Two Sum(1)

- Find sum of sum of all sub-sequences(0.818180207367)
- Sum of all Subarrays(0.579738671538)
- Find maximum sum possible equal sum of three stacks(0.579738671538)
- Print all possible sums of consecutive numbers with sum N(0.536892711852)
- Perfect Sum Problem (Print all subsets with given sum)(0.536892711852)
- Print all n-digit numbers whose sum of digits equals to given sum(0.502328778226)
- Finding sum of digits of a number until sum becomes single digit(0.473682022466)
- Sum of two large numbers(0.449436416524)
- Sum of subset differences(0.449436416524)
- Sum of dependencies in a graph(0.449436416524)

Add Two Numbers(2)

- Add 1 to a given number(0.579738671538)
- Given a number as a string, find the number of contiguous subsequences which recursively add up to 9(0.51675016217)
- Find all combinations that add upto given number(0.502328778226)
- Add two numbers without using arithmetic operators(0.502328778226)
- Add two numbers using ++ and/or —(0.502328778226)
- Write a program to add two numbers in base 14(0.449436416524)
- Add 1 to a number represented as linked list(0.449436416524)
- Add two numbers represented by linked lists | Set 2(0.410362644952)
- Add two numbers represented by linked lists | Set 1(0.410362644952)

- Smallest number divisible by first n numbers(0.368023208756)

Longest Substring Without Repeating Characters(3)

- Length of the longest substring without repeating characters(0.818180207367)
- Longest repeating and non-overlapping substring(0.602974816038)
- Longest Repeating Subsequence(0.411207055068)
- Longest Non-palindromic substring(0.411207055068)
- Find the first repeated character in a string(0.411207055068)
- Count substrings with same first and last characters(0.411207055068)
- Find the longest substring with k unique characters in a given string(0.407352604289)
- Suffix Tree Application 3 – Longest Repeated Substring(0.374807770059)
- Queries for characters in a repeated string(0.336096927276)
- Length of the longest valid substring(0.336096927276)

Median of Two Sorted Arrays(4)

- Median of two sorted arrays(1.0)
- Median of two sorted arrays of different sizes(0.656972921033)
- Sort an array when two halves are sorted(0.569707709055)
- Sort a nearly sorted (or K sorted) array(0.537125579156)
- Search in an almost sorted array(0.503102612415)
- Merge two sorted arrays(0.503102612415)
- Floor in a Sorted Array(0.503102612415)
- Ceiling in a sorted array(0.503102612415)

- Generate all possible sorted arrays from alternate elements of two given sorted arrays(0.474493294343)
- Sort an almost sorted array where only two elements are swapped(0.455201845765)

Longest Palindromic Substring(5)

- Longest Palindromic Substring | Set 2(0.656972921033)
- Longest Palindromic Substring | Set 1(0.656972921033)
- Palindrome Substring Queries(0.503102612415)
- Longest Non-palindromic substring(0.503102612415)
- Suffix Tree Application 6 – Longest Palindromic Substring(0.48267966065)
- Queries on substring palindrome formation(0.411207055068)
- Longest repeating and non-overlapping substring(0.411207055068)
- Length of the longest valid substring(0.411207055068)
- Length of Longest sub-string that can be removed(0.411207055068)
- Count All Palindrome Sub-Strings in a String(0.411207055068)

ZigZag Conversion(6)

- What is conversion constructor in C++?(0.260555671056)
- Type Conversion in Python(0.260555671056)
- Type Conversion in C(0.260555671056)
- Longest Zig-Zag Subsequence(0.260555671056)
- Flip-flop types and their Conversion(0.260555671056)
- Zigzag (or diagonal) traversal of Matrix(0.220288150562)
- Widening Primitive Conversion in Java(0.220288150562)

- Type conversion in Java with Examples(0.220288150562)
- Convert array into Zig-Zag fashion(0.220288150562)
- Conversion of Array To ArrayList in Java(0.220288150562)

Reverse Integer(7)

- Reverse digits of an integer with overflow handled(0.502328778226)
- Median in a stream of integers (running integers)(0.368023208756)
- Reversible numbers(0.336096927276)
- Count of m digit integers that are divisible by an integer n(0.311257467527)
- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Reverse and Add Function(0.260555671056)
- Perfect reversible string(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)

String to Integer (atoi)(8)

- Printing Integer between Strings in Java(0.411207055068)
- String to Integer in Java – parseInt()(0.356300429333)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Number of substrings divisible by 6 in a string of integers(0.318784021754)
- Different ways for Integer to String Conversions In Java(0.318784021754)
- Pairs of complete strings in two sets of strings(0.285306190981)

- Median in a stream of integers (running integers)(0.285306190981)
- Given two strings, find if first string is a subsequence of second(0.285306190981)
- Check if a given string is a valid number (Integer or Floating Point)(0.269517613246)
- Write your own atoi()(0.260555671056)

Palindrome Number(9)

- Check if a number is Palindrome(0.709297266606)
- Number of palindromic paths in a matrix(0.579738671538)
- Given a number, find the next smallest palindrome(0.579738671538)
- Generate all palindromic numbers less than n(0.579738671538)
- Largest palindrome which is product of two n-digit numbers(0.502328778226)
- Check if binary representation of a number is palindrome(0.502328778226)
- Minimum number of deletions to make a string palindrome(0.449436416524)
- Minimum number of Appends needed to make a string palindrome(0.410362644952)
- Find minimum number of merge operations to make an array palindrome(0.410362644952)
- Minimum number of palindromic subsequences to be removed to empty a binary string(0.379978361591)

Regular Expression Matching(10)

- Match a pattern and String without using regular expressions(0.579738671538)
- Regular Expressions, Regular Grammar and Regular Languages(0.537125579156)
- Regular Expressions in Python | Set 2 (Search, Match and Find All)(0.524591090446)

- Regular Expressions in Java(0.503102612415)
- How to write Regular Expressions?(0.503102612415)
- Regex (Regular Expression) In C++(0.411207055068)
- Designing Finite Automata from Regular Expression(0.356300429333)
- Match Expression where a single special character in pattern can match one or more characters(0.346768972134)
- Regular Expression in Python with Examples | Set 1(0.318784021754)
- Expression Tree(0.260555671056)

Container With Most Water(11)

- Trapping Rain Water(0.260555671056)
- The Two Water Jug Puzzle(0.260555671056)
- Program to find amount of water in a given glass(0.220288150562)
- Smallest window that contains all characters of string itself(0.194314340169)
- Count numbers that don't contain 3(0.194314340169)
- Check whether BST contains Dead End or not(0.194314340169)
- Measuring 6L water from 4L and 9L buckets(0.175786078393)
- Find smallest range containing elements from k lists(0.175786078393)
- Print list items containing all characters of a given word(0.161713780663)
- Measure one litre using two vessels and infinite water supply(0.161713780663)

Integer to Roman(12)

- Median in a stream of integers (running integers)(0.368023208756)
- Count of m digit integers that are divisible by an integer n(0.311257467527)

- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)
- Smallest of three integers without comparison operators(0.220288150562)
- Printing Integer between Strings in Java(0.220288150562)
- Multiply a given Integer with 3.5(0.220288150562)
- Longest Subarray of non-negative Integers(0.220288150562)

Roman to Integer(13)

- Median in a stream of integers (running integers)(0.368023208756)
- Count of m digit integers that are divisible by an integer n(0.311257467527)
- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)
- Smallest of three integers without comparison operators(0.220288150562)
- Printing Integer between Strings in Java(0.220288150562)
- Multiply a given Integer with 3.5(0.220288150562)
- Longest Subarray of non-negative Integers(0.220288150562)

Longest Common Prefix(14)

- Longest Common Prefix | Set 6 (Sorting)(0.579738671538)
- Longest Common Prefix | Set 6 (Sorting)(0.579738671538)

- Longest Common Prefix | Set 5 (Using Trie)(0.524591090446)
- Longest Common Prefix | Set 4 (Binary Search)(0.524591090446)
- Longest Common Prefix | Set 3 (Divide and Conquer)(0.524591090446)
- Longest Common Prefix | Set 2 (Character by Character Matching)(0.422233885287)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.422233885287)
- Printing Longest Common Subsequence(0.411207055068)
- Longest common subsequence with permutations allowed(0.356300429333)
- LCS (Longest Common Subsequence) of three strings(0.356300429333)

3Sum(15)

3Sum Closest(16)

- Two elements whose sum is closest to zero(0.220288150562)
- Multiple of x closest to n(0.220288150562)
- Find the closest pair from two sorted arrays(0.220288150562)
- Find the closest leaf in a Binary Tree(0.220288150562)
- Find the closest and smaller tidy number(0.220288150562)
- Find three closest elements from given three sorted arrays(0.194314340169)
- Find the closest element in Binary Search Tree(0.194314340169)
- Find k closest elements to a given value(0.194314340169)
- Closest Pair of Points | O(nlogn) Implementation(0.194314340169)
- Closest leaf to a given node in Binary Tree(0.175786078393)

Letter Combinations of a Phone Number(17)

- Find all combinations that add upto given number(0.291219418564)

- All combinations of strings that can be used to dial a number(0.291219418564)
- Print all combinations of points that can compose a given number(0.260555671056)
- Smallest number divisible by first n numbers(0.241213606675)
- Number with maximum number of prime factors(0.241213606675)
- Number of subtrees having odd count of even numbers(0.241213606675)
- Number of perfect squares between two given numbers(0.241213606675)
- Next higher number with same number of set bits(0.241213606675)
- How to check if a given number is Fibonacci number?(0.241213606675)
- Finding number of digits in n'th Fibonacci number(0.241213606675)

4Sum(18)

Remove Nth Node From End of List(19)

- Find n'th node from the end of a Linked List(0.431613418971)
- Write a function to get Nth node in a Linked List(0.380872608476)
- Remove every k-th node of the linked list(0.380872608476)
- Swap Kth node from beginning with Kth node from end in a Linked List(0.340733448316)
- Given a linked list, reverse alternate nodes and append at the end(0.31710746658)
- Delete N nodes after M nodes of a linked list(0.296672366897)
- Segregate even and odd nodes in a Linked List(0.252334201434)
- Remove duplicates from an unsorted linked list(0.252334201434)
- Remove duplicates from a sorted linked list(0.252334201434)
- Move all occurrences of an element to end in a linked list(0.252334201434)

Valid Parentheses(20)

- Find the number of valid parentheses expressions of given length(0.449436416524)
- Remove Invalid Parentheses(0.260555671056)
- Valid variants of main() in Java(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Print all combinations of balanced parentheses(0.220288150562)
- Length of the longest valid substring(0.220288150562)
- Check for balanced parentheses in an expression(0.220288150562)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)
- How to check if a string is a valid keyword in Python?(0.194314340169)

Merge Two Sorted Lists(21)

- Merge two sorted linked lists(0.776514530475)
- Merge Sort for Linked Lists(0.776514530475)
- Merge two sorted linked lists such that merged list is in reverse order(0.747201455332)
- Merge Sort(0.709297266606)
- Merge Sort for Doubly Linked List(0.656972921033)
- Merge K sorted linked lists(0.656972921033)
- Why Quick Sort preferred for Arrays and Merge Sort for Linked Lists?(0.579588527172)
- Merge two sorted arrays(0.503102612415)
- Iterative Merge Sort(0.503102612415)
- 3-way Merge Sort(0.503102612415)

Generate Parentheses(22)

- Generics in Java(0.336096927276)
- Generators in Python(0.336096927276)
- Generating Test Cases (generate() and generate_n() in C++)(0.336096927276)
- Test Case Generation | Set 5 (Generating random Sorted Arrays and Palindromes)(0.260555671056)
- Remove Invalid Parentheses(0.260555671056)
- Program for Sudoku Generator(0.260555671056)
- Generate Pythagorean Triplets(0.260555671056)
- Print all combinations of balanced parentheses(0.220288150562)
- Mid-Point Line Generation Algorithm(0.220288150562)
- Heap's Algorithm for generating permutations(0.220288150562)

Merge k Sorted Lists(23)

- Merge K sorted linked lists(0.818180207367)
- Merge two sorted linked lists(0.602974816038)
- Merge Sort for Linked Lists(0.602974816038)
- Merge two sorted linked lists such that merged list is in reverse order(0.580212787257)
- Merge Sort(0.579738671538)
- Merge Sort for Doubly Linked List(0.51014901931)
- Merge k sorted arrays | Set 1(0.450175502327)
- Why Quick Sort preferred for Arrays and Merge Sort for Linked Lists?(0.450058913045)
- Sort a nearly sorted (or K sorted) array(0.439015465545)

- Merge two sorted arrays(0.411207055068)

Swap Nodes in Pairs(24)

