### 字符串操作

#### M49. Group Anagrams

|  |
| --- |
| **M49. Group Anagrams** |
| Given an array of strings, group anagrams together.  **For example,** given: ["eat", "tea", "tan", "ate", "nat", "bat"],  **Return:**  [  ["ate", "eat","tea"],  ["nat","tan"],  ["bat"]  ]  **Note:** All inputs will be in lower-case.  **Tags：**Hash Table、String  **Similar Problems：**(E) Valid Anagram、(M) Group Shifted Strings |
| **Solution:** vector<vector<string>> groupAnagrams(vector<string>& strs); |

#### M91. Decode Ways

|  |
| --- |
| **M91. Decode Ways** |
| A message containing letters from A-Z is being encoded to numbers using the following mapping:  'A' -> 1  'B' -> 2  ...  'Z' -> 26  Given an encoded message containing digits, determine the total number of ways to decode it.  **For example,**  Given encoded message "12", it could be decoded as "AB" (1 2) or "L" (12).  The number of ways decoding "12" is 2.  **Tags：**Dynamic Programming、String |
| **Solution:** int numDecodings(string s); |

#### M93. Restore IP Addresses

|  |
| --- |
| **M93. Restore IP Addresses** |
| Given a string containing only digits, restore it by returning all possible valid IP address combinations.  **For example:**  Given "25525511135",  return ["255.255.11.135", "255.255.111.35"]. (Order does not matter)  **Tags：**Backtracking、String |
| **Solution:** vector<string> restoreIpAddresses(string s); |

#### M151. Reverse Words in a String

|  |
| --- |
| **M** **151. Reverse Words in a String** |
| Given an input string, reverse the string word by word.  **For example,**  Given s = "the sky is blue",  return "blue is sky the".  Update (2015-02-12):  For C programmers: Try to solve it in-place in O(1) space.  **Clarification:**   * What constitutes a word?   A sequence of non-space characters constitutes a word.   * Could the input string contain leading or trailing spaces?   Yes. However, your reversed string should not contain leading or trailing spaces.   * How about multiple spaces between two words?   Reduce them to a single space in the reversed string.  Tags：String  Similar Problems：(M) Reverse Words in a String II |
| **Solution:** |

#### M385. Mini Parser

|  |
| --- |
| **M** **385. Mini Parser** |
| Given a nested list of integers represented as a string, implement a parser to deserialize it.  Each element is either an integer, or a list -- whose elements may also be integers or other lists.  **Note:** You may assume that the string is well-formed:   * String is non-empty. * String does not contain white spaces. * String contains only digits 0-9, [, - ,, ].   **Example 1:**  Given s = "324",  You should return a NestedInteger object which contains a single integer 324.  **Example 2:**  Given s = "[123,[456,[789]]]",  Return a NestedInteger object containing a nested list with 2 elements:  1. An integer containing value 123.  2. A nested list containing two elements:  i. An integer containing value 456.  ii. A nested list with one element:  a. An integer containing value 789.  **Tags：**Stack String  **Similar Problems：**(M) Flatten Nested List Iterator (M) Ternary Expression Parser |
| **Solution:** NestedInteger deserialize(string s)； |

#### 组1

##### M8. String to Integer (atoi)

|  |
| --- |
| **M8. String to Integer (atoi)** |
| Implement atoi to convert a string to an integer.  **Hint:** Carefully consider all possible input cases. If you want a challenge, please do not see below and ask yourself what are the possible input cases.  **Notes:** It is intended for this problem to be specified vaguely (ie, no given input specs). You are responsible to gather all the input requirements up front.  **Update (2015-02-10):**  The signature of the C++ function had been updated. If you still see your function signature accepts a const char \* argument, please click the reload button to reset your code definition.  **Requirements for atoi:**  The function first discards as many whitespace characters as necessary until the first non-whitespace character is found. Then, starting from this character, takes an optional initial plus or minus sign followed by as many numerical digits as possible, and interprets them as a numerical value.  The string can contain additional characters after those that form the integral number, which are ignored and have no effect on the behavior of this function.  If the first sequence of non-whitespace characters in str is not a valid integral number, or if no such sequence exists because either str is empty or it contains only whitespace characters, no conversion is performed.  If no valid conversion could be performed, a zero value is returned. If the correct value is out of the range of representable values, INT\_MAX (2147483647) or INT\_MIN (-2147483648) is returned.  **Tags：**Math、String  **Similar Problems**：(E7) Reverse Integer、(H65) Valid Number |
| **Solution：**int myAtoi(string str); |

