <= 1000

868. Binary Gap

DescriptionHintsSubmissionsDiscussSolution

Given a positive integer N, find and return the longest distance between two consecutive 1's in the binary representation of N.

If there aren't two consecutive 1's, return 0.

Example 1:

Input: 22
Output: 2
Explanation:
22 in binary is 0b10110.
In the binary representation of 22, there are three ones, and two consecutive pairs of 1's.
The first consecutive pair of 1's have distance 2.
The second consecutive pair of 1's have distance 1.
The answer is the largest of these two distances, which is 2.

Example 2:

Input: 5
Output: 2
Explanation:
5 in binary is 0b101.

Example 3:

Input: 6
Output: 1
Explanation:
6 in binary is 0b110.

Example 4:

Input: 8
Output: 0
Explanation:
8 in binary is 0b1000.
There aren't any consecutive pairs of 1's in the binary representation of 8, so we return 0.

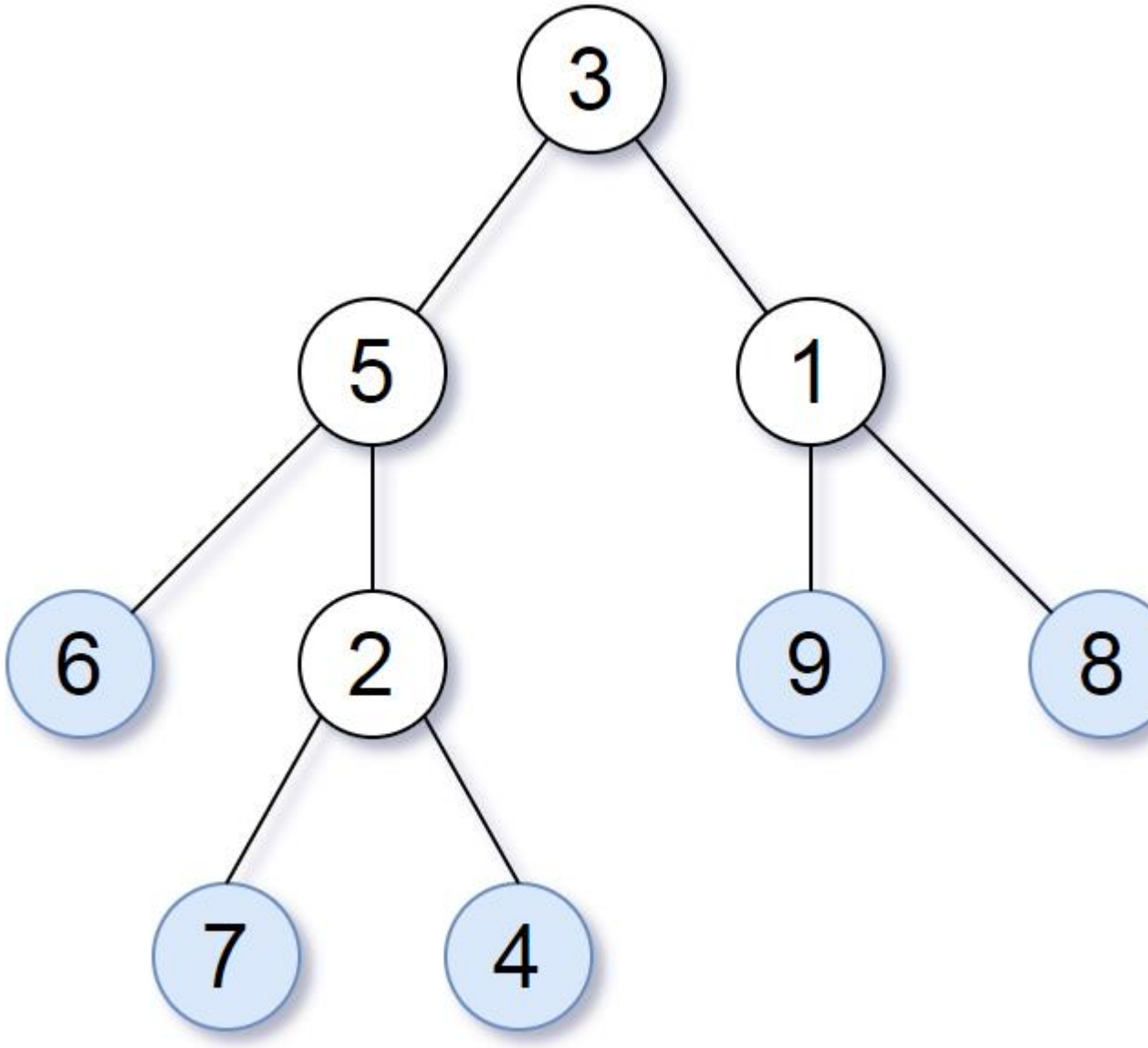
Note:

- 1 <= N <= 10^9

872. Leaf-Similar Trees

DescriptionHintsSubmissionsDiscussSolution

Consider all the leaves of a binary tree. From left to right order, the values of those leaves form a *leaf value sequence.*



For example, in the given tree above, the leaf value sequence is (6, 7, 4, 9, 8).

Two binary trees are considered *leaf-similar* if their leaf value sequence is the same.

Return true if and only if the two given trees with head nodes root1 and root2 are leaf-similar.

Note:

- Both of the given trees will have between 1 and 100 nodes.

874. Walking Robot Simulation

DescriptionHintsSubmissionsDiscussSolution

A robot on an infinite grid starts at point (0, 0) and faces north. The robot can receive one of three possible types of commands:

- -2: turn left 90 degrees
- -1: turn right 90 degrees
- 1 <= x <= 9: move forward x units

Some of the grid squares are obstacles.

The i-th obstacle is at grid point (obstacles[i][0], obstacles[i][1])

If the robot would try to move onto them, the robot stays on the previous grid square instead (but still continues following the rest of the route.)

Return the **square** of the maximum Euclidean distance that the robot will be from the origin.

Example 1:

Input: commands = [4,-1,3], obstacles = []
Output: 25
Explanation: robot will go to (3, 4)

Example 2:

Input: commands = [4,-1,4,-2,4], obstacles = [[2,4]]
Output: 65
Explanation: robot will be stuck at (1, 4) before turning left and going to (1, 8)

Note:

1. 0 <= commands.length <= 10000
2. 0 <= obstacles.length <= 10000
3. -30000 <= obstacle[i][0] <= 30000
4. -30000 <= obstacle[i][1] <= 30000
5. The answer is guaranteed to be less than 2 ^ 31.

876. Middle of the Linked List

DescriptionHintsSubmissionsDiscussSolution

Given a non-empty, singly linked list with head node head, return a middle node of linked list.

If there are two middle nodes, return the second middle node.

Example 1:

Input: [1,2,3,4,5]
Output: Node 3 from this list (Serialization: [3,4,5])
The returned node has value 3. (The judge's serialization of this node is [3,4,5]).
Note that we returned a ListNode object ans, such that:
ans.val = 3, ans.next.val = 4, ans.next.next.val = 5, and ans.next.next.next = NULL.

Example 2:

Input: [1,2,3,4,5,6]
Output: Node 4 from this list (Serialization: [4,5,6])
Since the list has two middle nodes with values 3 and 4, we return the second one.

Note:

- The number of nodes in the given list will be between 1 and 100.

883. Projection Area of 3D Shapes

DescriptionHintsSubmissionsDiscussSolution

On a N * N grid, we place some 1 * 1 * 1 cubes that are axis-aligned with the x, y, and z axes.

Each value v = grid[i][j] represents a tower of v cubes placed on top of grid cell (i, j).

Now we view the *projection* of these cubes onto the xy, yz, and zx planes.

A projection is like a shadow, that maps our 3 dimensional figure to a 2 dimensional plane.

Here, we are viewing the "shadow" when looking at the cubes from the top, the front, and the side.