- Swap nodes in a linked list without swapping data(0.455201845765)
- Swap Kth node from beginning with Kth node from end in a Linked List(0.329894545665)
- Two nodes of a BST are swapped, correct the BST(0.291069102382)
- Swap Nodes in Binary tree of every k'th level(0.291069102382)
- Minimum number of swaps required for arranging pairs adjacent to each other(0.291069102382)
- Identify all Grand-Parent Nodes of each Node in a Map(0.285306190981)
- Given an array of pairs, find all symmetric pairs in it(0.285306190981)
- Print all nodes that are at distance k from a leaf node(0.260555671056)
- Print all nodes at distance k from a given node(0.260555671056)
- Number of swaps to sort when only adjacent swapping allowed(0.241299136472)

Reverse Nodes in k-Group(25)

- Reverse alternate K nodes in a Singly Linked List(0.291069102382)
- Identify all Grand-Parent Nodes of each Node in a Map(0.285306190981)
- Given a linked list, reverse alternate nodes and append at the end(0.269517613246)
- Reversible numbers(0.260555671056)
- Print all nodes that are at distance k from a leaf node(0.260555671056)
- Print all nodes at distance k from a given node(0.260555671056)
- Find all reachable nodes from every node present in a given set(0.241299136472)

- Delete N nodes after M nodes of a linked list(0.241299136472)
- Reverse and Add Function(0.201993092498)
- Perfect reversible string(0.201993092498)

Remove Duplicates from Sorted Array(26)

- Remove duplicates from sorted array(1.0)
- Remove duplicates from an array of small primes(0.51014901931)
- Remove duplicates from a sorted linked list(0.51014901931)
- Sort an array when two halves are sorted(0.465646219099)
- Remove all occurrences of duplicates from a sorted Linked List(0.450175502327)
- Find Equal (or Middle) Point in a sorted array with duplicates(0.450175502327)
- Sort a nearly sorted (or K sorted) array(0.439015465545)
- Search in an almost sorted array(0.411207055068)
- Merge two sorted arrays(0.411207055068)
- Median of two sorted arrays(0.411207055068)

Remove Element(27)

- How to remove an element from ArrayList in Java?(0.579738671538)
- Remove minimum elements from array such that no three consecutive element are either increasing or decreasing(0.549988394922)
- Maximum sum subarray removing at most one element(0.502328778226)
- Remove minimum elements from either side such that 2*min becomes more than max(0.449436416524)
- Make two sets disjoint by removing minimum elements(0.449436416524)
- K-th smallest element after removing some integers from natural numbers(0.410362644952)

- Find minimum possible size of array with given rules for removing elements(0.379978361591)
- Third largest element in an array of distinct elements(0.368023208756)
- Find the two non-repeating elements in an array of repeating elements(0.368023208756)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.368023208756)

Implement strStr()(28)

- Implement your own itoa()(0.336096927276)
- Implement Your Own sizeof(0.336096927276)
- Recursive Implementation of atoi()(0.260555671056)
- Implementing Atbash Cipher(0.260555671056)
- Implementation of a Falling Matrix(0.260555671056)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Implementation of Binomial Heap(0.260555671056)
- Implementation of Affine Cipher(0.260555671056)
- Implement two stacks in an array(0.260555671056)

Divide Two Integers(29)

- Minimum positive integer to divide a number such that the result is an odd(0.410362644952)
- Median in a stream of integers (running integers)(0.368023208756)
- Count of m digit integers that are divisible by an integer n(0.311257467527)

- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)
- Smallest of three integers without comparison operators(0.220288150562)
- Printing Integer between Strings in Java(0.220288150562)
- Multiply a given Integer with 3.5(0.220288150562)

Substring with Concatenation of All Words(30)

- Word formation using concatenation of two dictionary words(0.455201845765)
- Print Kth character in sorted concatenated substrings of a string(0.291069102382)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- Find if a given string can be represented from a substring by iterating the substring “n” times(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Palindrome Substring Queries(0.201993092498)
- Longest Non-palindromic substring(0.201993092498)
- Length Of Last Word in a String(0.201993092498)

Next Permutation(31)

- Permutation Coefficient(0.579738671538)
- Permutation and Combination in Python(0.449436416524)

- Lexicographically next permutation in C++(0.449436416524)
- K difference permutation(0.449436416524)
- How to find Lexicographically previous permutation?(0.449436416524)
- Generate all binary permutations such that there are more or equal 1's than 0's before every point in all permutations(0.449436416524)
- Check if two arrays are permutations of each other(0.449436416524)
- BogoSort or Permutation Sort(0.449436416524)
- Print all permutations with repetition of characters(0.379978361591)
- Print all palindrome permutations of a string(0.379978361591)

Longest Valid Parentheses(32)

- Length of the longest valid substring(0.411207055068)
- Find the number of valid parentheses expressions of given length(0.318784021754)
- Remove Invalid Parentheses(0.201993092498)
- Longest alternating subsequence(0.201993092498)
- Longest Zig-Zag Subsequence(0.201993092498)
- Longest Repeating Subsequence(0.201993092498)
- Longest Non-palindromic substring(0.201993092498)
- Longest Geometric Progression(0.201993092498)
- Longest Consecutive Subsequence(0.201993092498)
- Valid variants of main() in Java(0.17077611319)

Search in Rotated Sorted Array(33)

- Search an element in a sorted and rotated array(0.818180207367)

- Search in an almost sorted array(0.776514530475)
- Find the Rotation Count in Rotated Sorted array(0.635001221407)
- Search, insert and delete in a sorted array(0.51014901931)
- Find the minimum element in a sorted and rotated array(0.51014901931)
- Sort an array when two halves are sorted(0.465646219099)
- Sort a nearly sorted (or K sorted) array(0.439015465545)
- Program for array rotation(0.411207055068)
- Merge two sorted arrays(0.411207055068)
- Median of two sorted arrays(0.411207055068)

Search for a Range(34)

- Best First Search (Informed Search)(0.411207055068)
- Linear Search vs Binary Search(0.368023208756)
- Interpolation search vs Binary search(0.368023208756)
- Anagram Substring Search (Or Search for all permutations)(0.368023208756)
- Why is Binary Search preferred over Ternary Search?(0.336096927276)
- Linear Search(0.336096927276)
- Jump Search(0.336096927276)
- Interpolation Search(0.336096927276)
- Fibonacci Search(0.336096927276)
- Exponential Search(0.336096927276)

Search Insert Position(35)

- Trie | (Insert and Search)(0.503102612415)

- Binary Search Tree | Set 1 (Search and Insertion)(0.418906716157)
- Search, insert and delete in an unsorted array(0.356300429333)
- Search, insert and delete in a sorted array(0.356300429333)
- Binary Search Tree insert with Parent Pointer(0.318784021754)
- Best First Search (Informed Search)(0.318784021754)
- Treap | Set 2 (Implementation of Search, Insert and Delete)(0.291069102382)
- K Dimensional Tree | Set 1 (Search and Insert)(0.291069102382)
- Linear Search vs Binary Search(0.285306190981)
- Interpolation search vs Binary search(0.285306190981)

Valid Sudoku(36)

- Program for Sudoku Generator(0.260555671056)
- Valid variants of main() in Java(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Length of the longest valid substring(0.220288150562)
- Backtracking | Set 7 (Sudoku)(0.220288150562)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)
- How to check if a string is a valid keyword in Python?(0.194314340169)
- Find the number of valid parentheses expressions of given length(0.175786078393)
- Print all valid words that are possible using Characters of Array(0.161713780663)

Sudoku Solver(37)

- Program for Sudoku Generator(0.260555671056)

- Backtracking | Set 7 (Sudoku)(0.220288150562)

Count and Say(38)

- Counting Sort(0.336096927276)
- Count substrings with same first and last characters(0.260555671056)
- Count of parallelograms in a plane(0.260555671056)
- Count numbers with same first and last digits(0.260555671056)
- Count all increasing subsequences(0.260555671056)
- Count Divisors of Factorial(0.260555671056)
- Count Distinct Subsequences(0.260555671056)
- Find Surpasser Count of each element in array(0.220288150562)
- Find Count of Single Valued Subtrees(0.220288150562)
- Count words in a given string(0.220288150562)

Combination Sum(39)

- Find sum of sum of all sub-sequences(0.474330706497)
- Sum of all Subarrays(0.336096927276)
- Find maximum sum possible equal sum of three stacks(0.336096927276)
- Combinations with repetitions(0.336096927276)
- Print all possible sums of consecutive numbers with sum N(0.311257467527)
- Perfect Sum Problem (Print all subsets with given sum)(0.311257467527)
- Print all n-digit numbers whose sum of digits equals to given sum(0.291219418564)
- Finding sum of digits of a number until sum becomes single digit(0.274611786436)

- Sum of two large numbers(0.260555671056)
- Sum of subset differences(0.260555671056)

Combination Sum II(40)

- Find sum of sum of all sub-sequences(0.36771998047)
- Sum of all Subarrays(0.260555671056)
- Find maximum sum possible equal sum of three stacks(0.260555671056)
- Combinations with repetitions(0.260555671056)
- Print all possible sums of consecutive numbers with sum N(0.241299136472)
- Perfect Sum Problem (Print all subsets with given sum)(0.241299136472)
- Print all n-digit numbers whose sum of digits equals to given sum(0.225764846003)
- Finding sum of digits of a number until sum becomes single digit(0.212889950749)
- Sum of two large numbers(0.201993092498)
- Sum of subset differences(0.201993092498)

First Missing Positive(41)

- Find the smallest positive number missing from an unsorted array | Set 1(0.379978361591)
- Find the Missing Number(0.336096927276)
- Find the smallest missing number(0.260555671056)
- Find missing elements of a range(0.260555671056)
- What are C++ features missing in Java?(0.220288150562)
- Program for Method Of False Position(0.220288150562)

- Position of rightmost set bit(0.220288150562)
- Position of an element after stable sort(0.220288150562)
- Find the missing number in Geometric Progression(0.220288150562)
- Find the missing number in Arithmetic Progression(0.220288150562)

Trapping Rain Water(42)

- Trapping Rain Water(1.0)
- The Two Water Jug Puzzle(0.201993092498)
- Program to find amount of water in a given glass(0.17077611319)
- Measuring 6L water from 4L and 9L buckets(0.136276341439)
- Measure one litre using two vessels and infinite water supply(0.125366937987)

Multiply Strings(43)

- Multiply Large Numbers represented as Strings(0.502328778226)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)
- Multiply two polynomials(0.336096927276)
- Check if given string can be split into four distinct strings(0.336096927276)

Wildcard Matching(44)

- Wildcard Pattern Matching(0.709297266606)
- String matching where one string contains wildcard characters(0.379978361591)
- Wildcards in Java(0.336096927276)
- SQL | Wildcard operators(0.260555671056)
- Maximum Bipartite Matching(0.260555671056)
- Match Expression where a single special character in pattern can match one or more characters(0.260555671056)
- Template matching using OpenCV in Python(0.194314340169)
- Find first non matching leaves in two binary trees(0.194314340169)
- Find all strings that match specific pattern in a dictionary(0.194314340169)
- Match a pattern and String without using regular expressions(0.175786078393)

Jump Game II(45)

- Jump Search(0.260555671056)
- Combinatorial Game Theory | Set 2 (Game of Nim)(0.241299136472)
- Implementation of Tic-Tac-Toe game(0.201993092498)
- Implementation of Minesweeper Game(0.201993092498)
- Hangman Game in Python(0.201993092498)
- A Number Link Game(0.201993092498)
- The prisoner's dilemma in Game theory(0.17077611319)
- Puzzle 73 | The Card Game(0.17077611319)
- Puzzle 69 |The Number Game(0.17077611319)
- Project Idea | (A Game of Anagrams)(0.17077611319)

Permutations(46)

- Permutation Coefficient(0.579738671538)
- Permutation and Combination in Python(0.449436416524)
- Lexicographically next permutation in C++(0.449436416524)
- K difference permutation(0.449436416524)
- How to find Lexicographically previous permutation?(0.449436416524)
- Generate all binary permutations such that there are more or equal 1's than 0's before every point in all permutations(0.449436416524)
- Check if two arrays are permutations of each other(0.449436416524)
- BogoSort or Permutation Sort(0.449436416524)
- Print all permutations with repetition of characters(0.379978361591)
- Print all palindrome permutations of a string(0.379978361591)

Permutations II(47)

- Permutation Coefficient(0.336096927276)
- Permutation and Combination in Python(0.260555671056)
- Lexicographically next permutation in C++(0.260555671056)
- K difference permutation(0.260555671056)
- How to find Lexicographically previous permutation?(0.260555671056)
- Generate all binary permutations such that there are more or equal 1's than 0's before every point in all permutations(0.260555671056)
- Check if two arrays are permutations of each other(0.260555671056)
- BogoSort or Permutation Sort(0.260555671056)
- Print all permutations with repetition of characters(0.220288150562)
- Print all palindrome permutations of a string(0.220288150562)

Rotate Image(48)

