

Cpp exercise 出處及內容

ListNode folder :

LeetCode 21. Merge Two Sorted Lists

code : MergeSortedLists.cpp

Merge the two lists in a one sorted list. The list should be made by splicing together the nodes of the first two lists. Return the head of the merged linked list.

LeetCode 83. Remove Duplicates from Sorted List

code : RemoveDuplicatesSortedList.cpp

Given the head of a sorted linked list, delete all duplicates such that each element appears only once. Return the linked list sorted as well.

LeetCode 94. Binary Tree Inorder Traversal

code : BinaryTreeInorderTraversal.cpp

Given the root of a binary tree, return the inorder traversal of its nodes' values.

LeetCode 100. Same Tree

code : SameTree.cpp

Given the roots of two binary trees p and q, 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.

LeetCode 101. Symmetric Tree

code : SymmetricTree.cpp

Given the root of a binary tree, check whether it is a mirror of itself (i.e., symmetric around its center).

LeetCode 104. Maximum Depth of Binary Tree

code : MaxDepthBinaryTree.cpp

Given the root of a binary tree, return its maximum depth.

A binary tree's maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

LeetCode 108. Convert Sorted Array to Binary Search Tree

code : ConvertSortedArr2BST.cpp

Given an integer array `nums` where the elements are sorted in ascending order, convert it to a height-balanced binary search tree.

A height-balanced binary tree is a binary tree in which the depth of the two subtrees of every node never differs by more than one.

LeetCode 110. Balanced Binary Tree

code : `BalancedBT.cpp`

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 left and right subtrees of every node differ in height by no more than 1.

LeetCode 111. Minimum Depth of Binary Tree

code : `MinDepthBT.cpp`

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.

LeetCode 112. Path Sum

code : `PathSum.cpp`

Given the root of a binary tree and an integer `targetSum`, return `true` if the tree has a root-to-leaf path such that adding up all the values along the path equals `targetSum`.

A leaf is a node with no children.

LeetCode 141. Linked List Cycle

code : `LinkedListCycle.cpp`

Given `head`, the head of a linked list, determine if the linked list has a cycle in it.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Return `true` if there is a cycle in the linked list. Otherwise, return `false`.

LeetCode 144. Binary Tree Preorder Traversal

code : `BTPreorderTraversal.cpp`

Given the root of a binary tree, return the preorder traversal of its nodes' values.

LeetCode 145. Binary Tree Postorder Traversal

code : `BTPostorderTraversal.cpp`

Given the root of a binary tree, return the postorder traversal of its nodes' values.

LeetCode 155. Min Stack

code : `minStack.cpp`

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

LeetCode 160. Intersection of Two Linked Lists

code : IntersectionLinkedLists.cpp

Given the heads of two singly linked-lists headA and headB, return the node at which the two lists intersect. If the two linked lists have no intersection at all, return null.

LeetCode 203. Remove Linked List Elements

code : RemoveLinkedListElements.cpp

Given the head of a linked list and an integer val, remove all the nodes of the linked list that has Node.val == val, and return the new head.

LeetCode 206. Reverse Linked List

code : reverseList.cpp

Given the head of a singly linked list, reverse the list, and return the reversed list.

LeetCode 225. Implement Stack using Queues

code : ImplementStackusingQueues.cpp

Implement a last-in-first-out (LIFO) stack using only two queues. The implemented stack should support all the functions of a normal stack (push, top, pop, and empty).

LeetCode 226. Invert Binary Tree

code : InvertBT.cpp

Given the root of a binary tree, invert the tree, and return its root.

LeetCode 232. Implement Queue using Stacks

code : ImplementQueueusingStacks.cpp

Implement a first in first out (FIFO) queue using only two stacks. The implemented queue should support all the functions of a normal queue (push, peek, pop, and empty).

LeetCode 234. Palindrome Linked List

code : PalindromeLinkedList.cpp

Given the head of a singly linked list, return true if it is a palindrome.

LeetCode 235. Lowest Common Ancestor of a Binary Search Tree

code : LowestCommonAncestorBST.cpp

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).”