##### E7. Reverse Integer

|  |
| --- |
| **E7. 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 hold integers within the 32-bit signed integer range. For the purpose of this problem, assume that your function returns 0 when the reversed integer overflows.  **Tags：**Math  **Similar Problems**：(M8) String to Integer (atoi) |
| **Solution：**int reverse(int x); |

##### H65. Valid Number

|  |
| --- |
| **H65. Valid Number** |
| Validate if a given string is numeric.  **Some examples:**  "0" => true  " 0.1 " => true  "abc" => false  "1 a" => false  "2e10" => true  **Note:** It is intended for the problem statement to be ambiguous. You should gather all requirements up front before implementing one.  **Update (2015-02-10):**  The signature of the C++ function had been updated. If you still see your function signature accepts a const char \* argument, please click the reload button to reset your code definition.  **Tags：**Math, String  **Similar Problems**：(M8) String to Integer (atoi) |
| **Solution:** bool isNumber(string s) ; |

#### 组2

##### M5. Longest Palindromic Substring

|  |
| --- |
| **M5. Longest Palindromic Substring** |
| Given a string s, find the longest palindromic substring in s. You may assume that the maximum length of s is 1000.  **Example:**  Input: "babad"  Output: "bab"  Note: "aba" is also a valid answer.  **Example:**  Input: "cbbd"  Output: "bb"  **Tags：**String  **Similar Problems：**(H214) Shortest Palindrome、(E) Palindrome Permutation、 (H336) Palindrome Pairs、 (M516) Longest Palindromic Subsequence、(M647) Palindromic Substrings |
| **Solution:** string longestPalindrome(string s); |

##### H214. Shortest Palindrome

|  |
| --- |
| **H214. Shortest Palindrome** |
| 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.  **For example:**  Given "aacecaaa", return "aaacecaaa".  Given "abcd", return "dcbabcd".  **Tags：**String  **Similar Problems：**(M5) Longest Palindromic Substring、(E28) Implement strStr()、(H336) Palindrome Pairs |
| **Solution:** string shortestPalindrome(string s); |

##### E28. Implement strStr()

|  |
| --- |
| **E28. Implement strStr()** |
| 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  **Tags：**String、Two Pointers  **Similar Problems：**(H214) Shortest Palindrome、(E459) Repeated Substring Pattern |
| **Solution:** int strStr(string haystack, string needle); |

##### E459. Repeated Substring Pattern

|  |
| --- |
| **E459. Repeated Substring Pattern** |
| 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.)  **Tags：**String  **Similar Problems：**(E28) Implement strStr()、(E686) Repeated String Match |
| **Solution:** bool repeatedSubstringPattern(string s); |

##### E686. Repeated String Match

|  |
| --- |
| **E686. Repeated String Match** |
| 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.  **Tags：**String  **Similar Problems：**(E459) Repeated Substring Pattern |
| **Solution:** int repeatedStringMatch(string A, string B); |

##### H336. Palindrome Pairs

|  |
| --- |
| **H336. Palindrome Pairs** |
| Given a list of unique words, find all pairs of distinct indices (i, j) in the given list, so that the concatenation of the two words, i.e. words[i] + words[j] is a palindrome.  **Example 1:**  Given words = ["bat", "tab", "cat"]  Return [[0, 1], [1, 0]]  The palindromes are ["battab", "tabbat"]  **Example 2:**  Given words = ["abcd", "dcba", "lls", "s", "sssll"]  Return [[0, 1], [1, 0], [3, 2], [2, 4]]  The palindromes are ["dcbaabcd", "abcddcba", "slls", "llssssll"]  **Tags：**String、Hash Table、Trie  **Similar Problems：**(M5) Longest Palindromic Substring、(H214) Shortest Palindrome |
| **Solution:** vector<vector<int>> palindromePairs(vector<string>& words); |