- Left Rotation and Right Rotation of a String(0.368023208756)
- Find the Rotation Count in Rotated Sorted array(0.368023208756)
- Image Processing in Java | Set 4 (Colored image to Negative image conversion)(0.364020643353)
- Image Processing in Java | Set 3 (Colored image to greyscale image conversion)(0.364020643353)
- Image Procesing in Java | Set 6 (Colored image to Sepia image conversion)(0.364020643353)
- Image Processing in Java | Set 10 (Watermarking an image)(0.311257467527)
- Project Idea | (Model based Image Compression of Medical Images)(0.291219418564)
- Image Processing in Java | Set 8 (Creating mirror image)(0.291219418564)
- Image Processing in Java | Set 11 (Changing orientation of image)(0.291219418564)
- Image Processing in Java | Set 7 (Creating a random pixel image)(0.274611786436)

Group Anagrams(49)

- SQL | GROUP BY(0.336096927276)
- Group Shifted String(0.260555671056)
- Check whether two strings are anagram of each other(0.260555671056)
- Project Idea | (A Game of Anagrams)(0.220288150562)
- Group words with same set of characters(0.220288150562)
- Count of total anagram substrings(0.220288150562)
- A Group chat application in Java(0.220288150562)

- number-theoryGenerators of finite cyclic group under addition(0.194314340169)
- is_permutation() in C++ and its application for anagram search(0.194314340169)
- UHG(United Health Group) Interview Experience(0.194314340169)

Pow(x, n)(50)

- Construct a unique matrix n x n for an input n(0.4003049304)
- No of Factors of n!(0.336096927276)
- Primitive root of a prime number n modulo n(0.311257467527)
- Longest Increasing Subsequence Size (N log N)(0.311257467527)
- Count digits in given number N which divide N(0.311257467527)
- Construction of Longest Increasing Subsequence (N log N)(0.311257467527)
- What is use of %n in printf() ?(0.260555671056)
- Print n x n spiral matrix using O(1) extra space(0.260555671056)
- Longest Monotonically Increasing Subsequence Size (N log N): Simple implementation(0.260555671056)
- Legendre's formula (Given p and n, find the largest x such that p^x divides n!)(0.260555671056)

N-Queens(51)

- Printing all solutions in N-Queen Problem(0.379978361591)

N-Queens II(52)

- Printing all solutions in N-Queen Problem(0.220288150562)
- Flipkart Interview | Set 7 (For SDE II)(0.175786078393)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.161713780663)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.161713780663)

- Flipkart Interview Experience | Set 17 (For SDE II)(0.161713780663)
- Amazon Interview experience | Set 326 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 348 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 313 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 312 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 163 (For SDE II)(0.161713780663)

Maximum Subarray(53)

- Maximum Product Subarray(0.709297266606)
- Sliding Window Maximum (Maximum of all subarrays of size k)(0.590594008858)
- Maximum circular subarray sum(0.579738671538)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.51675016217)
- Maximum sum subarray removing at most one element(0.502328778226)
- Maximum subarray sum modulo m(0.502328778226)
- Find the maximum subarray XOR in a given array(0.502328778226)
- Find maximum average subarray of k length(0.502328778226)
- Find Maximum Sum Strictly Increasing Subarray(0.502328778226)
- Maximum sum two non-overlapping subarrays of given size(0.449436416524)

Spiral Matrix(54)

- Circular Matrix (Construct a matrix with numbers 1 to m*n in spiral way)(0.51675016217)
- Print a given matrix in spiral form(0.502328778226)

- Sum of both diagonals of a spiral odd-order square matrix(0.449436416524)
- Print a given matrix in reverse spiral form(0.449436416524)
- Print K'th element in spiral form of matrix(0.449436416524)
- Queries in a Matrix(0.336096927276)
- Matrix Introduction(0.336096927276)
- Matrix Exponentiation(0.336096927276)
- Determinant of a Matrix(0.336096927276)
- Print n x n spiral matrix using O(1) extra space(0.30321606445)

Jump Game(55)

- Jump Search(0.336096927276)
- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)

Merge Intervals(56)

- Merge Overlapping Intervals(0.709297266606)
- Merge Sort(0.336096927276)

- Interval Tree(0.336096927276)
- Check if any two intervals overlap among a given set of intervals(0.311257467527)
- Merge two sorted linked lists such that merged list is in reverse order(0.260555671056)
- Merge two sorted arrays(0.260555671056)
- Merge operations using STL in C++ (merge, includes, set_union, set_intersection, set_difference, ..)(0.260555671056)
- Iterative Merge Sort(0.260555671056)
- 3-way Merge Sort(0.260555671056)
- Merge two sorted linked lists(0.220288150562)

Insert Interval(57)

- Interval Tree(0.336096927276)
- Insertion Sort(0.336096927276)
- Inserting elements in std::map (insert, emplace and operator [])(0.336096927276)
- Check if any two intervals overlap among a given set of intervals(0.311257467527)
- Trie | (Insert and Search)(0.260555671056)
- SQL | INSERT INTO Statement(0.260555671056)
- Recursive Insertion Sort(0.260555671056)
- Merge Overlapping Intervals(0.260555671056)
- Binary Insertion Sort(0.260555671056)
- Threaded Binary Tree | Insertion(0.220288150562)

Length of Last Word(58)

- Length Of Last Word in a String(0.709297266606)
- Word Ladder (Length of shortest chain to reach a target word)(0.549988394922)
- Word formation using concatenation of two dictionary words(0.336096927276)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Run Length Encoding(0.260555671056)
- Variable length arguments for Macros(0.220288150562)
- Reverse words in a given string(0.220288150562)
- Repeated subsequence of length 2 or more(0.220288150562)

Spiral Matrix II(59)

- Circular Matrix (Construct a matrix with numbers 1 to m*n in spiral way)(0.366529477546)
- Print a given matrix in spiral form(0.356300429333)
- Sum of both diagonals of a spiral odd-order square matrix(0.318784021754)
- Print a given matrix in reverse spiral form(0.318784021754)
- Print K'th element in spiral form of matrix(0.318784021754)
- Queries in a Matrix(0.260555671056)
- Matrix Introduction(0.260555671056)
- Matrix Exponentiation(0.260555671056)
- Determinant of a Matrix(0.260555671056)
- Print n x n spiral matrix using O(1) extra space(0.215070325706)

Permutation Sequence(60)

- Recaman's sequence(0.336096927276)
- Permutation Coefficient(0.336096927276)
- Padovan Sequence(0.336096927276)
- Look-and-Say Sequence(0.336096927276)
- Juggler Sequence(0.336096927276)
- Farey Sequence(0.336096927276)
- Aliquot Sequence(0.336096927276)
- String with additive sequence(0.260555671056)
- Permutation and Combination in Python(0.260555671056)
- Lexicographically next permutation in C++(0.260555671056)

Rotate List(61)

- Rotate a Linked List(0.709297266606)
- Left Rotation and Right Rotation of a String(0.368023208756)
- Find the Rotation Count in Rotated Sorted array(0.368023208756)
- Recursively print all sentences that can be formed from list of word lists(0.311257467527)
- Check if a linked list is Circular Linked List(0.291219418564)
- Sublist Search (Search a linked list in another list)(0.274611786436)
- In-place Merge two linked lists without changing links of first list(0.274611786436)
- Sparse Matrix and its representations | Set 2 (Using List of Lists and Dictionary of keys)(0.260555671056)
- Rotate bits of a number(0.260555671056)

- Rotate Matrix Elements(0.260555671056)

Unique Paths(62)

- Printing Paths in Dijkstra’s Shortest Path Algorithm(0.336096927276)
- Dyck path(0.336096927276)
- SQL | UNIQUE Constraint(0.260555671056)
- Find whether there is path between two cells in matrix(0.260555671056)
- Shortest path in a Binary Maze(0.220288150562)
- Path with maximum average value(0.220288150562)
- Path Traversal Attack and Prevention(0.220288150562)
- Numbers having Unique (or Distinct) digits(0.220288150562)
- Number of palindromic paths in a matrix(0.220288150562)
- Maximum path sum in a triangle.(0.220288150562)

Unique Paths II(63)

- Printing Paths in Dijkstra’s Shortest Path Algorithm(0.260555671056)
- Dyck path(0.260555671056)
- SQL | UNIQUE Constraint(0.201993092498)
- Find whether there is path between two cells in matrix(0.201993092498)
- Shortest path in a Binary Maze(0.17077611319)
- Path with maximum average value(0.17077611319)
- Path Traversal Attack and Prevention(0.17077611319)
- Numbers having Unique (or Distinct) digits(0.17077611319)
- Number of palindromic paths in a matrix(0.17077611319)

- Maximum path sum in a triangle.(0.17077611319)

Minimum Path Sum(64)

- Minimum Sum Path In 3-D Array(0.656972921033)
- Maximum path sum in a triangle.(0.411207055068)
- Maximum Sum Path in Two Arrays(0.411207055068)
- Find sum of sum of all sub-sequences(0.36771998047)
- Minimum sum of two elements from two arrays such that indexes are not same(0.356300429333)
- Maximum Path Sum in a Binary Tree(0.356300429333)
- Sum of minimum absolute difference of each array element(0.318784021754)
- Sum of all the numbers that are formed from root to leaf paths(0.318784021754)
- Minimum sum of two numbers formed from digits of an array(0.318784021754)
- Minimum sum of two numbers formed from digits of an array(0.318784021754)

Valid Number(65)

- Find the number of valid parentheses expressions of given length(0.449436416524)
- Check if a given string is a valid number (Integer or Floating Point)(0.379978361591)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)

- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)

Plus One(66)

Add Binary(67)

- Program to add two binary strings(0.579738671538)
- Binary Search(0.336096927276)
- Binary Heap(0.336096927276)
- Gray to Binary and Binary to Gray conversion(0.311257467527)
- Check whether a binary tree is a full binary tree or not(0.311257467527)
- Binary Tree to Binary Search Tree Conversion(0.291219418564)
- Binary Tree | Set 3 (Types of Binary Tree)(0.274611786436)
- fork() and Binary Tree(0.260555671056)
- Threaded Binary Tree(0.260555671056)
- Reverse and Add Function(0.260555671056)

Text Justification(68)

- Convert Text to Speech in Python(0.220288150562)
- Tokenize text using NLTK in python(0.194314340169)
- Textwrap – Text wrapping and filling in Python(0.194314340169)
- Reading and Writing to text files in Python(0.194314340169)
- Formatted text in Linux Terminal using Python(0.175786078393)
- Different ways of Reading a text file in Java(0.175786078393)

- Counting number of lines, words, characters and paragraphs in a text file using Java(0.133785092946)
- C program to Replace a word in a text by another given word(0.133785092946)

Sqrt(x)(69)

Climbing Stairs(70)

- Count ways to reach the n'th stair(0.194314340169)

Simplify Path(71)

- Printing Paths in Dijkstra's Shortest Path Algorithm(0.336096927276)
- Dyck path(0.336096927276)
- Find whether there is path between two cells in matrix(0.260555671056)
- Simplifying Context Free Grammars(0.220288150562)
- Shortest path in a Binary Maze(0.220288150562)
- Path with maximum average value(0.220288150562)
- Path Traversal Attack and Prevention(0.220288150562)
- Number of palindromic paths in a matrix(0.220288150562)
- Maximum path sum in a triangle.(0.220288150562)
- Maximum Sum Path in Two Arrays(0.220288150562)

Edit Distance(72)

- Check if edit distance between two strings is one(0.579738671538)
- Dynamic Programming | Set 5 (Edit Distance)(0.449436416524)
- Hamming Distance between two strings(0.260555671056)
- Find the minimum distance between two numbers(0.260555671056)

- Find Shortest distance from a guard in a Bank(0.220288150562)
- Print nodes at k distance from root(0.194314340169)
- Placements | QA | Trigonometry & Height and Distances(0.194314340169)
- Placements | QA | Time Speed Distance(0.194314340169)
- Minimum distance to travel to cover all intervals(0.194314340169)
- Maximum distance between two occurrences of same element in array(0.194314340169)

Set Matrix Zeroes(73)

- Total coverage of all zeros in a binary matrix(0.356300429333)
- Set theory | Set Operations(0.318784021754)
- Minimum operations required to set all elements of binary matrix(0.291069102382)
- Dynamic Programming | Set 8 (Matrix Chain Multiplication)(0.291069102382)
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)(0.291069102382)
- Inplace rotate square matrix by 90 degrees | Set 1(0.269517613246)
- Sparse Set(0.260555671056)
- Sets in Python(0.260555671056)
- Set in Java(0.260555671056)
- Queries in a Matrix(0.260555671056)

Search a 2D Matrix(74)

- Search element in a sorted matrix(0.411207055068)
- Search a Word in a 2D Grid of characters(0.356300429333)
- Best First Search (Informed Search)(0.318784021754)

- Linear Search vs Binary Search(0.285306190981)
- Interpolation search vs Binary search(0.285306190981)
- Anagram Substring Search (Or Search for all permutations)(0.285306190981)
- Print 2D matrix in different lines and without curly braces in C/C++?(0.269517613246)
- Why is Binary Search preferred over Ternary Search?(0.260555671056)
- Queries in a Matrix(0.260555671056)
- Matrix Introduction(0.260555671056)