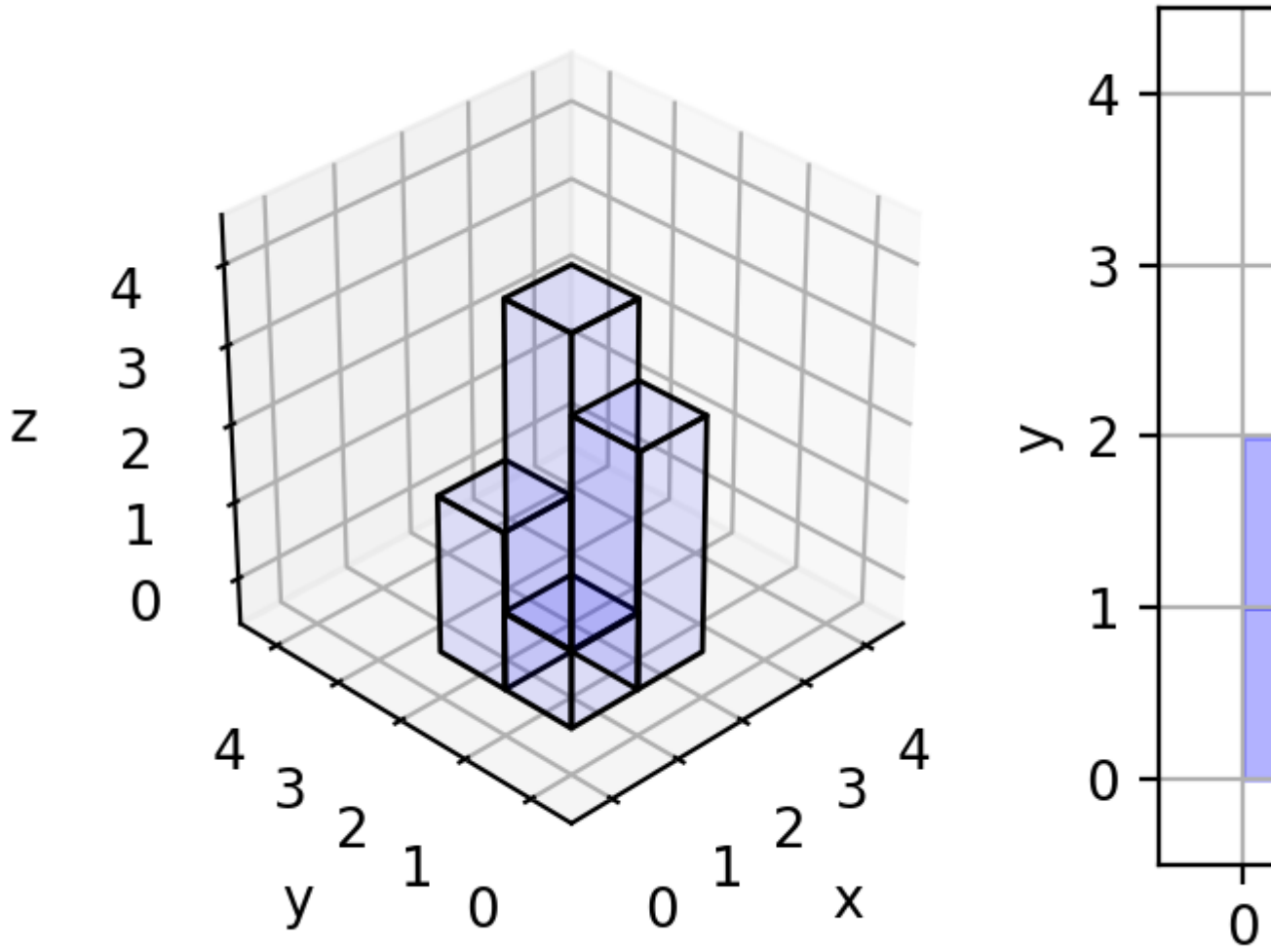
Return the total area of all three projections.

Example 1:

Input: [[2]]
Output: 5

Example 2:

Input: [[1,2],[3,4]]
Output: 17
Explanation:
Here are the three projections ("shadows") of the shape made with each axis-aligned plane.