##### M516. Longest Palindromic Subsequence

|  |
| --- |
| **M516. Longest Palindromic Subsequence** |
| Given a string s, find the longest palindromic subsequence's length in s. You may assume that the maximum length of s is 1000.  **Example 1:**  Input: "bbbab"  Output: 4  One possible longest palindromic subsequence is "bbbb".  **Example 2:**  Input: "cbbd"  Output: 2  One possible longest palindromic subsequence is "bb".  **Tags：**Dynamic Programming  **Similar Problems：(**M5) Longest Palindromic Substring、(M647) Palindromic Substrings、(H730) Count Different Palindromic Subsequences |
| **Solution:** int longestPalindromeSubseq(string s); |

##### H730. Count Different Palindromic Subsequences

|  |
| --- |
| **H730. Count Different Palindromic Subsequences** |
| Given a string S, find the number of different non-empty palindromic subsequences in S, and return that number modulo 10^9 + 7.  A subsequence of a string S is obtained by deleting 0 or more characters from S.  A sequence is palindromic if it is equal to the sequence reversed.  Two sequences A\_1, A\_2, ... and B\_1, B\_2, ... are different if there is some i for which A\_i != B\_i.  **Example 1:**  Input: S = 'bccb'  Output: 6  **Explanation:**  The 6 different non-empty palindromic subsequences are 'b', 'c', 'bb', 'cc', 'bcb', 'bccb'.  **Note that** 'bcb' is counted only once, even though it occurs twice.  **Example 2:**  Input: S = 'abcdabcdabcdabcdabcdabcdabcdabcddcbadcbadcbadcbadcbadcbadcbadcba'  Output: 104860361  **Explanation:**  There are 3104860382 different non-empty palindromic subsequences, which is 104860361 modulo 10^9 + 7.  **Note:**  The length of S will be in the range [1, 1000].  Each character S[i] will be in the set {'a', 'b', 'c', 'd'}.  **Tags：**String、Dynamic Programming  **Similar Problems：**(M516) Longest Palindromic Subsequence |
| **Solution:** int countPalindromicSubsequences(string S); |

##### M647. Palindromic Substrings

|  |
| --- |
| **M647. Palindromic Substrings** |
| Given a string, your task is to count how many palindromic substrings in this string.  The substrings with different start indexes or end indexes are counted as different substrings even they consist of same characters.  **Example 1:**  Input: "abc"  Output: 3  **Explanation:** Three palindromic strings: "a", "b", "c".  **Example 2:**  Input: "aaa"  Output: 6  **Explanation:** Six palindromic strings: "a", "a", "a", "aa", "aa", "aaa".  **Note:**  The input string length won't exceed 1000.  **Tags：**String、Dynamic Programming  **Similar Problems：**(M5) Longest Palindromic Substring、(M516) Longest Palindromic Subsequence |
| **Solution:** int countSubstrings(string s); |

#### 组3

##### M12. Integer to Roman

|  |
| --- |
| **M12. Integer to Roman** |
| Given an integer, convert it to a roman numeral.  Input is guaranteed to be within the range from 1 to 3999.  **Tags：**Math、String  **Similar Problems：** (E13) Roman to Integer 、(H273) Integer to English Words |
| **Solution:** string intToRoman(int num); |

##### E13. Roman to Integer

|  |
| --- |
| **E13. Roman to Integer** |
| Given a roman numeral, convert it to an integer.  Input is guaranteed to be within the range from 1 to 3999.  **Tags：**String、Math  **Similar Problems：**(M12) Integer to Roman |
| **Solution:** int romanToInt(string s); |