Sort Colors(75)

- Sort a nearly sorted (or K sorted) array(0.450175502327)
- Tag Sort (To get both sorted and original)(0.411207055068)
- Sort an array when two halves are sorted(0.411207055068)
- Odd-Even Sort / Brick Sort(0.411207055068)
- Sorting Strings using Bubble Sort(0.368023208756)
- Bead Sort | A Natural Sorting Algorithm(0.368023208756)
- Tree Sort(0.336096927276)
- Stooge Sort(0.336096927276)
- Sorting Terminology(0.336096927276)
- Sort an almost sorted array where only two elements are swapped(0.336096927276)

Minimum Window Substring(76)

- Find maximum of minimum for every window size in a given array(0.291069102382)
- Second minimum element using minimum comparisons(0.260555671056)

- Maximum and minimum of an array using minimum number of comparisons(0.241299136472)
- Minimum steps to delete a string after repeated deletion of palindrome substrings(0.237739238575)
- Find if a given string can be represented from a substring by iterating the substring "n" times(0.225764846003)
- Window Sliding Technique(0.201993092498)
- Palindrome Substring Queries(0.201993092498)
- Minimum step to reach one(0.201993092498)
- Longest Non-palindromic substring(0.201993092498)
- Find the minimum distance between two numbers(0.201993092498)

Combinations(77)

- Combinations with repetitions(0.579738671538)
- Permutation and Combination in Python(0.449436416524)
- Print all combinations of balanced parentheses(0.379978361591)
- Placements | QA | Permutation and Combination(0.379978361591)
- Find all combinations that add upto given number(0.335175743328)
- All combinations of strings that can be used to dial a number(0.335175743328)
- Print all combinations of points that can compose a given number(0.30321606445)
- Using Chinese Remainder Theorem to Combine Modular equations(0.278942545326)
- QA – Placement Quizzes | Permutation and Combination | Question 9(0.25969799324)
- QA – Placement Quizzes | Permutation and Combination | Question 8(0.25969799324)

Subsets(78)

- Partition a set into two subsets such that the difference of subset sums is minimum(0.536892711852)
- Sum of subset differences(0.449436416524)
- Sum of average of all subsets(0.449436416524)
- Sum of the products of all possible Subsets(0.379978361591)
- Sum of maximum elements of all subsets(0.379978361591)
- Sum of XOR of all possible subsets(0.379978361591)
- Subset with sum divisible by m(0.379978361591)
- Maximum and Minimum Product Subsets(0.379978361591)
- Largest divisible subset in array(0.379978361591)
- Largest Subset with GCD 1(0.379978361591)

Word Search(79)

- Search a Word in a 2D Grid of characters(0.502328778226)
- Best First Search (Informed Search)(0.411207055068)
- Linear Search vs Binary Search(0.368023208756)
- Interpolation search vs Binary search(0.368023208756)
- Anagram Substring Search (Or Search for all permutations)(0.368023208756)
- Word formation using concatenation of two dictionary words(0.336096927276)
- Why is Binary Search preferred over Ternary Search?(0.336096927276)
- Linear Search(0.336096927276)
- Jump Search(0.336096927276)
- Interpolation Search(0.336096927276)

Remove Duplicates from Sorted Array II(80)

- Remove duplicates from sorted array(0.818180207367)
- Remove duplicates from an array of small primes(0.431613418971)
- Remove duplicates from a sorted linked list(0.431613418971)
- Sort an array when two halves are sorted(0.403470577019)
- Remove all occurrences of duplicates from a sorted Linked List(0.380872608476)
- Find Equal (or Middle) Point in a sorted array with duplicates(0.380872608476)
- Sort a nearly sorted (or K sorted) array(0.380395708026)
- Search in an almost sorted array(0.356300429333)
- Merge two sorted arrays(0.356300429333)
- Median of two sorted arrays(0.356300429333)

Search in Rotated Sorted Array II(81)

- Search an element in a sorted and rotated array(0.669418851727)
- Search in an almost sorted array(0.656972921033)
- Find the Rotation Count in Rotated Sorted array(0.53724507516)
- Search, insert and delete in a sorted array(0.431613418971)
- Find the minimum element in a sorted and rotated array(0.431613418971)
- Sort an array when two halves are sorted(0.403470577019)
- Sort a nearly sorted (or K sorted) array(0.380395708026)
- Program for array rotation(0.356300429333)
- Merge two sorted arrays(0.356300429333)
- Median of two sorted arrays(0.356300429333)

Remove Duplicates from Sorted List II(82)

- Remove duplicates from a sorted linked list(0.669418851727)
- Remove all occurrences of duplicates from a sorted Linked List(0.580332984677)
- Remove duplicates from sorted array(0.51014901931)
- Remove duplicates from an unsorted linked list(0.431613418971)
- Sort linked list which is already sorted on absolute values(0.296672366897)
- Given a linked list which is sorted, how will you insert in sorted way(0.296672366897)
- Remove all duplicates from a given string(0.291219418564)
- Recursively remove all adjacent duplicates(0.291219418564)
- Merge two sorted linked lists(0.291219418564)
- Merge Sort for Linked Lists(0.291219418564)

Remove Duplicates from Sorted List(83)

- Remove duplicates from a sorted linked list(0.818180207367)
- Remove all occurrences of duplicates from a sorted Linked List(0.709297266606)
- Remove duplicates from sorted array(0.602974816038)
- Remove duplicates from an unsorted linked list(0.51014901931)
- Sort linked list which is already sorted on absolute values(0.342390186113)
- Given a linked list which is sorted, how will you insert in sorted way(0.342390186113)
- Remove all duplicates from a given string(0.336096927276)
- Recursively remove all adjacent duplicates(0.336096927276)
- Merge two sorted linked lists(0.336096927276)
- Merge Sort for Linked Lists(0.336096927276)

Largest Rectangle in Histogram(84)

- Largest Rectangular Area in a Histogram | Set 2(0.318784021754)
- Largest Rectangular Area in a Histogram | Set 1(0.318784021754)
- Find the largest rectangle of 1's with swapping of columns allowed(0.318784021754)
- Find if two rectangles overlap(0.260555671056)
- Largest subarray with GCD one(0.201993092498)
- Find the largest three elements in an array(0.201993092498)
- Second largest element in BST(0.17077611319)
- Program to find largest element in an array(0.17077611319)
- Largest permutation after at most k swaps(0.17077611319)
- Largest divisible subset in array(0.17077611319)

Maximal Rectangle(85)

- Find if two rectangles overlap(0.336096927276)
- Maximizing Unique Pairs from two arrays(0.220288150562)
- Count number of squares in a rectangle(0.220288150562)
- Check if four segments form a rectangle(0.220288150562)
- Stock Buy Sell to Maximize Profit(0.194314340169)
- Maximize number of 0s by flipping a subarray(0.194314340169)
- Puzzle 12 | (Maximize probability of White Ball)(0.175786078393)
- Place k elements such that minimum distance is maximized(0.175786078393)
- Modify array to maximize sum of adjacent differences(0.175786078393)
- Maximum size rectangle binary sub-matrix with all 1s(0.175786078393)

Partition List(86)

- Partitioning a linked list around a given value and If we don't care about making the elements of the list "stable"(0.465100545562)
- Partitioning a linked list around a given value and keeping the original order(0.410362644952)
- Recursively print all sentences that can be formed from list of word lists(0.311257467527)
- Check if a linked list is Circular Linked List(0.291219418564)
- Sublist Search (Search a linked list in another list)(0.274611786436)
- In-place Merge two linked lists without changing links of first list(0.274611786436)
- Sparse Matrix and its representations | Set 2 (Using List of Lists and Dictionary of keys)(0.260555671056)
- Rotate a Linked List(0.260555671056)
- Partition a number into two divisble parts(0.260555671056)
- Merge two sorted linked lists such that merged list is in reverse order(0.260555671056)

Scramble String(87)

- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second

string(0.336096927276)

- Check if given string can be split into four distinct strings(0.336096927276)
- Sort a string according to the order defined by another string(0.311257467527)
- Find the smallest window in a string containing all characters of another string(0.311257467527)

Merge Sorted Array(88)

- Merge two sorted arrays(1.0)
- Merge Sort(0.709297266606)
- Merge two sorted arrays with O(1) extra space(0.579738671538)
- Merge k sorted arrays | Set 1(0.579738671538)
- Why Quick Sort preferred for Arrays and Merge Sort for Linked Lists?(0.579588527172)
- Sort an array when two halves are sorted(0.569707709055)
- Sort a nearly sorted (or K sorted) array(0.537125579156)
- Search in an almost sorted array(0.503102612415)
- Median of two sorted arrays(0.503102612415)
- Iterative Merge Sort(0.503102612415)

Gray Code(89)

- Generate n-bit Gray Codes(0.579738671538)
- Fibonacci Coding(0.336096927276)
- Gray to Binary and Binary to Gray conversion(0.311257467527)
- Secure coding – What is it all about?(0.260555671056)
- Prufer Code to Tree Creation(0.220288150562)

- Packaging and Publishing Python code(0.220288150562)
- Optimization Tips for Python Code(0.220288150562)
- Code Injection and Mitigation with Example(0.220288150562)
- Writing OS Independent Code in C/C++(0.194314340169)
- Write Code to Determine if Two Trees are Identical(0.194314340169)

Subsets II(90)

- Partition a set into two subsets such that the difference of subset sums is minimum(0.311257467527)
- Sum of subset differences(0.260555671056)
- Sum of average of all subsets(0.260555671056)
- Sum of the products of all possible Subsets(0.220288150562)
- Sum of maximum elements of all subsets(0.220288150562)
- Sum of XOR of all possible subsets(0.220288150562)
- Subset with sum divisible by m(0.220288150562)
- Maximum and Minimum Product Subsets(0.220288150562)
- Largest divisible subset in array(0.220288150562)
- Largest Subset with GCD 1(0.220288150562)

Decode Ways(91)

- Decode a given pattern in two ways (Flipkart Interview Question)(0.410362644952)
- Huffman Decoding(0.336096927276)
- Ways to copy a vector in C++(0.220288150562)
- Efficient way to multiply with 7(0.220288150562)

- All ways to add parenthesis for evaluation(0.220288150562)
- Ways to read input from console in Java(0.194314340169)
- Three way partitioning of an array around a given range(0.194314340169)
- Sort a Matrix in all way increasing order(0.194314340169)
- Number of ways to traverse an N-ary tree(0.194314340169)
- How to read content of GeeksforGeeks in an organized way?(0.194314340169)

Reverse Linked List II(92)

- Can we reverse a linked list in less than O(n)?(0.776514530475)
- Reverse a Doubly Linked List(0.602974816038)
- Write a function to reverse a linked list(0.51014901931)
- Check if a linked list is Circular Linked List(0.474330706497)
- Reverse a Linked List in groups of given size(0.450175502327)
- In-place Merge two linked lists without changing links of first list(0.439404118785)
- Merge two sorted linked lists such that merged list is in reverse order(0.424429533893)
- Rotate a Linked List(0.411207055068)
- Merge a linked list into another linked list at alternate positions(0.411207055068)
- Identical Linked Lists(0.411207055068)

Restore IP Addresses(93)

- IP Addressing | Classless Addressing(0.569707709055)
- IP Addressing | Introduction and Classful Addressing(0.502929265114)

- Program to validate an IP address(0.411207055068)
- Java program to find IP address of your computer(0.356300429333)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 7(0.318784021754)
- Computer Networks | IP Addressing | Question 6(0.318784021754)

Binary Tree Inorder Traversal(94)

- Find all possible binary trees with given Inorder Traversal(0.709297266606)
- Inorder Non-threaded Binary Tree Traversal without Recursion or Stack(0.634808797178)
- Construct Special Binary Tree from given Inorder traversal(0.634808797178)
- Inorder Tree Traversal without Recursion(0.602974816038)
- Diagonal Traversal of Binary Tree(0.602974816038)
- Density of Binary Tree in One Traversal(0.602974816038)
- Boundary Traversal of binary tree(0.602974816038)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Tree Traversals (Inorder, Preorder and Postorder)(0.51014901931)
- Inorder Tree Traversal without recursion and without stack!(0.51014901931)

Unique Binary Search Trees II(95)

- Binary Tree to Binary Search Tree Conversion(0.572463774455)

- Binary Search(0.502328778226)
- Minimum swap required to convert binary tree to binary search tree(0.461313774437)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Binary Search Tree | Set 1 (Search and Insertion)(0.439338734046)
- Treap (A Randomized Binary Search Tree)(0.431613418971)
- Threaded Binary Search Tree | Deletion(0.431613418971)
- Merge Two Balanced Binary Search Trees(0.431613418971)
- Inorder Successor in Binary Search Tree(0.431613418971)
- How to handle duplicates in Binary Search Tree?(0.431613418971)