Example 3:

Input: [[1,0],[0,2]]
Output: 8

Example 4:

Input: [[1,1,1],[1,0,1],[1,1,1]]
Output: 14

Example 5:

Input: [[2,2,2],[2,1,2],[2,2,2]]
Output: 21

Note:

- 1 <= grid.length = grid[0].length <= 50
- 0 <= grid[i][j] <= 50

884. Uncommon Words from Two Sentences

DescriptionHintsSubmissionsDiscussSolution

We are given two sentences A and B. (A *sentence* is a string of space separated words. Each *word* consists only of lowercase letters.)

A word is *uncommon* if it appears exactly once in one of the sentences, and does not appear in the other sentence.

Return a list of all uncommon words.

You may return the list in any order.

Example 1:

Input: A = "this apple is sweet", B = "this apple is sour"
Output: ["sweet","sour"]

Example 2:

Input: A = "apple apple", B = "banana"
Output: ["banana"]

Note:

1. 0 <= A.length <= 200
2. 0 <= B.length <= 200
3. A and B both contain only spaces and lowercase letters.

888. Fair Candy Swap

DescriptionHintsSubmissionsDiscussSolution

Alice and Bob have candy bars of different sizes: A[i] is the size of the i-th bar of candy that Alice has, and B[j] is the size of the j-th bar of candy that Bob has.

Since they are friends, they would like to exchange one candy bar each so that after the exchange, they both have the same total amount of candy. (*The total amount of candy a person has is the sum of the sizes of candy bars they have.*)

Return an integer array ans where ans[0] is the size of the candy bar that Alice must exchange, and ans[1] is the size of the candy bar that Bob must exchange.

If there are multiple answers, you may return any one of them. It is guaranteed an answer exists.

Example 1:

Input: A = [1,1], B = [2,2]
Output: [1,2]

Example 2:

Input: A = [1,2], B = [2,3]
Output: [1,2]

Example 3:

Input: A = [2], B = [1,3]
Output: [2,3]

Example 4:

Input: A = [1,2,5], B = [2,4]
Output: [5,4]

Note:

- 1 <= A.length <= 10000
- 1 <= B.length <= 10000
- 1 <= A[i] <= 100000
- 1 <= B[i] <= 100000
- It is guaranteed that Alice and Bob have different total amounts of candy.
- It is guaranteed there exists an answer.

892. Surface Area of 3D Shapes

DescriptionHintsSubmissionsDiscussSolution

On a N * N grid, we place some 1 * 1 * 1 cubes.

Each value v = grid[i][j] represents a tower of v cubes placed on top of grid cell (i, j).

Return the total surface area of the resulting shapes.

Example 1:

```
Input: [[2]]
Output: 10
```

Example 2:

```
Input: [[1,2],[3,4]]
Output: 34
```

Example 3:

```
Input: [[1,0],[0,2]]
Output: 16
```

Example 4:

```
Input: [[1,1,1],[1,0,1],[1,1,1]]
Output: 32
```

Example 5:

Input: [[2,2,2],[2,1,2],[2,2,2]]
Output: 46

Note:

- 1 <= N <= 50
- 0 <= grid[i][j] <= 50

893. Groups of Special-Equivalent Strings

DescriptionHintsSubmissionsDiscussSolution

You are given an array A of strings.

Two strings S and T are *special-equivalent* if after any number of *moves*, S == T.

A *move* consists of choosing two indices i and j with i % 2 == j % 2, and swapping S[i] with S[j].

Now, a *group of special-equivalent strings from A* is a non-empty subset S of A such that any string not in S is not special-equivalent with any string in S.

Return the number of groups of special-equivalent strings from A.

Example 1:

Input: ["a","b","c","a","c","c"]
Output: 3
Explanation: 3 groups ["a","a"], ["b"], ["c","c","c"]

Example 2:

Input: ["aa","bb","ab","ba"]
Output: 4
Explanation: 4 groups ["aa"], ["bb"], ["ab"], ["ba"]

Example 3:

```
Input: ["abc","acb","bac","bca","cab","cba"]
Output: 3
Explanation: 3 groups ["abc","cba"], ["acb","bca"], ["bac","cab"]
```

Example 4:

```
Input: ["abcd","cdab","adcb","cbad"]
Output: 1
Explanation: 1 group ["abcd","cdab","adcb","cbad"]
```

Note:

- `1 <= A.length <= 1000`
- `1 <= A[i].length <= 20`
- All `A[i]` have the same length.
- All `A[i]` consist of only lowercase letters.