##### H273. Integer to English Words

|  |
| --- |
| **H273. Integer to English Words** |
| Convert a non-negative integer to its english words representation. Given input is guaranteed to be less than 231 - 1.  **For example,**  123 -> "One Hundred Twenty Three"  12345 -> "Twelve Thousand Three Hundred Forty Five"  1234567 -> "One Million Two Hundred Thirty Four Thousand Five Hundred Sixty Seven"  **Tags：**String、Math  **Similar Problems：**(M12) Integer to Roman |
| **Solution:** string numberToWords(int num); |

#### 组4

##### M43. Multiply Strings

|  |
| --- |
| **M43. Multiply Strings** |
| Given two non-negative integers num1 and num2 represented as strings, return the product of num1 and num2.  **Note:**   1. The length of both num1 and num2 is < 110. 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.   **Tags：**Math、String  **Similar Problems：**(M2) Add Two Numbers、(E66) Plus One、(E67) Add Binary、(E415) Add Strings |
| **Solution:** string multiply(string num1, string num2); |

##### M2. Add Two Numbers

|  |
| --- |
| **M2. Add Two Numbers** |
| You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order and each of their nodes contain a single digit. Add the two numbers and return it as a linked list.  You may assume the two numbers do not contain any leading zero, except the number 0 itself.  Input: (2 -> 4 -> 3) + (5 -> 6 -> 4)  Output: 7 -> 0 -> 8  **Tags：**Math、Linked List  **Similar Problems：**(E67) Add Binary、(E415) Add Strings、(M43). Multiply Strings、(M445) Add Two Numbers II、(E371) Sum of Two Integers |
| **Solution:** ListNode\* addTwoNumbers(ListNode\* l1, ListNode\* l2); |

##### M445. Add Two Numbers II

|  |
| --- |
| **M445. Add Two Numbers II** |
| You are given two non-empty linked lists representing two non-negative integers. The most significant digit comes first and each of their nodes contain a single digit. Add the two numbers and return it as a linked list.  You may assume the two numbers do not contain any leading zero, except the number 0 itself.  **Follow up:**  What if you cannot modify the input lists? In other words, reversing the lists is not allowed.  **Example:**  Input: (7 -> 2 -> 4 -> 3) + (5 -> 6 -> 4)  Output: 7 -> 8 -> 0 -> 7  **Tags：**Linked List  **Similar Problems：**(M2) Add Two Numbers |
| **Solution:** ListNode\* addTwoNumbers(ListNode\* l1, ListNode\* l2); |

##### E371. Sum of Two Integers

|  |
| --- |
| **E371. Sum of Two Integers** |
| Calculate the sum of two integers a and b, but you are not allowed to use the operator + and -.  **Example:**  Given a = 1 and b = 2, return 3.  **Tags：**Bit Manipulation  **Similar Problems：**(M2) Add Two Numbers |
| **Solution:** int getSum(int a, int b); |

##### E415. Add Strings

|  |
| --- |
| **E415. Add Strings** |
| 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.   **Tags：**Math  **Similar Problems：**(M2) Add Two Numbers、(M43). Multiply Strings |
| **Solution:** string addStrings(string num1, string num2); |

##### E67. Add Binary

|  |
| --- |
| **E67. Add Binary** |
| Given two binary strings, return their sum (also a binary string).  **For example,**  a = "11"  b = "1"  Return "100".  **Tags：**String、Math  **Similar Problems：**(M2) Add Two Numbers、(M43). Multiply Strings、 (E66) Plus One |
| **Solution:** string addBinary(string a, string b); |

##### E66. Plus One

|  |
| --- |
| **E66. Plus One** |
| Given a non-negative integer represented as a non-empty array of digits, plus one to the integer.  You may assume the integer do not contain any leading zero, except the number 0 itself.  The digits are stored such that the most significant digit is at the head of the list.  **Tags：**Array、Math  **Similar Problems：**(M43). Multiply Strings、(E67) Add Binary |
| **Solution:** vector<int> plusOne(vector<int>& digits); |