Unique Binary Search Trees(96)

- Binary Tree to Binary Search Tree Conversion(0.676628251794)
- Binary Search(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)
- Binary Search Tree | Set 1 (Search and Insertion)(0.519280018803)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Treap (A Randomized Binary Search Tree)(0.51014901931)
- Threaded Binary Search Tree | Deletion(0.51014901931)
- Merge Two Balanced Binary Search Trees(0.51014901931)
- Inorder Successor in Binary Search Tree(0.51014901931)
- How to handle duplicates in Binary Search Tree?(0.51014901931)

Interleaving String(97)

- Dynamic Programming | Set 33 (Find if a string is interleaved of two other strings)(0.590594008858)
- Print all interleavings of given two strings(0.579738671538)
- Check whether a given string is an interleaving of two other given strings(0.549988394922)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)

Validate Binary Search Tree(98)

- Binary Tree to Binary Search Tree Conversion(0.676628251794)
- Binary Search(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)
- Binary Search Tree | Set 1 (Search and Insertion)(0.519280018803)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Treap (A Randomized Binary Search Tree)(0.51014901931)
- Threaded Binary Search Tree | Deletion(0.51014901931)

- Merge Two Balanced Binary Search Trees(0.51014901931)
- Inorder Successor in Binary Search Tree(0.51014901931)
- How to handle duplicates in Binary Search Tree?(0.51014901931)

Recover Binary Search Tree(99)

- Binary Tree to Binary Search Tree Conversion(0.676628251794)
- Binary Search(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)
- Binary Search Tree | Set 1 (Search and Insertion)(0.519280018803)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Treap (A Randomized Binary Search Tree)(0.51014901931)
- Threaded Binary Search Tree | Deletion(0.51014901931)
- Merge Two Balanced Binary Search Trees(0.51014901931)
- Inorder Successor in Binary Search Tree(0.51014901931)
- How to handle duplicates in Binary Search Tree?(0.51014901931)

Same Tree(100)

- Convert a given tree to its Sum Tree(0.634808797178)
- Binary Indexed Tree or Fenwick Tree(0.634808797178)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.579738671538)
- Tree Sort(0.579738671538)
- Tournament Tree (Winner Tree) and Binary Heap(0.579738671538)
- Quad Tree(0.579738671538)

- Interval Tree(0.579738671538)
- Expression Tree(0.579738671538)
- Double Tree(0.579738671538)
- Continuous Tree(0.579738671538)

Symmetric Tree(101)

- Symmetric Tree (Mirror Image of itself)(0.579738671538)
- Check for Symmetric Binary Tree (Iterative Approach)(0.449436416524)
- Convert a given tree to its Sum Tree(0.368023208756)
- Binary Indexed Tree or Fenwick Tree(0.368023208756)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.336096927276)
- Tree Sort(0.336096927276)
- Tournament Tree (Winner Tree) and Binary Heap(0.336096927276)
- Quad Tree(0.336096927276)
- Interval Tree(0.336096927276)
- Expression Tree(0.336096927276)

Binary Tree Level Order Traversal(102)

- Level Order Tree Traversal(0.818180207367)
- Perfect Binary Tree Specific Level Order Traversal(0.747407354006)
- Given level order traversal of a Binary Tree, check if the Tree is a Min-Heap(0.719040093496)
- Print a Binary Tree in Vertical Order | Set 3 (Using Level Order Traversal)(0.634633579703)
- Perfect Binary Tree Specific Level Order Traversal | Set 2(0.622540746814)

- Construct a tree from Inorder and Level order traversals(0.580332984677)
- Reverse Level Order Traversal(0.51014901931)
- Get Level of a node in a Binary Tree(0.51014901931)
- Diagonal Traversal of Binary Tree(0.51014901931)
- Density of Binary Tree in One Traversal(0.51014901931)

Binary Tree Zigzag Level Order Traversal(103)

- Level Order Tree Traversal(0.709297266606)
- Perfect Binary Tree Specific Level Order Traversal(0.632790458368)
- Given level order traversal of a Binary Tree, check if the Tree is a Min-Heap(0.608773392327)
- Print a Binary Tree in Vertical Order | Set 3 (Using Level Order Traversal)(0.537310840793)
- Perfect Binary Tree Specific Level Order Traversal | Set 2(0.527072475829)
- Construct a tree from Inorder and Level order traversals(0.503102612415)
- Reverse Level Order Traversal(0.450175502327)
- Get Level of a node in a Binary Tree(0.450175502327)
- Diagonal Traversal of Binary Tree(0.450175502327)
- Density of Binary Tree in One Traversal(0.450175502327)

Maximum Depth of Binary Tree(104)

- Maximum width of a binary tree(0.602974816038)
- Find maximum (or minimum) in Binary Tree(0.602974816038)
- Find Minimum Depth of a Binary Tree(0.602974816038)
- Check whether a binary tree is a full binary tree or not(0.519174772633)

- Maximum Path Sum in a Binary Tree(0.51014901931)
- Find maximum level sum in Binary Tree(0.51014901931)
- Calculate depth of a full Binary tree from Preorder(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)
- Write a Program to Find the Maximum Depth or Height of a Tree(0.450175502327)
- Maximum sum of nodes in Binary tree such that no two are adjacent(0.450175502327)

Construct Binary Tree from Preorder and Inorder Traversal(105)

- Construct Tree from given Inorder and Preorder traversals(0.716811741443)
- Construct Special Binary Tree from given Inorder traversal(0.632790458368)
- Construct Full Binary Tree from given preorder and postorder traversals(0.632790458368)
- Tree Traversals (Inorder, Preorder and Postorder)(0.580332984677)
- Construct a Binary Tree from Postorder and Inorder(0.580332984677)
- If you are given two traversal sequences, can you construct the binary tree?(0.503102612415)
- Find all possible binary trees with given Inorder Traversal(0.503102612415)
- Construct a tree from Inorder and Level order traversals(0.503102612415)
- Construct a special tree from given preorder traversal(0.503102612415)
- Inorder Non-threaded Binary Tree Traversal without Recursion or Stack(0.450268144656)

Construct Binary Tree from Inorder and Postorder Traversal(106)

- Construct a Binary Tree from Postorder and Inorder(0.84664735365)

- Construct Special Binary Tree from given Inorder traversal(0.632790458368)
- Construct Full Binary Tree from given preorder and postorder traversals(0.632790458368)
- Tree Traversals (Inorder, Preorder and Postorder)(0.580332984677)
- If you are given two traversal sequences, can you construct the binary tree?(0.503102612415)
- Find all possible binary trees with given Inorder Traversal(0.503102612415)
- Construct a tree from Inorder and Level order traversals(0.503102612415)
- Construct a Binary Search Tree from given postorder(0.503102612415)
- Construct Tree from given Inorder and Preorder traversals(0.503102612415)
- Inorder Non-threaded Binary Tree Traversal without Recursion or Stack(0.450268144656)

Binary Tree Level Order Traversal II(107)

- Level Order Tree Traversal(0.709297266606)
- Perfect Binary Tree Specific Level Order Traversal(0.632790458368)
- Given level order traversal of a Binary Tree, check if the Tree is a Min-Heap(0.608773392327)
- Print a Binary Tree in Vertical Order | Set 3 (Using Level Order Traversal)(0.537310840793)
- Perfect Binary Tree Specific Level Order Traversal | Set 2(0.527072475829)
- Construct a tree from Inorder and Level order traversals(0.503102612415)
- Get Level of a node in a Binary Tree(0.450175502327)
- Reverse Level Order Traversal(0.450175502327)
- Diagonal Traversal of Binary Tree(0.450175502327)
- Density of Binary Tree in One Traversal(0.450175502327)

Convert Sorted Array to Binary Search Tree(108)

- Search in an almost sorted array(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.533207479545)
- Binary Tree to Binary Search Tree Conversion(0.505164486208)
- Tree Sort(0.449436416524)
- Binary Search(0.449436416524)
- Check if given sorted sub-sequence exists in binary search tree(0.411207055068)
- Convert a Binary Tree to Threaded binary tree | Set 2 (Efficient)(0.407081366967)
- Check whether a binary tree is a full binary tree or not(0.402484879511)
- Binary Search Tree | Set 1 (Search and Insertion)(0.38768972948)
- Convert a Binary Tree to Threaded binary tree | Set 1 (Using Queue)(0.385193598874)

Convert Sorted List to Binary Search Tree(109)

- Minimum swap required to convert binary tree to binary search tree(0.533207479545)
- Binary Tree to Binary Search Tree Conversion(0.505164486208)
- Convert a Binary Tree to a Circular Doubly Link List(0.450268144656)
- Tree Sort(0.449436416524)
- Binary Search(0.449436416524)
- Convert a Binary Tree into Doubly Linked List in spiral fashion(0.411207055068)
- Check if given sorted sub-sequence exists in binary search tree(0.411207055068)
- Convert a Binary Tree to Threaded binary tree | Set 2 (Efficient)(0.407081366967)

- Check whether a binary tree is a full binary tree or not(0.402484879511)
- Binary Search Tree | Set 1 (Search and Insertion)(0.38768972948)

Balanced Binary Tree(110)

- Merge Two Balanced Binary Search Trees(0.656972921033)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Count Balanced Binary Trees of Height h(0.579738671538)
- Count Balanced Binary Trees of Height h(0.579738671538)
- Check if a given Binary Tree is height balanced like a Red-Black Tree(0.579588527172)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)

Minimum Depth of Binary Tree(111)

- Find Minimum Depth of a Binary Tree(1.0)
- Find maximum (or minimum) in Binary Tree(0.602974816038)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Calculate depth of a full Binary tree from Preorder(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)

- Find the node with minimum value in a Binary Search Tree(0.450175502327)
- Binary Tree | Set 3 (Types of Binary Tree)(0.439404118785)
- fork() and Binary Tree(0.411207055068)
- Threaded Binary Tree(0.411207055068)

Path Sum(112)

- Maximum path sum in a triangle.(0.579738671538)
- Maximum Sum Path in Two Arrays(0.579738671538)
- Minimum Sum Path In 3-D Array(0.502328778226)
- Maximum Path Sum in a Binary Tree(0.502328778226)
- Find sum of sum of all sub-sequences(0.474330706497)
- Sum of all the numbers that are formed from root to leaf paths(0.449436416524)
- Maximum sum of a path in a Right Number Triangle(0.449436416524)
- Find the maximum path sum between two leaves of a binary tree(0.449436416524)
- Root to leaf path sum equal to a given number(0.410362644952)
- Print all the paths from root, with a specified sum in Binary tree(0.410362644952)

Path Sum II(113)

- Maximum path sum in a triangle.(0.411207055068)
- Maximum Sum Path in Two Arrays(0.411207055068)
- Find sum of sum of all sub-sequences(0.36771998047)
- Minimum Sum Path In 3-D Array(0.356300429333)
- Maximum Path Sum in a Binary Tree(0.356300429333)

- Sum of all the numbers that are formed from root to leaf paths(0.318784021754)
- Maximum sum of a path in a Right Number Triangle(0.318784021754)
- Find the maximum path sum between two leaves of a binary tree(0.318784021754)
- Root to leaf path sum equal to a given number(0.291069102382)
- Print all the paths from root, with a specified sum in Binary tree(0.291069102382)

Flatten Binary Tree to Linked List(114)

- Flattening a Linked List(0.656972921033)
- Extract Leaves of a Binary Tree in a Doubly Linked List(0.519387993313)
- Convert a Binary Tree to a Circular Doubly Link List(0.519387993313)
- Construct Complete Binary Tree from its Linked List Representation(0.519387993313)
- Flatten a multilevel linked list(0.51014901931)
- Convert a Binary Tree into Doubly Linked List in spiral fashion(0.474330706497)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Convert a given Binary Tree to Doubly Linked List | Set 4(0.439274990316)
- Convert a given Binary Tree to Doubly Linked List | Set 3(0.439274990316)
- Convert a given Binary Tree to Doubly Linked List | Set 2(0.439274990316)

Distinct Subsequences(115)

- Count Distinct Subsequences(0.709297266606)
- Count distinct occurrences as a subsequence(0.579738671538)
- Find all distinct subset (or subsequence) sums of an array(0.502328778226)

- Subarrays with distinct elements(0.260555671056)
- Shortest Uncommon Subsequence(0.260555671056)
- SQL | Distinct Clause(0.260555671056)
- Queries on subsequence of string(0.260555671056)
- Longest alternating subsequence(0.260555671056)
- Longest Zig-Zag Subsequence(0.260555671056)
- Longest Repeating Subsequence(0.260555671056)

Populating Next Right Pointers in Each Node(116)