896. Monotonic Array

DescriptionHintsSubmissionsDiscussSolution

An array is *monotonic* if it is either monotone increasing or monotone decreasing.

An array A is monotone increasing if for all i <= j, A[i] <= A[j]. An array A is monotone decreasing if for all i <= j, A[i] >= A[j].

Return true if and only if the given array A is monotonic.

Example 1:

```
Input: [1,2,2,3]
Output: true
```

Example 2:

```
Input: [6,5,4,4]
Output: true
```

Example 3:

```
Input: [1,3,2]
Output: false
```

Example 4:

```
Input: [1,2,4,5]
Output: true
```

Example 5:

```
Input: [1,1,1]
Output: true
```

Note:

1. 1 <= A.length <= 50000
2. -100000 <= A[i] <= 100000

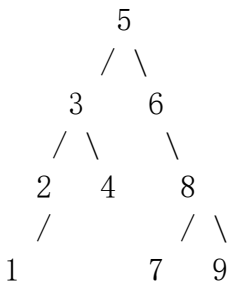
897. Increasing Order Search Tree

DescriptionHintsSubmissionsDiscussSolution

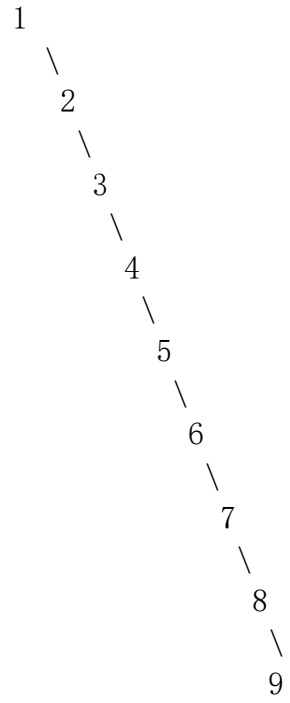
Given a tree, rearrange the tree in **in-order** so that the leftmost node in the tree is now the root of the tree, and every node has no left child and only 1 right child.

Example 1:

```
Input: [5,3,6,2,4,null,8,1,null,null,null,7,9]
```



```
Output: [1,null,2,null,3,null,4,null,5,null,6,null,7,null,8,null,9]
```



Note:

1. The number of nodes in the given tree will be between 1 and 100.
2. Each node will have a unique integer value from 0 to 1000.

905. Sort Array By Parity

DescriptionHintsSubmissionsDiscussSolution

Given an array A of non-negative integers, return an array consisting of all the even elements of A, followed by all the odd elements of A.

You may return any answer array that satisfies this condition.

Example 1:

Input: [3,1,2,4]
Output: [2,4,3,1]
The outputs [4,2,3,1], [2,4,1,3], and [4,2,1,3] would also be accepted.

Note:

1. 1 <= A.length <= 5000
2. 0 <= A[i] <= 5000

908. Smallest Range I

DescriptionHintsSubmissionsDiscussSolution

Given an array A of integers, for each integer A[i] we may choose any x with -K <= x <= K, and add x to A[i].

After this process, we have some array B.

Return the smallest possible difference between the maximum value of B and the minimum value of B.

Example 1:

Input: A = [1], K = 0
Output: 0
Explanation: B = [1]

Example 2:

Input: A = [0,10], K = 2
Output: 6
Explanation: B = [2,8]

Example 3:

Input: A = [1,3,6], K = 3
Output: 0
Explanation: B = [3,3,3] or B = [4,4,4]

Note:

1. 1 <= A.length <= 10000
2. 0 <= A[i] <= 10000

3. 0 <= K <= 10000

---

910. Smallest Range II

DescriptionHintsSubmissionsDiscussSolution

Given an array A of integers, for each integer A[i] we need to choose **either x = -K or x = K**, and add x to A[i] **(only once)**.