LeetCode 237. Delete Node in a Linked List

code : DeleteNodeLinkedList.cpp

Given the head of a linked list and an integer val, remove all the nodes of the linked list that has Node.val == val, and return the new head.

LeetCode 257. Binary Tree Paths

code : BTPaths.cpp

Write a function to delete a node in a singly-linked list. You will not be given access to the head of the list, instead you will be given access to the node to be deleted directly.

It is guaranteed that the node to be deleted is not a tail node in the list.

LeetCode 404. Sum of Left Leaves

code : SumLeftLeaves.cpp

Given the root of a binary tree, return the sum of all left leaves.

A leaf is a node with no children. A left leaf is a leaf that is the left child of another node.

String folder :

LeetCode 13. Roman to Integer

code : RomanIntegercpp.cpp

Given a roman numeral, convert it to an integer.

LeetCode 14. Longest Common Prefix

code : LongestCommon.cpp

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

LeetCode 20. Valid Parentheses

code : ValidParentheses.cpp

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

An input string is valid if:

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

LeetCode 28. Implement strStr()

code : ImplementstrStr().cpp

Given two strings needle and haystack, return the index of the first occurrence of needle in haystack, or -1 if needle is not part of haystack.

LeetCode 58. Length of Last Word

code : LengthofLastWord.cpp

Given a string s consisting of words and spaces, return the length of the last word in the string.

LeetCode 66. Plus One

code : PlusOne.cpp

You are given a large integer represented as an integer array `digits`, where each `digits[i]` is the *i*th digit of the integer. The digits are ordered from most significant to least significant in left-to-right order. The large integer does not contain any leading 0's.

LeetCode 67. Add Binary

code : AddBinary.cpp

Given two binary strings `a` and `b`, return their sum as a binary string.

LeetCode 168. Excel Sheet Column Title

code : ColumnTitle.cpp

Given an integer `columnNumber`, return its corresponding column title as it appears in an Excel sheet.

LeetCode 171. Excel Sheet Column Number

code : ColumnNumber.cpp

Given a string `columnTitle` that represents the column title as appears in an Excel sheet, return its corresponding column number.

LeetCode 205. Isomorphic Strings

code : IsomorphicString.cpp

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

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

LeetCode 228. Summary Ranges

code : SummaryRanges.cpp

A range `[a,b]` is the set of all integers from `a` to `b` (inclusive).

Return the smallest sorted list of ranges that cover all the numbers in the array exactly.

LeetCode 242. Valid Anagram

code : Anagram.cpp

Given two strings `s` and `t`, return `true` if `t` is an anagram of `s`, and `false` otherwise.

An Anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

LeetCode 290. Word Pattern

code : WordPattern.cpp

Given a pattern and a string `s`, find if `s` follows the same pattern.

LeetCode 344. Reverse String

code : ReverseString.cpp

Write a function that reverses a string. The input string is given as an array of characters `s`.

LeetCode 345. Reverse Vowels of a String

code : ReverseVowelsString.cpp

Given a string s, reverse only all the vowels in the string and return it.

LeetCode 383. Ransom Note

code : ransomNote.cpp

Given two strings ransomNote and magazine, return true if ransomNote can be constructed from magazine and false otherwise.

Each letter in magazine can only be used once in ransomNote.

LeetCode 387. First Unique Character in a String

code : FirstUniqueCharacter.cpp

Given a string s, find the first non-repeating character in it and return its index. If it does not exist, return -1.

LeetCode 389. Find the Difference

code : FindtheDifference.cpp

String t is generated by random shuffling string s and then add one more letter at a random position.

Return the letter that was added to t.

LeetCode 392. Is Subsequence

code : IsSubsequence.cpp

Given two strings s and t, return true if s is a subsequence of t, or false otherwise.

A subsequence of a string is a new string that is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters.

LeetCode 405. Convert a Number to Hexadecimal

code : ConvertNumberHexadecimal.cpp

Given an integer num, return a string representing its hexadecimal representation. For negative integers, two's complement method is used.

All the letters in the answer string should be lowercase characters, and there should not be any leading zeros in the answer except for the zero itself.

LeetCode 409. Longest Palindrome

code : LongestPalindrome.cpp

Given a string s which consists of lowercase or uppercase letters, return the length of the longest palindrome that can be built with those letters.