- Point arbit pointer to greatest value right side node in a linked list(0.348993907955)
- Populate Inorder Successor for all nodes(0.336096927276)
- Find next right node of a given key(0.336096927276)
- Double Pointer (Pointer to Pointer) in C(0.316762744302)
- Delete nodes which have a greater value on right side(0.291219418564)
- Identify all Grand-Parent Nodes of each Node in a Map(0.241213606675)
- Point to next higher value node in a linked list with an arbitrary pointer(0.220288150562)
- Print all nodes that are at distance k from a leaf node(0.220288150562)
- Print all nodes at distance k from a given node(0.220288150562)
- Pointer to an Array | Array Pointer(0.220288150562)

Populating Next Right Pointers in Each Node II(117)

- Point arbit pointer to greatest value right side node in a linked list(0.295267555382)

- Populate Inorder Successor for all nodes(0.291219418564)
- Find next right node of a given key(0.291219418564)
- Double Pointer (Pointer to Pointer) in C(0.279413774604)
- Delete nodes which have a greater value on right side(0.252334201434)
- Identify all Grand-Parent Nodes of each Node in a Map(0.212772510465)
- Print all nodes that are at distance k from a leaf node(0.194314340169)
- Print all nodes at distance k from a given node(0.194314340169)
- Pointer to an Array | Array Pointer(0.194314340169)
- Opaque Pointer(0.194314340169)

Pascal's Triangle(118)

- Pascal's Triangle(0.336096927276)
- Classify a triangle(0.336096927276)
- Triangle with no point inside(0.260555671056)
- Find other two sides of a right angle triangle(0.260555671056)
- Find all angles of a given triangle(0.260555671056)
- Travel Triangle Interview Experience(0.220288150562)
- Puzzle 67 | Fit Triangle(0.220288150562)
- Number of Triangles in an Undirected Graph(0.220288150562)
- Maximum path sum in a triangle.(0.220288150562)
- Find coordinates of the triangle given midpoint of each side(0.220288150562)

Pascal's Triangle II(119)

- Pascal's Triangle(0.260555671056)

- Classify a triangle(0.260555671056)
- Triangle with no point inside(0.201993092498)
- Find other two sides of a right angle triangle(0.201993092498)
- Find all angles of a given triangle(0.201993092498)
- Travel Triangle Interview Experience(0.17077611319)
- Puzzle 67 | Fit Triangle(0.17077611319)
- Number of Triangles in an Undirected Graph(0.17077611319)
- Maximum path sum in a triangle.(0.17077611319)
- Find coordinates of the triangle given midpoint of each side(0.17077611319)

Triangle(120)

- Pascal's Triangle(0.579738671538)
- Classify a triangle(0.579738671538)
- Triangle with no point inside(0.449436416524)
- Find other two sides of a right angle triangle(0.449436416524)
- Find all angles of a given triangle(0.449436416524)
- Travel Triangle Interview Experience(0.379978361591)
- Puzzle 67 | Fit Triangle(0.379978361591)
- Number of Triangles in an Undirected Graph(0.379978361591)
- Maximum path sum in a triangle.(0.379978361591)
- Find coordinates of the triangle given midpoint of each side(0.379978361591)

Best Time to Buy and Sell Stock(121)

- Stock Buy Sell to Maximize Profit(0.431613418971)

- Maximum profit by buying and selling a share at most k times(0.344642141038)
- Maximum profit by buying and selling a share at most twice(0.225764846003)
- Changing One Clock Time to Other Time in Minimum Number of Operations(0.179953413782)
- Time Functions in Python | Set 1 (time(), ctime(), sleep()…)(0.168368421637)
- An interesting time complexity question(0.150640184987)
- A Time Complexity Question(0.150640184987)
- The Stock Span Problem(0.150640184987)
- What to do at the time of Wrong Answer (WA)?(0.127359529795)
- Time Complexity of building a heap(0.127359529795)

Best Time to Buy and Sell Stock II(122)

- Stock Buy Sell to Maximize Profit(0.380872608476)
- Maximum profit by buying and selling a share at most k times(0.304125741875)
- Maximum profit by buying and selling a share at most twice(0.201993092498)
- Changing One Clock Time to Other Time in Minimum Number of Operations(0.16279449512)
- Time Functions in Python | Set 1 (time(), ctime(), sleep()…)(0.152314155194)
- The Stock Span Problem(0.136276341439)
- An interesting time complexity question(0.136276341439)
- A Time Complexity Question(0.136276341439)
- [TopTalent.in] How Flipkart gets the best out of their applicants(0.115215543378)
- What to do at the time of Wrong Answer (WA)?(0.115215543378)

Best Time to Buy and Sell Stock III(123)

- Stock Buy Sell to Maximize Profit(0.380872608476)
- Maximum profit by buying and selling a share at most k times(0.304125741875)
- Maximum profit by buying and selling a share at most twice(0.201993092498)
- Changing One Clock Time to Other Time in Minimum Number of Operations(0.16279449512)
- Time Functions in Python | Set 1 (time(), ctime(), sleep()...)(0.152314155194)
- The Stock Span Problem(0.136276341439)
- An interesting time complexity question(0.136276341439)
- A Time Complexity Question(0.136276341439)
- [TopTalent.in] How Flipkart gets the best out of their applicants(0.115215543378)
- What to do at the time of Wrong Answer (WA)?(0.115215543378)

Binary Tree Maximum Path Sum(124)

- Maximum Path Sum in a Binary Tree(1.0)
- Find the maximum path sum between two leaves of a binary tree(0.84664735365)
- Find the maximum sum leaf to root path in a Binary Tree(0.747407354006)
- Find maximum level sum in Binary Tree(0.669418851727)
- Maximum sum of nodes in Binary tree such that no two are adjacent(0.580332984677)
- Print all the paths from root, with a specified sum in Binary tree(0.519387993313)
- Maximum Consecutive Increasing Path Length in Binary Tree(0.519387993313)
- Maximum width of a binary tree(0.51014901931)

- Maximum path sum in a triangle.(0.51014901931)
- Maximum Sum Path in Two Arrays(0.51014901931)

Valid Palindrome(125)

- Palindromic Primes(0.336096927276)
- Smallest Palindrome after replacement(0.260555671056)
- Palindrome Substring Queries(0.260555671056)
- Lexicographically first palindromic string(0.260555671056)
- Check if a number is Palindrome(0.260555671056)
- Valid variants of main() in Java(0.220288150562)
- Queries on substring palindrome formation(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Print all palindromic partitions of a string(0.220288150562)

Word Ladder II(126)

- Word Ladder (Length of shortest chain to reach a target word)(0.390105265183)
- Word formation using concatenation of two dictionary words(0.260555671056)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Snake and Ladder Problem(0.201993092498)
- Length Of Last Word in a String(0.201993092498)
- Reverse words in a given string(0.17077611319)

- Group words with same set of characters(0.17077611319)
- Find the k most frequent words from a file(0.17077611319)

Word Ladder(127)

- Word Ladder (Length of shortest chain to reach a target word)(0.549988394922)
- Word formation using concatenation of two dictionary words(0.336096927276)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Snake and Ladder Problem(0.260555671056)
- Length Of Last Word in a String(0.260555671056)
- Reverse words in a given string(0.220288150562)
- Group words with same set of characters(0.220288150562)
- Find the k most frequent words from a file(0.220288150562)

Longest Consecutive Sequence(128)

- Longest consecutive sequence in Binary tree(0.656972921033)
- Longest Consecutive Subsequence(0.503102612415)
- Delete consecutive same words in a sequence(0.411207055068)
- Length of the Longest Consecutive 1s in Binary Representation(0.318784021754)
- Find longest sequence of 1's in binary representation with one flip(0.318784021754)
- Find length of the longest consecutive path from a given starting character(0.291069102382)
- Recaman's sequence(0.260555671056)

- Padovan Sequence(0.260555671056)
- Look-and-Say Sequence(0.260555671056)
- Juggler Sequence(0.260555671056)

Sum Root to Leaf Numbers(129)

- Sum of all the numbers that are formed from root to leaf paths(0.709297266606)
- Root to leaf path sum equal to a given number(0.634808797178)
- Sum of two large numbers(0.411207055068)
- Sum of Perrin Numbers(0.411207055068)
- Sum of Fibonacci Numbers(0.411207055068)
- N-th root of a number(0.411207055068)
- Find number of subarrays with even sum(0.411207055068)
- Find cubic root of a number(0.411207055068)
- Fifth root of a number(0.411207055068)
- Even Fibonacci Numbers Sum(0.411207055068)

Surrounded Regions(130)

- Puzzle 64 | Surround the Villages(0.220288150562)
- Find length of the largest region in Boolean Matrix(0.194314340169)
- Given a matrix of ‘O’ and ‘X’, find the largest subsquare surrounded by ‘X’(0.133785092946)
- Given a matrix of ‘O’ and ‘X’, replace ‘O’ with ‘X’ if surrounded by ‘X’(0.101528524038)

Palindrome Partitioning(131)

- Print all palindromic partitions of a string(0.579738671538)

- Given a string, print all possible palindromic partitions(0.449436416524)
- Dynamic Programming | Set 17 (Palindrome Partitioning)(0.449436416524)
- Palindromic Primes(0.336096927276)
- Smallest Palindrome after replacement(0.260555671056)
- Partition a number into two divisble parts(0.260555671056)
- Palindrome Substring Queries(0.260555671056)
- Lexicographically first palindromic string(0.260555671056)
- Find a partition point in array(0.260555671056)
- Check if a number is Palindrome(0.260555671056)

Palindrome Partitioning II(132)

- Print all palindromic partitions of a string(0.411207055068)
- Given a string, print all possible palindromic partitions(0.318784021754)
- Dynamic Programming | Set 17 (Palindrome Partitioning)(0.318784021754)
- Palindromic Primes(0.260555671056)
- Smallest Palindrome after replacement(0.201993092498)
- Partition a number into two divisble parts(0.201993092498)
- Palindrome Substring Queries(0.201993092498)
- Lexicographically first palindromic string(0.201993092498)
- Find a partition point in array(0.201993092498)
- Check if a number is Palindrome(0.201993092498)

Clone Graph(133)

- Clone an Undirected Graph(0.709297266606)

- Graph and its representations(0.336096927276)
- Cloning in java(0.336096927276)
- Bridges in a graph(0.336096927276)
- Biconnected graph(0.336096927276)
- Transitive closure of a graph(0.260555671056)
- Sum of dependencies in a graph(0.260555671056)
- Graph implementation using STL for competitive programming | Set 2 (Weighted graph)(0.260555671056)
- Find k-cores of an undirected graph(0.260555671056)
- Find a Mother Vertex in a Graph(0.260555671056)

Gas Station(134)

- Minimum Number of Platforms Required for a Railway/Bus Station(0.175786078393)

Candy(135)

- Find the minimum and maximum amount to buy all N candies(0.335175743328)

Single Number(136)

- How can we sum the digits of a given number in single statement?(0.449436416524)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)

- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)

Single Number II(137)

- How can we sum the digits of a given number in single statement?(0.318784021754)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)
- Find the Number Occurring Odd Number of Times(0.285306190981)

Copy List with Random Pointer(138)

- Clone a linked list with next and random pointer | Set 2(0.407352604289)
- Clone a linked list with next and random pointer | Set 1(0.407352604289)
- Clone a linked list with next and random pointer in O(1) space(0.407352604289)
- Double Pointer (Pointer to Pointer) in C(0.316762744302)

- copy in Python (Deep Copy and Shallow Copy)(0.295058719041)
- Clone a Binary Tree with Random Pointers(0.291219418564)
- Select a Random Node from a Singly Linked List(0.260555671056)
- Point to next higher value node in a linked list with an arbitrary pointer(0.220288150562)
- How to write C functions that modify head pointer of a Linked List?(0.220288150562)
- When is copy constructor called?(0.220288150562)

Word Break(139)

- Word Break Problem using Backtracking(0.502328778226)
- Dynamic Programming | Set 32 (Word Break Problem)(0.410362644952)
- Word formation using concatenation of two dictionary words(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.291219418564)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Length Of Last Word in a String(0.260555671056)
- Reverse words in a given string(0.220288150562)
- Group words with same set of characters(0.220288150562)

Word Break II(140)

- Word Break Problem using Backtracking(0.356300429333)
- Dynamic Programming | Set 32 (Word Break Problem)(0.291069102382)
- Word formation using concatenation of two dictionary words(0.260555671056)

- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Length Of Last Word in a String(0.201993092498)
- Reverse words in a given string(0.17077611319)
- Group words with same set of characters(0.17077611319)

Linked List Cycle(141)

- Check if a linked list is Circular Linked List(0.580332984677)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Rotate a Linked List(0.503102612415)
- Merge a linked list into another linked list at alternate positions(0.503102612415)
- Identical Linked Lists(0.503102612415)
- Flattening a Linked List(0.503102612415)
- Can we reverse a linked list in less than O(n)?(0.503102612415)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.450268144656)
- XOR Linked List – A Memory Efficient Doubly Linked List | Set 2(0.429410856634)
- XOR Linked List – A Memory Efficient Doubly Linked List | Set 1(0.429410856634)

Linked List Cycle II(142)

- Check if a linked list is Circular Linked List(0.474330706497)