After this process, we have some array B.

Return the smallest possible difference between the maximum value of B and the minimum value of B.

Example 1:

```
Input: A = [1], K = 0
Output: 0
Explanation: B = [1]
```

Example 2:

```
Input: A = [0,10], K = 2
Output: 6
Explanation: B = [2,8]
```

Example 3:

```
Input: A = [1,3,6], K = 3
Output: 3
Explanation: B = [4,6,3]
```

Note:

1. 1 <= A.length <= 10000
2. 0 <= A[i] <= 10000
3. 0 <= K <= 10000

914. X of a Kind in a Deck of Cards

DescriptionHintsSubmissionsDiscussSolution

In a deck of cards, each card has an integer written on it.

Return true if and only if you can choose X >= 2 such that it is possible to split the entire deck into 1 or more groups of cards, where:

- Each group has exactly X cards.
- All the cards in each group have the same integer.

Example 1:

Input: [1,2,3,4,4,3,2,1]
Output: true
Explanation: Possible partition [1,1],[2,2],[3,3],[4,4]

Example 2:

Input: [1,1,1,2,2,2,3,3]
Output: false
Explanation: No possible partition.

Example 3:

Input: [1]
Output: false
Explanation: No possible partition.

Example 4:

Input: [1,1]
Output: true
Explanation: Possible partition [1,1]

Example 5:

Input: [1,1,2,2,2,2]
Output: true

Explanation: Possible partition [1,1],[2,2],[2,2]

Note:

1. 1 <= deck.length <= 10000
2. 0 <= deck[i] < 10000

917. Reverse Only Letters

DescriptionHintsSubmissionsDiscussSolution

Given a string S, return the "reversed" string where all characters that are not a letter stay in the same place, and all letters reverse their positions.

Example 1:

Input: "ab-cd"
Output: "dc-ba"

Example 2:

Input: "a-bC-dEf-ghIj"
Output: "j-Ih-gfE-dCba"

Example 3:

Input: "Test1ng-Leet=code-Q!"
Output: "Qedo1ct-eeLg=ntse-T!"

Note:

1. S.length <= 100
2. 33 <= S[i].ASCIIcode <= 122
3. S doesn't contain \ or "

918. Maximum Sum Circular Subarray

分析

难度 中
来源
博客

题目

Given a **circular array C** of integers represented by A, find the maximum possible sum of a non-empty subarray of **C**.

Here, a *circular array* means the end of the array connects to the beginning of the array. (Formally, C[i] = A[i] when 0 <= i < A.length, and C[i+A.length] = C[i] when i >= 0.)

Also, a subarray may only include each element of the fixed buffer A at most once. (Formally, for a subarray C[i], C[i+1], ..., C[j], there does not exist i <= k1, k2 <= j with k1 % A.length = k2 % A.length.)

Example 1:

Input: [1,-2,3,-2]
Output: 3
Explanation: Subarray [3] has maximum sum 3

Example 2:

Input: [5,-3,5]
Output: 10
Explanation: Subarray [5,5] has maximum sum 5 + 5 = 10

Example 3:

Input: [3,-1,2,-1]
Output: 4
Explanation: Subarray [2,-1,3] has maximum sum 2 + (-1) + 3 = 4

Example 4:

```
Input: [3,-2,2,-3]
Output: 3
Explanation:  Subarray [3] and [3,-2,2] both have maximum sum 3

Example 5:

Input: [-2,-3,-1]
Output: -1
Explanation:  Subarray [-1] has maximum sum -1
```

Note:

1. -30000 <= A[i] <= 30000
2. 1 <= A.length <= 30000

919. Complete Binary Tree Inserter

分析

难度 中等
来源
https://leetcode.com/problems/complete-binary-tree-inserter/
博客

题目

A *complete* binary tree is a binary tree in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible.

Write a data structure CBTInserter that is initialized with a complete binary tree and supports the following operations:

- CBTInserter(TreeNode root) initializes the data structure on a given tree with head node root;

- CBTInserter.insert(int v) will insert a TreeNode into the tree with value node.val = v so that the tree remains complete, and returns the value of the parent of the inserted TreeNode;
- CBTInserter.get_root() will return the head node of the tree.