LeetCode 412. Fizz Buzz

code : FizzBuzz.cpp

Given an integer n, return a string array answer (1-indexed) where:

- answer[i] == "FizzBuzz" if i is divisible by 3 and 5.
- answer[i] == "Fizz" if i is divisible by 3.

- `answer[i] == "Buzz"` if `i` is divisible by 5.
- `answer[i] == i` (as a string) if none of the above conditions are true.

LeetCode 415. Add Strings

code : AddStrings.cpp

Given two non-negative integers, `num1` and `num2` represented as string, return the sum of `num1` and `num2` as a string.

LeetCode 434. Number of Segments in a String

code : countSegments.cpp

Given a string `s`, return the number of segments in the string.

A segment is defined to be a contiguous sequence of non-space characters.

Simple folder :

LeetCode 1. Two Sum

code : TwoSum.cpp

Given an array of integers `nums` and an integer `target`, return indices of the two numbers such that they add up to `target`.

LeetCode 9. Palindrome Number

code : PalindromeNumber.cpp

Given an integer `x`, return true if `x` is palindrome integer.

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

LeetCode 26. Remove Duplicates from Sorted Array

code : RemoveDuplicatesfromSortedArray.cpp

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

LeetCode 27. Remove Element

code : RemoveElement.cpp

Given an integer array `nums` and an integer `val`, remove all occurrences of `val` in `nums` in-place. The relative order of the elements may be changed.

LeetCode 35. Search Insert Position

code : SearchInsertPosition.cpp

Given a sorted array of distinct integers 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.

LeetCode 53. Maximum Subarray

code : MaximumSubarray.cpp

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

LeetCode 69. Sqrt(x)

code : Sqrt.cpp

Given a non-negative integer x, compute and return the square root of x. Since the return type is an integer, the decimal digits are truncated, and only the integer part of the result is returned.

LeetCode 118. Pascal's Triangle

code : PascalTriangle.cpp

Given an integer numRows, return the first numRows of Pascal's triangle.

LeetCode 121. Best Time to Buy and Sell Stock

code : BestTimeBuySell.cpp

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

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

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

LeetCode 136. Single Number

code : SingleNumber.cpp

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

LeetCode 169. Majority Element

code : MajorityElement.cpp

Given an array nums of size n, return the majority element.

The majority element is the element that appears more than $\lfloor n / 2 \rfloor$ times.

You may assume that the majority element always exists in the array.

LeetCode 217. Contains Duplicate

code : ContainsDuplicate.cpp

Given an integer array nums, return true if any value appears at least twice in the array, and return false if every element is distinct.

LeetCode 231. Power of Two

code : Power2.cpp

Given an integer n, return true if it is a power of two. Otherwise, return false.

LeetCode 268. Missing Number

code : MissingNumber.cpp

Given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.

LeetCode 283. Move Zeroes

code : MoveZeroes.cpp

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

LeetCode 303. Range Sum Query - Immutable

code : RangeSumQuery .cpp

Given an integer array nums, handle multiple queries of the following type:
Calculate the sum of the elements of nums between indices left and right inclusive where $left \leq right$.

LeetCode 326. Power of Three

code : Power3 .cpp

Given an integer n, return true if it is a power of three. Otherwise, return false.

LeetCode 342. Power of Four

code : Power4.cpp

Given an integer n, return true if it is a power of four. Otherwise, return false.

LeetCode 349. Intersection of Two Arrays

code : IntersectionArrays.cpp

Given two integer arrays nums1 and nums2, return an array of their intersection. Each element in the result must be unique and you may return the result in any order.

LeetCode 414. Third Maximum Number

code : ThirdMaximum.cpp

Given an integer array nums, return the third distinct maximum number in this array. If the third maximum does not exist, return the maximum number.

LeetCode 441. Arranging Coins

code : ArrangingCoins.cpp

You have n coins and you want to build a staircase with these coins. The staircase consists of k rows where the ith row has exactly i coins. The last row of the staircase may be incomplete.

Exercise folder :

內容為 C++ 教材目錄中的範例

教材連結: C++ 教材目錄