#### 组5

##### M5. Longest Palindromic Substring

|  |
| --- |
| **M5. Longest Palindromic Substring** |
| Given a string s, find the longest palindromic substring in s. You may assume that the maximum length of s is 1000.  **Example:**  Input: "babad"  Output: "bab"  Note: "aba" is also a valid answer.  **Example:**  Input: "cbbd"  Output: "bb"  **Tags：**String  **Similar Problems：**(H214) Shortest Palindrome、(E) Palindrome Permutation、 (H336) Palindrome Pairs、 (M516) Longest Palindromic Subsequence、(M647) Palindromic Substrings |
| **Solution:** string longestPalindrome(string s); |

#### 独立组 00（该组题目均独立）

##### M3. Longest Substring Without Repeating Characters

|  |
| --- |
| **M3. Longest Substring Without Repeating Characters** |
| Given a string, find the length of the longest substring without repeating characters.  **Examples:**  Given "abcabcbb", the answer is "abc", which the length is 3.  Given "bbbbb", the answer is "b", with the length of 1.  Given "pwwkew", 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.  **Tags：**Hash Table、Two Pointers、String  **Similar Problems：** (H) Longest Substring with At Most Two Distinct Characters |
| **Solution:** int lengthOfLongestSubstring(string s); |

##### E520. Detect Capital

|  |
| --- |
| **E520. Detect Capital** |
| 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.  **Tags：**String  **题目大意：**判断一个字母是否大小写正确：要么全是大写，要么全是小写，或者首字母大写其他小写，否则不满足题意～ |
| **Solution:** bool detectCapitalUse(string word); |

### 树

#### 组1 路径求和、左右结点互联

##### E112. Path Sum

|  |
| --- |
| **E112. Path Sum** |
| Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum.  **For example:**  Given the below binary tree and sum = 22,    return true, as there exist a root-to-leaf path 5->4->11->2 which sum is 22.  **Tags：**Tree, Depth-first Search  **Similar Problems：** (M113) Path Sum II, (H124) Binary Tree Maximum Path Sum, (M129) Sum Root to Leaf Numbers, (E437) Path Sum III  **题目大意：**给定一个二叉树和一个值sum，判断是否存在一个从根节点到叶子节点的路径，使得路径上每个节点值之和等于sum? |
| **Solution：** bool hasPathSum(TreeNode\* root, int sum); |

##### M113. Path Sum II

|  |
| --- |
| **M113. Path Sum II** |
| Given a binary tree and a sum, find all root-to-leaf paths where each path's sum equals the given sum.  **For example:**  Given the below binary tree and sum = 22,    **Tags：**Tree, Depth-first Search  **Similar Problems：** (E112) Path Sum, (E257) Binary Tree Paths, (E437)Path Sum III  **题目大意：**给定一个二叉树和一个值sum，找到所有从根节点到叶子节点的路径，且路径上每个节点值之和等于sum。 |
| **Solution：**vector<vector<int>> pathSum(TreeNode\* root, int sum); |

##### E437. Path Sum III

|  |
| --- |
| **E437. Path Sum III** |
| The 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：**    **Tags：**Tree  **Similar Problems：** (E112) Path Sum、(M113) Path Sum II、(E687) Longest Univalue Path |
| **Solution：**int pathSum(TreeNode\* root, int sum); |

##### E687. Longest Univalue Path

|  |
| --- |
| **E687. Longest Univalue Path** |
| 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:**    **Note:** The given binary tree has not more than 10000 nodes. The height of the tree is not more than 1000.  **Tags：**Tree、Recursion  **Similar Questions：**(H124) Binary Tree Maximum Path Sum Count、(E437) Path Sum III |
| **Solution:** int longestUnivaluePath(TreeNode\* root); |