Example 1:

Input: inputs = ["CBTInserter","insert","get_root"], inputs = [[[1]],[2],[]]
Output: [null,1,[1,2]]

Example 2:

Input: inputs = ["CBTInserter","insert","insert","get_root"], inputs = [[[1,2,3,4,5,6]],[7],[8],[]]
Output: [null,3,4,[1,2,3,4,5,6,7,8]]

Note:

1. The initial given tree is complete and contains between 1 and 1000 nodes.
2. CBTInserter.insert is called at most 10000 times per test case.
3. Every value of a given or inserted node is between 0 and 5000.

921. Minimum Add to Make Parentheses Valid

分析

难度 中
来源
https://leetcode.com/problems/minimum-add-to-make-parentheses-valid/
博客

题目

Given a string S of '(' and ')' parentheses, we add the minimum number of parentheses ('(' or ')', and in any positions) so that the resulting parentheses string is valid.

Formally, a parentheses string is valid if and only if:

- It is the empty string, or
- It can be written as AB (A concatenated with B), where A and B are valid strings, or
- It can be written as (A), where A is a valid string.

Given a parentheses string, return the minimum number of parentheses we must add to make the resulting string valid.

Example 1:

Input: "())"
Output: 1

Example 2:

Input: "((("
Output: 3

Example 3:

Input: "()"
Output: 0

Example 4:

Input: "()))(("
Output: 4

Note:

1. S.length <= 1000
2. S only consists of '(' and ')' characters.

922. Sort Array By Parity II

分析

难度 易

来源

https://leetcode.com/problems/sort-array-by-parity-ii/

博客

题目

Given an array A of non-negative integers, half of the integers in A are odd, and half of the integers are even.

Sort the array so that whenever A[i] is odd, i is odd; and whenever A[i] is even, i is even.

You may return any answer array that satisfies this condition.

Example 1:

Input: [4,2,5,7]
Output: [4,5,2,7]
Explanation: [4,7,2,5], [2,5,4,7], [2,7,4,5] would also have been accepted.

Note:

1. 2 <= A.length <= 20000
2. A.length % 2 == 0
3. 0 <= A[i] <= 1000

923. 3Sum With Multiplicity

分析

难度 中

来源

https://leetcode.com/problems/3sum-with-multiplicity/

博客

题目

Given an integer array A, and an integer target, return the number of tuples i, j, k such that i < j < k and A[i] + A[j] + A[k] == target.

As the answer can be very large, return it modulo 10^9 + 7.

Example 1:

Input: A = [1,1,2,2,3,3,4,4,5,5], target = 8
Output: 20
Explanation:
Enumerating by the values (A[i], A[j], A[k]):
(1, 2, 5) occurs 8 times;
(1, 3, 4) occurs 8 times;
(2, 2, 4) occurs 2 times;
(2, 3, 3) occurs 2 times.

Example 2:

Input: A = [1,1,2,2,2,2], target = 5
Output: 12
Explanation:
A[i] = 1, A[j] = A[k] = 2 occurs 12 times:
We choose one 1 from [1,1] in 2 ways,
and two 2s from [2,2,2,2] in 6 ways.

Note:

1. 3 <= A.length <= 3000
2. 0 <= A[i] <= 100
3. 0 <= target <= 300

925. Long Pressed Name

分析

难度 易

来源

https://leetcode.com/problems/long-pressed-name/

博客

题目

Your friend is typing his name into a keyboard. Sometimes, when typing a character c, the key might get *long pressed*, and the character will be typed 1 or more times.

You examine the typed characters of the keyboard. Return True if it is possible that it was your friends name, with some characters (possibly none) being long pressed.

Example 1:

Input: name = "alex", typed = "aaleex"
Output: true
Explanation: 'a' and 'e' in 'alex' were long pressed.

Example 2:

Input: name = "saeed", typed = "ssaaedd"
Output: false
Explanation: 'e' must have been pressed twice, but it wasn't in the typed output.