- In-place Merge two linked lists without changing links of first list(0.439404118785)
- Rotate a Linked List(0.411207055068)
- Merge a linked list into another linked list at alternate positions(0.411207055068)
- Identical Linked Lists(0.411207055068)
- Flattening a Linked List(0.411207055068)
- Can we reverse a linked list in less than O(n)?(0.411207055068)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.368023208756)
- XOR Linked List – A Memory Efficient Doubly Linked List | Set 2(0.350975664632)
- XOR Linked List – A Memory Efficient Doubly Linked List | Set 1(0.350975664632)

Reorder List(143)

- Recursively print all sentences that can be formed from list of word lists(0.311257467527)
- Check if a linked list is Circular Linked List(0.291219418564)
- Sublist Search (Search a linked list in another list)(0.274611786436)
- In-place Merge two linked lists without changing links of first list(0.274611786436)
- Sparse Matrix and its representations | Set 2 (Using List of Lists and Dictionary of keys)(0.260555671056)
- Rotate a Linked List(0.260555671056)
- Merge two sorted linked lists such that merged list is in reverse order(0.260555671056)
- Merge a linked list into another linked list at alternate posi-

tions(0.260555671056)

- List methods in Python(0.260555671056)
- Length of longest palindrome list in a linked list using O(1) extra space(0.260555671056)

Binary Tree Preorder Traversal(144)

- Construct Full Binary Tree from given preorder and postorder traversals(0.634808797178)
- Diagonal Traversal of Binary Tree(0.602974816038)
- Density of Binary Tree in One Traversal(0.602974816038)
- Boundary Traversal of binary tree(0.602974816038)
- Check if a given array can represent Preorder Traversal of Binary Search Tree(0.536892711852)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Tree Traversals (Inorder, Preorder and Postorder)(0.51014901931)
- Check if leaf traversal of two Binary Trees is same?(0.51014901931)
- Calculate depth of a full Binary tree from Preorder(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)

Binary Tree Postorder Traversal(145)

- Construct Full Binary Tree from given preorder and postorder traversals(0.634808797178)
- Diagonal Traversal of Binary Tree(0.602974816038)
- Density of Binary Tree in One Traversal(0.602974816038)
- Boundary Traversal of binary tree(0.602974816038)
- Check whether a binary tree is a full binary tree or not(0.519174772633)

- Tree Traversals (Inorder, Preorder and Postorder)(0.51014901931)
- Construct a Binary Tree from Postorder and Inorder(0.51014901931)
- Check if leaf traversal of two Binary Trees is same?(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)
- If you are given two traversal sequences, can you construct the binary tree?(0.450175502327)

LRU Cache(146)

- Implement LRU Cache(0.709297266606)
- Cache Memory(0.336096927276)
- Performance of loops (A caching question)(0.220288150562)
- What's difference between CPU Cache and TLB?(0.194314340169)
- Initializing and Cache Mechanism in Linux Kernel(0.194314340169)
- How to Implement Reverse DNS Look Up Cache?(0.194314340169)
- How to Implement Forward DNS Look Up Cache?(0.194314340169)
- Cache Organization | Set 1 (Introduction)(0.194314340169)
- Program for Page Replacement Algorithms | Set 1 (LRU)(0.161713780663)

Insertion Sort List(147)

- Insertion Sort(0.709297266606)
- Given a linked list which is sorted, how will you insert in sorted way(0.668731876126)
- Sorted insert for circular linked list(0.656972921033)
- Insertion Sort for Singly Linked List(0.656972921033)
- Recursive Insertion Sort(0.503102612415)

- Binary Insertion Sort(0.503102612415)
- Sort linked list which is already sorted on absolute values(0.418906716157)
- Minimum insertions to sort an array(0.411207055068)
- Merge two sorted linked lists(0.411207055068)
- Merge Sort for Linked Lists(0.411207055068)

Sort List(148)

- Sort linked list which is already sorted on absolute values(0.590594008858)
- Given a linked list which is sorted, how will you insert in sorted way(0.590594008858)
- Merge two sorted linked lists(0.579738671538)
- Merge Sort for Linked Lists(0.579738671538)
- Intersection of two Sorted Linked Lists(0.579738671538)
- Sort a linked list that is sorted alternating ascending and descending orders?(0.549988394922)
- Why Quick Sort preferred for Arrays and Merge Sort for Linked Lists?(0.51675016217)
- Sorted insert for circular linked list(0.502328778226)
- Sorted Linked List to Balanced BST(0.502328778226)
- Remove duplicates from a sorted linked list(0.502328778226)

Max Points on a Line(149)

- Minimum lines to cover all points(0.411207055068)
- Count maximum points on same line(0.411207055068)
- Non-crossing lines to connect points in a circle(0.356300429333)

- Find an Integer point on a line segment with given two ends(0.318784021754)
- Number of Integral Points between Two Points(0.318784021754)
- Write a one line C function to round floating point numbers(0.269517613246)
- Given a linked list of line segments, remove middle points(0.269517613246)
- Print level order traversal line by line | Set 1(0.225764846003)
- Prime points (Points that split a number into two primes)(0.225764846003)
- Level order traversal line by line | Set 2 (Using Two Queues)(0.212889950749)

Evaluate Reverse Polish Notation(150)

- Reversible numbers(0.220288150562)
- Expression Evaluation(0.220288150562)
- Reverse and Add Function(0.17077611319)
- Perfect reversible string(0.17077611319)
- Evaluation order of operands(0.17077611319)
- Evaluation of Expression Tree(0.17077611319)
- Can we reverse a linked list in less than O(n)?(0.17077611319)
- Reverse words in a given string(0.144383555277)
- Reverse a stack using recursion(0.144383555277)
- Reverse a Doubly Linked List(0.144383555277)

Reverse Words in a String(151)

- Reverse words in a given string(0.776514530475)
- Perfect reversible string(0.503102612415)
- Length Of Last Word in a String(0.503102612415)

- Count words in a given string(0.411207055068)
- String containing first letter of every word in a given string with spaces(0.390105265183)
- Write a program to reverse an array or string(0.356300429333)
- Reverse a string preserving space positions(0.356300429333)
- Program to find Smallest and Largest Word in a String(0.356300429333)
- Program to extract words from a given String(0.356300429333)
- Print reverse of a string using recursion(0.356300429333)

Maximum Product Subarray(152)

- Maximum Product Subarray(1.0)
- Maximum Product Subarray | Set 2 (Using Two Traversals)(0.524591090446)
- Sliding Window Maximum (Maximum of all subarrays of size k)(0.418906716157)
- Maximum circular subarray sum(0.411207055068)
- Maximum and Minimum Product Subsets(0.411207055068)
- Breaking an Integer to get Maximum Product(0.411207055068)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.366529477546)
- Number of subarrays for which product and sum are equal(0.356300429333)
- Maximum sum subarray removing at most one element(0.356300429333)
- Maximum subarray sum modulo m(0.356300429333)

Find Minimum in Rotated Sorted Array(153)

- Find the minimum element in a sorted and rotated array(0.818180207367)

- Find the Rotation Count in Rotated Sorted array(0.635001221407)
- Minimum insertions to sort an array(0.602974816038)
- Search an element in a sorted and rotated array(0.51014901931)
- Sort an array when two halves are sorted(0.465646219099)
- Minimum number of swaps required to sort an array(0.450175502327)
- Count minimum number of "move-to-front" moves to sort an array(0.450175502327)
- Find the Minimum length Unsorted Subarray, sorting which makes the complete array sorted(0.450058913045)
- Sort a nearly sorted (or K sorted) array(0.439015465545)
- Search in an almost sorted array(0.411207055068)

Find Minimum in Rotated Sorted Array II(154)

- Find the minimum element in a sorted and rotated array(0.669418851727)
- Find the Rotation Count in Rotated Sorted array(0.53724507516)
- Minimum insertions to sort an array(0.51014901931)
- Search an element in a sorted and rotated array(0.431613418971)
- Sort an array when two halves are sorted(0.403470577019)
- Minimum number of swaps required to sort an array(0.380872608476)
- Count minimum number of "move-to-front" moves to sort an array(0.380872608476)
- Find the Minimum length Unsorted Subarray, sorting which makes the complete array sorted(0.380773967693)
- Sort a nearly sorted (or K sorted) array(0.380395708026)
- Search in an almost sorted array(0.356300429333)

Min Stack(155)

- Spaghetti Stack(0.336096927276)
- Stack | Set 3 (Reverse a string using stack)(0.311257467527)
- Stack Unwinding in C++(0.260555671056)
- Stack Class in Java(0.260555671056)
- Implement two stacks in an array(0.260555671056)
- How to create mergable stack?(0.260555671056)
- Sort a stack using recursion(0.220288150562)
- Reverse a stack using recursion(0.220288150562)
- Implement Stack using Queues(0.220288150562)
- Implement Queue using Stacks(0.220288150562)

Binary Tree Upside Down(156)

- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)
- Diameter of a Binary Tree(0.503102612415)
- Bottom View of a Binary Tree(0.503102612415)

Read N Characters Given Read4(157)

- Count digits in given number N which divide N(0.296672366897)
- Given a string, find its first non-repeating character(0.291219418564)
- Nearest prime less than given number n(0.252334201434)
- Minimum sum of squares of character counts in a given string after removing k characters(0.245583963593)
- Legendre's formula (Given p and n, find the largest x such that p^x divides n!)(0.245583963593)
- Find the first non-repeating character from a stream of characters(0.237739238575)
- Construct a unique matrix n x n for an input n(0.231436178389)
- Program to count occurrence of a given character in a string(0.225764846003)
- Print * in place of characters for reading passwords in C(0.225764846003)
- Optimal read list for given number of days(0.225764846003)

Read N Characters Given Read4 II - Call multiple times(158)

- Count digits in given number N which divide N(0.224412943851)
- Multiple of x closest to n(0.220288150562)
- Given a string, find its first non-repeating character(0.220288150562)
- Convert given time into words(0.220288150562)
- Given a number n, count all multiples of 3 and/or 5 in set {1, 2, 3, … n}(0.204527164103)
- Given an array of of size n and a number k, find all elements that appear more than n/k times(0.203803708465)
- Count Fibonacci numbers in given range in O(Log n) time and O(1) space(0.203803708465)

- Nearest prime less than given number n(0.19087406613)
- Minimum sum of squares of character counts in a given string after removing k characters(0.18576795948)
- Legendre's formula (Given p and n, find the largest x such that p^x divides n!)(0.18576795948)

Longest Substring with At Most Two Distinct Characters(159)

- Length of the longest substring without repeating characters(0.51014901931)
- Longest Non-palindromic substring(0.411207055068)
- Count substrings with same first and last characters(0.411207055068)
- Find the longest substring with k unique characters in a given string(0.407352604289)
- Count number of substrings with exactly k distinct characters(0.407352604289)
- Longest repeating and non-overlapping substring(0.336096927276)
- Length of the longest valid substring(0.336096927276)
- Length of Longest sub-string that can be removed(0.336096927276)
- Longest Common Prefix | Set 2 (Character by Character Matching)(0.318849541433)
- Searching characters and substring in a String in Java(0.291219418564)

Intersection of Two Linked Lists(160)

- Union and Intersection of two Linked Lists(0.776514530475)
- Intersection of two Sorted Linked Lists(0.776514530475)
- Check if a linked list is Circular Linked List(0.580332984677)
- Write a function to get the intersection point of two Linked Lists.(0.579738671538)

- Union and Intersection of two linked lists | Set-3 (Hashing)(0.579738671538)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Rotate a Linked List(0.503102612415)
- Merge a linked list into another linked list at alternate positions(0.503102612415)
- Identical Linked Lists(0.503102612415)
- Flattening a Linked List(0.503102612415)

One Edit Distance(161)

- Check if edit distance between two strings is one(0.579738671538)
- Dynamic Programming | Set 5 (Edit Distance)(0.449436416524)
- Hamming Distance between two strings(0.260555671056)
- Find the minimum distance between two numbers(0.260555671056)
- Find Shortest distance from a guard in a Bank(0.220288150562)
- Print nodes at k distance from root(0.194314340169)
- Placements | QA | Trigonometry & Height and Distances(0.194314340169)
- Placements | QA | Time Speed Distance(0.194314340169)
- Minimum distance to travel to cover all intervals(0.194314340169)
- Maximum distance between two occurrences of same element in array(0.194314340169)

Find Peak Element(162)

- Find a peak element(1.0)
- Find a peak element in a 2D array(0.579738671538)

- Third largest element in an array of distinct elements(0.368023208756)
- Find the two non-repeating elements in an array of repeating elements(0.368023208756)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.368023208756)
- Find elements larger than half of the elements in an array(0.368023208756)
- Find all elements in array which have at-least two greater elements(0.368023208756)
- Elements before which no element is bigger in array(0.368023208756)
- Sum of all elements between k1'th and k2'th smallest elements(0.336096927276)
- Replace every element with the least greater element on its right(0.336096927276)