##### E257. Binary Tree Paths

|  |
| --- |
| **E257. Binary Tree Paths** |
| Given a binary tree, return all root-to-leaf paths.  For example, given the following binary tree:    All root-to-leaf paths are:  ["1->2->5", "1->3"]  **Related Topics：**Tree、Depth-first Search  **Similar Questions：**(M113)Path Sum II |
| **Solution:** vector<string> binaryTreePaths(TreeNode\* root); |

##### H124. Binary Tree Maximum Path Sum

|  |
| --- |
| **H124. Binary Tree Maximum Path Sum** |
| Given a binary tree, find the maximum path sum.  For this problem, a path is defined as any sequence of nodes from some starting node to any node in the tree along the parent-child connections. The path must contain at least one node and does not need to go through the root.  **For example:**  Given the below binary tree,    Return 6.  **Related Topics：**Tree、Depth-first Search  **Similar Questions：**(E112)Path Sum、(M129)Sum Root to Leaf Numbers、(E687)Longest Univalue Path |
| **Solution：**int maxPathSum(TreeNode\* root) |

##### M129. Sum Root to Leaf Numbers

|  |
| --- |
| **M129. Sum Root to Leaf Numbers** |
| Given a binary tree containing digits from 0-9 only, each root-to-leaf path could represent a number.  An example is the root-to-leaf path 1->2->3 which represents the number 123.  Find the total sum of all root-to-leaf numbers.  For example,    The root-to-leaf path 1->2 represents the number 12.  The root-to-leaf path 1->3 represents the number 13.  Return the sum = 12 + 13 = 25.  **Related Topics：**Tree、Depth-first Search  **Similar Questions：**(E112)Path Sum、(H124) Binary Tree Maximum Path Sum |
| **Solution:** int sumNumbers(TreeNode\* root); |

#### 组2 逐层遍历

##### M102. Binary Tree Level Order Traversal

|  |
| --- |
| **M102. Binary Tree Level Order Traversal** |
| 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],    return its level order traversal as:  [  [3],  [9,20],  [15,7]  ]  .  **Related Topics：**Tree、Breadth-first Search  **Similar Questions：**(M103) Binary Tree Zigzag Level Order Traversal、(E107) Binary Tree Level Order Traversal II、(E111) Minimum Depth of Binary Tree、() Binary Tree Vertical Order Traversal、(E637) Average of Levels in Binary Tree |
| **Solution：**vector<vector<int>> levelOrder(TreeNode\* root) |

##### M103. Binary Tree Zigzag Level Order Traversal

|  |
| --- |
| **M103. Binary Tree Zigzag Level Order Traversal** |
| Given a binary tree, return the zigzag level order traversal of its nodes' values. (ie, from left to right, then right to left for the next level and alternate between).  **For example:**  Given binary tree [3,9,20,null,null,15,7],    return its zigzag level order traversal as:  [  [3],  [20,9],  [15,7]  ]  .  **Related Topics：**Stack、Tree、Breadth-first Search  **Similar Questions：**(M102) Binary Tree Level Order Traversal |
| **Solution：**vector<vector<int>> zigzagLevelOrder(TreeNode\* root) |

##### E104. Maximum Depth of Binary Tree

|  |
| --- |
| **E104. Maximum Depth of Binary Tree** |
| 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.  **Related Topics：**Tree、Depth-first Search  **Similar Questions：**(E110) Balanced Binary Tree、(E111) Minimum Depth of Binary Tree |
| **Solution：**int maxDepth(TreeNode\* root) |

##### E107. Binary Tree Level Order Traversal II

|  |
| --- |
| **E107. Binary Tree Level Order Traversal II** |
| 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],    return its bottom-up level order traversal as:  [  [15,7],  [9,20],  [3]  ]  **Related Topics：**Tree、Breadth-first Search  **Similar Questions：**(M102) Binary Tree Level Order Traversal、(E637) Average of Levels in Binary Tree |
| **Solution：**vector<vector<int>> levelOrderBottom(TreeNode\* root) |