Example 3:

Input: name = "leelee", typed = "lleeelee"
Output: true

Example 4:

Input: name = "laiden", typed = "laiden"
Output: true
Explanation: It's not necessary to long press any character.

Note:

1. name.length <= 1000
2. typed.length <= 1000
3. The characters of name and typed are lowercase letters.

926. Flip String to Monotone Increasing

分析

难度 中

来源

https://leetcode.com/problems/flip-string-to-monotone-increasing/

博客

题目

A string of '0's and '1's is *monotone increasing* if it consists of some number of '0's (possibly 0), followed by some number of '1's (also possibly 0.)

We are given a string S of '0's and '1's, and we may flip any '0' to a '1' or a '1' to a '0'.

Return the minimum number of flips to make S monotone increasing.

Example 1:

Input: "00110"
Output: 1
Explanation: We flip the last digit to get 00111.

Example 2:

Input: "010110"
Output: 2
Explanation: We flip to get 011111, or alternatively 000111.

Example 3:

Input: "00011000"
Output: 2
Explanation: We flip to get 00000000.

Note:

1. 1 <= S.length <= 20000
2. S only consists of '0' and '1' characters.

929. Unique Email Addresses

分析

难度 易
来源
https://leetcode.com/problems/unique-email-addresses/

博客

题目

Every email consists of a local name and a domain name, separated by the @ sign.

For example, in alice@leetcode.com, alice is the local name, and leetcode.com is the domain name.

Besides lowercase letters, these emails may contain '.'s or '+'s.

If you add periods ('.') between some characters in the **local name** part of an email address, mail sent there will be forwarded to the same address without dots in the local name. For example, "alice.z@leetcode.com" and "alicez@leetcode.com" forward to the same email address. (Note that this rule does not apply for domain names.)

If you add a plus ('+') in the **local name**, everything after the first plus sign will be **ignored**. This allows certain emails to be filtered, for example m.y+name@email.com will be forwarded to my@email.com. (Again, this rule does not apply for domain names.)

It is possible to use both of these rules at the same time.

Given a list of emails, we send one email to each address in the list. How many different addresses actually receive mails?

Example 1:

Input:
["test.email+alex@leetcode.com","test.e.mail+bob.cathy@leetcode.com",
"testemail+david@lee.tcode.com"]
Output: 2
Explanation: "testemail@leetcode.com" and "testemail@lee.tcode.com"
actually receive mails

Note:

- 1 <= emails[i].length <= 100
- 1 <= emails.length <= 100
- Each emails[i] contains exactly one '@' character.

930. Binary Subarrays With Sum

分析

难度 中
来源
https://leetcode.com/problems/binary-subarrays-with-sum/
博客

题目

In an array A of 0s and 1s, how many **non-empty** subarrays have sum S?

Example 1:

Input: A = [1,0,1,0,1], S = 2
Output: 4
Explanation:
The 4 subarrays are bolded below:
[**1,0,1**,0,1]
[**1,0,1,0**,1]
[1,**0,1,0,1**]
[1,0,**1,0,1**]

Note:

1. A.length <= 30000
2. 0 <= S <= A.length
3. A[i] is either 0 or 1.

931. Minimum Falling Path Sum

分析

难度 中

来源

https://leetcode.com/problems/minimum-falling-path-sum/

博客

题目

Given a **square** array of integers A, we want the **minimum** sum of a *falling path* through A.

A falling path starts at any element in the first row, and chooses one element from each row. The next row's choice must be in a column that is different from the previous row's column by at most one.

Example 1:

Input: [[1,2,3],[4,5,6],[7,8,9]]
Output: 12
Explanation:
The possible falling paths are:

- [1,4,7], [1,4,8], [1,5,7], [1,5,8], [1,5,9]
- [2,4,7], [2,4,8], [2,5,7], [2,5,8], [2,5,9], [2,6,8], [2,6,9]
- [3,5,7], [3,5,8], [3,5,9], [3,6,8], [3,6,9]

The falling path with the smallest sum is [1,4,7], so the answer is 12.

Note:

1. 1 <= A.length == A[0].length <= 100
2. -100 <= A[i][j] <= 100