##### E110. Balanced Binary Tree

|  |
| --- |
| **E110. Balanced Binary Tree** |
| 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.  **Related Topics：**Tree、Depth-first Search  **Similar Questions：**(E104) Maximum Depth of Binary Tree |
| **Solution：**bool isBalanced(TreeNode\* root) |

##### E111. Minimum Depth of Binary Tree

|  |
| --- |
| **E111. Minimum Depth of Binary Tree** |
| 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.  **Related Topics：**Tree、Depth-first Search、Breadth-first Search  **Similar Questions：**(M102) Binary Tree Level Order Traversal、(E104) Maximum Depth of Binary Tree |
| **Solution：**int minDepth(TreeNode\* root) |

##### E637. Average of Levels in Binary Tree

|  |
| --- |
| **E637. Average of Levels in Binary Tree** |
| 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:    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:**  The range of node's value is in the range of 32-bit signed integer.  **Related Topics：**Tree  **Similar Questions：**(M102) Binary Tree Level Order Traversal、(E107) Binary Tree Level Order Traversal II |
| **Solution：**vector<double> averageOfLevels(TreeNode\* root) |

#### 组

##### M

|  |
| --- |
| **M** |
| The |
| Solution： |

#### 组

##### M

|  |
| --- |
| **M** |
| The |
| Solution： |

### 链表LinkList

|  |
| --- |
| struct ListNode {  int val;  ListNode \*next;  ListNode(int x) : val(x), next(NULL) {}  }; |

#### 组1

##### E26. Remove Duplicates from Sorted Array

|  |
| --- |
| **E26. Remove Duplicates from Sorted Array** |
| Given a sorted array, 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:**  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 new length.  **Related Topics：**Array、Two Pointers  **Similar Questions：**(E27)Remove Element |
| **Solutions：**int removeDuplicates(vector<int>& nums); |

##### E27. Remove Element

|  |
| --- |
| **E27. Remove Element** |
| Given an array and a value, 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:**  Given nums = [3,2,2,3], val = 3,  Your function should return length = 2, with the first two elements of nums being 2.  **Related Topics：**Array、Two Pointers  **Similar Questions：**(E26)Remove Duplicates from Sorted Array、(E203)Remove Linked List Elements、(E283)Move Zeroes |
| **Solution：**int removeElement(vector<int>& nums, int val); |

##### E203. Remove Linked List Elements

|  |
| --- |
| **E203. Remove Linked List Elements** |
| Remove all elements from a linked list of integers that have value val.  **Example**  Given: 1 --> 2 --> 6 --> 3 --> 4 --> 5 --> 6, val = 6  Return: 1 --> 2 --> 3 --> 4 --> 5  **Related Topics：**Linked List  **Similar Questions：**(E27)Remove Element、(E237)Delete Node in a Linked List |
| **Solutions：**ListNode\* removeElements(ListNode\* head, int val); |

##### E237. Delete Node in a Linked List

|  |
| --- |
| **E237. Delete Node in a Linked List** |
| Write a function to delete a node (except the tail) in a singly linked list, given only access to that node.  Supposed the linked list is 1 -> 2 -> 3 -> 4 and you are given the third node with value 3, the linked list should become 1 -> 2 -> 4 after calling your function.  **Related Topics：**Linked List  **Similar Questions：**(E203)Remove Linked List Elements |
| **Solutions：**void deleteNode(ListNode\* node); |

##### E283. Move Zeroes

|  |
| --- |
| **E283. Move Zeroes** |
| 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.  For example, given nums = [0, 1, 0, 3, 12], after calling your function, nums should be [1, 3, 12, 0, 0].  **Note:**  You must do this in-place without making a copy of the array.  Minimize the total number of operations.  **Related Topics：**Array、Two Pointers  **Similar Questions：**(E)Remove Element |
| **Solutions：**void moveZeroes(vector<int>& nums); |

#### 组1

##### M

|  |
| --- |
| **M** |
| The |

##### M

|  |
| --- |
| **M** |
| The |

##### M

|  |
| --- |
| **M** |
| The |

##### M

|  |
| --- |
| **M** |
| The |