Missing Ranges(163)

- Find missing elements of a range(0.709297266606)
- Print missing elements that lie in range 0 – 99(0.379978361591)
- Find the Missing Number(0.336096927276)
- Bitwise and (or &) of a range(0.336096927276)
- Binary Indexed Tree : Range Update and Range Queries(0.311257467527)
- Range LCM Queries(0.260555671056)
- Perfect cubes in a range(0.260555671056)
- Find the smallest missing number(0.260555671056)
- range() vs xrange() in Python(0.220288150562)
- What are C++ features missing in Java?(0.220288150562)

Maximum Gap(164)

- Sliding Window Maximum (Maximum of all subarrays of size k)(0.311257467527)

- Maximum Product Subarray(0.260555671056)
- Maximum Bipartite Matching(0.260555671056)
- Find the maximum number of handshakes(0.260555671056)
- Type of array and its maximum element(0.220288150562)
- Sum of maximum elements of all subsets(0.220288150562)
- Subsequence with maximum odd sum(0.220288150562)
- Puzzle 22 | (Maximum Chocolates)(0.220288150562)
- Path with maximum average value(0.220288150562)
- Minimum and Maximum values of an expression with * and +(0.220288150562)

Compare Version Numbers(165)

- Compare two Version numbers(1.0)
- Comparable vs Comparator in Java(0.318784021754)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)

Fraction to Recurring Decimal(166)

- Find Recurring Sequence in a Fraction(0.503102612415)
- Convert Binary fraction to Decimal(0.411207055068)
- Convert decimal fraction to binary number(0.356300429333)
- Program to add two fractions(0.201993092498)
- Fractional Knapsack Problem(0.201993092498)
- Fraction module in Python(0.201993092498)
- Setting decimal precision in C(0.17077611319)
- Greedy Algorithm for Egyptian Fraction(0.17077611319)
- Remove recurring digits in a given number(0.150640184987)
- Quickly convert Decimal to other bases in Python(0.150640184987)

Two Sum II - Input array is sorted(167)

- Sort an array when two halves are sorted(0.403470577019)
- Count pairs in a sorted array whose sum is less than x(0.380872608476)
- Sort a nearly sorted (or K sorted) array(0.380395708026)
- Search in an almost sorted array(0.356300429333)
- Merge two sorted arrays(0.356300429333)
- Median of two sorted arrays(0.356300429333)
- Floor in a Sorted Array(0.356300429333)
- Ceiling in a sorted array(0.356300429333)
- Find original array from encrypted array (An array of sums of other elements)(0.35602438493)
- Check if a sorted array can be divided in pairs whose sum is k(0.344642141038)

Excel Sheet Column Title(168)

- Find Excel column number from column title(0.635001221407)
- Find Excel column name from a given column number(0.411065370983)
- Puzzle 40 | (Find missing Row in Excel)(0.127359529795)
- Find the largest rectangle of 1's with swapping of columns allowed(0.115215543378)
- Sum of matrix in which each element is absolute difference of its row and column numbers(0.0986796179799)
- Sorting 2D Vector in C++ | Set 3 (By number of columns)(0.0986796179799)
- Sorting 2D Vector in C++ | Set 1 (By row and column)(0.0986796179799)
- Replace every matrix element with maximum of GCD of row or column(0.0986796179799)
- Search in a row wise and column wise sorted matrix(0.0926978966863)
- Sum of matrix element where each elements is integer division of row and column(0.0876868198014)

Majority Element(169)

- Majority Element(1.0)
- Check for Majority Element in a sorted array(0.502328778226)
- Third largest element in an array of distinct elements(0.368023208756)
- Find the two non-repeating elements in an array of repeating elements(0.368023208756)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.368023208756)
- Find elements larger than half of the elements in an array(0.368023208756)
- Find all elements in array which have at-least two greater elements(0.368023208756)

- Elements before which no element is bigger in array(0.368023208756)
- Sum of all elements between k1'th and k2'th smallest elements(0.336096927276)
- Replace every element with the least greater element on its right(0.336096927276)

Two Sum III - Data structure design(170)

- Design an efficient data structure for given operations(0.380872608476)
- Persistent data structures(0.356300429333)
- Overview of Data Structures | Set 1 (Linear Data Structures)(0.356300429333)
- Data Mining(0.335175743328)
- Applications of tree data structure(0.291219418564)
- Applications of Queue Data Structure(0.291219418564)
- Applications of Heap Data Structure(0.291219418564)
- Design data structures for a very large social network like Facebook or Linkedln(0.277396228976)
- Design and Implement Special Stack Data Structure | Added Space Optimized Version(0.277396228976)
- Design a data structure that supports insert, delete, search and getRandom in constant time(0.277396228976)

Excel Sheet Column Number(171)

- Find Excel column number from column title(0.635001221407)
- Find Excel column name from a given column number(0.635001221407)
- Smallest number divisible by first n numbers(0.241213606675)
- Number with maximum number of prime factors(0.241213606675)
- Number of subtrees having odd count of even numbers(0.241213606675)

- Number of perfect squares between two given numbers(0.241213606675)
- Next higher number with same number of set bits(0.241213606675)
- How to check if a given number is Fibonacci number?(0.241213606675)
- Finding number of digits in n'th Fibonacci number(0.241213606675)
- Find the missing number in a string of numbers with no separator(0.241213606675)

Factorial Trailing Zeroes(172)

- Count trailing zeroes in factorial of a number(0.656972921033)
- Smallest number with at least n trailing zeroes in factorial(0.579738671538)
- Remove Trailing Zeros From string in C++(0.356300429333)
- Remove Trailing Zeros From String in Java(0.356300429333)
- Count trailing zero bits using lookup table(0.291069102382)
- Find the number of zeroes(0.260555671056)
- Double factorial(0.260555671056)
- Move all zeroes to end of array(0.201993092498)
- Last non-zero digit of a factorial(0.201993092498)
- Find all triplets with zero sum(0.201993092498)

Binary Search Tree Iterator(173)

- Iterative Search for a key 'x' in Binary Tree(0.709297266606)
- Binary Tree to Binary Search Tree Conversion(0.676628251794)
- Binary Search(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)

- Binary Search Tree | Set 1 (Search and Insertion)(0.519280018803)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Treap (A Randomized Binary Search Tree)(0.51014901931)
- Threaded Binary Search Tree | Deletion(0.51014901931)
- Merge Two Balanced Binary Search Trees(0.51014901931)
- Iterative Method to find Height of Binary Tree(0.51014901931)

Dungeon Game(174)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)
- Program for Conway's Game Of Life(0.220288150562)

Largest Number(179)

- Find the Largest number with given number of digits and sum of digits(0.51675016217)
- Largest subset whose all elements are Fibonacci numbers(0.502328778226)
- Largest palindrome which is product of two n-digit numbers(0.502328778226)

- Largest sum subarray with at-least k numbers(0.449436416524)
- Largest subarray with equal number of 0s and 1s(0.449436416524)
- Largest number smaller than or equal to n and digits in non-decreasing order(0.379978361591)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)

Reverse Words in a String II(186)

- Reverse words in a given string(0.602974816038)
- Perfect reversible string(0.411207055068)
- Length Of Last Word in a String(0.411207055068)
- Count words in a given string(0.336096927276)
- String containing first letter of every word in a given string with spaces(0.318849541433)
- Write a program to reverse an array or string(0.291219418564)
- Reverse a string preserving space positions(0.291219418564)
- Program to find Smallest and Largest Word in a String(0.291219418564)
- Program to extract words from a given String(0.291219418564)
- Print reverse of a string using recursion(0.291219418564)

Repeated DNA Sequences(187)

- Recaman's sequence(0.260555671056)
- Padovan Sequence(0.260555671056)

- Look-and-Say Sequence(0.260555671056)
- Juggler Sequence(0.260555671056)
- Farey Sequence(0.260555671056)
- Aliquot Sequence(0.260555671056)
- String with additive sequence(0.201993092498)
- Repeated subtraction among two numbers(0.201993092498)
- Longest Repeating Subsequence(0.201993092498)
- Jolly Jumper Sequence(0.201993092498)

Best Time to Buy and Sell Stock IV(188)

- Stock Buy Sell to Maximize Profit(0.380872608476)
- Maximum profit by buying and selling a share at most k times(0.304125741875)
- Maximum profit by buying and selling a share at most twice(0.201993092498)
- Changing One Clock Time to Other Time in Minimum Number of Operations(0.16279449512)
- Time Functions in Python | Set 1 (time(), ctime(), sleep()…)(0.152314155194)
- The Stock Span Problem(0.136276341439)
- An interesting time complexity question(0.136276341439)
- A Time Complexity Question(0.136276341439)
- [TopTalent.in] How Flipkart gets the best out of their applicants(0.115215543378)
- What to do at the time of Wrong Answer (WA)?(0.115215543378)

Rotate Array(189)

- Program for array rotation(0.709297266606)

- Find the Rotation Count in Rotated Sorted array(0.709052873586)
- Reversal algorithm for array rotation(0.579738671538)
- Program to cyclically rotate an array by one(0.579738671538)
- Search an element in a sorted and rotated array(0.502328778226)
- Find the minimum element in a sorted and rotated array(0.502328778226)
- Block swap algorithm for array rotation(0.502328778226)
- Maximum sum of i*arr[i] among all rotations of a given array(0.449436416524)
- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)

Reverse Bits(190)

- Write an Efficient C Program to Reverse Bits of a Number(0.410362644952)
- Toggle all the bits of a number except k-th bit.(0.368023208756)
- Reversible numbers(0.336096927276)
- Swap all odd and even bits(0.260555671056)
- Rotate bits of a number(0.260555671056)
- Reverse and Add Function(0.260555671056)
- Perfect reversible string(0.260555671056)
- Check if bits of a number has count of consecutive set bits in increasing order(0.260555671056)
- Can we reverse a linked list in less than O(n)?(0.260555671056)
- Bit Fields in C(0.260555671056)

Number of 1 Bits(191)

- Count total set bits in all numbers from 1 to n(0.524591090446)
- Rotate bits of a number(0.503102612415)
- Toggle all the bits of a number except k-th bit.(0.502929265114)
- Next higher number with same number of set bits(0.502929265114)
- Closest (or Next) smaller and greater numbers with same number of set bits(0.418906716157)
- Toggling k-th bit of a number(0.411207055068)
- Swap bits in a given number(0.411207055068)
- How to turn off a particular bit in a number?(0.411207055068)
- Check if two numbers are bit rotations of each other or not(0.411207055068)
- Add 1 to a given number(0.411207055068)

House Robber(198)

- Encrypt a string into the Rovarspraket (The Robber Language)(0.194314340169)
- Encrypt a string into the Rovarspraket (The Robber Language)(0.194314340169)

Binary Tree Right Side View(199)

- Print Right View of a Binary Tree(0.818180207367)
- Bottom View of a Binary Tree(0.776514530475)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Print Nodes in Top View of Binary Tree(0.51014901931)
- Print Left View of a Binary Tree(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)

- Binary Tree | Set 3 (Types of Binary Tree)(0.439404118785)
- fork() and Binary Tree(0.411207055068)
- Threaded Binary Tree(0.411207055068)
- Foldable Binary Trees(0.411207055068)

Number of Islands(200)

- Count number of islands where every island is row-wise and column-wise separated(0.549988394922)
- Find the number of islands | Set 1 (Using DFS)(0.449436416524)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)

Bitwise AND of Numbers Range(201)

- Bitwise and (or &) of a range(0.709297266606)
- Find numbers with n-divisors in a given range(0.411207055068)
- Querying maximum number of divisors that a number in a given range has(0.390105265183)
- Print all Good numbers in given range(0.356300429333)

- Numbers whose bitwise OR and sum with N are equal(0.356300429333)
- Count factorial numbers in a given range(0.356300429333)
- Number of elements with odd factors in given range(0.318784021754)
- Find the highest occurring digit in prime numbers in a range(0.318784021754)
- Find numbers with K odd divisors in a given range(0.318784021754)
- Russian Peasant (Multiply two numbers using bitwise operators)(0.291069102382)

Happy Number(202)

- Happy Number(1.0)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)

Remove Linked List Elements(203)

- Move last element to front of a given Linked List(0.602974816038)
- Remove duplicates from an unsorted linked list(0.51014901931)
- Remove duplicates from a sorted linked list(0.51014901931)

- Move all occurrences of an element to end in a linked list(0.51014901931)
- Detect and Remove Loop in a Linked List(0.51014901931)
- Pairwise swap elements of a given linked list by changing links(0.48097310796)
- Check if a linked list is Circular Linked List(0.474330706497)
- Search an element in a Linked List (Iterative and Recursive)(0.450175502327)
- Remove every k-th node of the linked list(0.450175502327)
- Remove all occurrences of duplicates from a sorted Linked List(0.450175502327)

Count Primes(204)

- Queries on the sum of prime factor counts in a range(0.449436416524)
- Count pairs with sum as a prime number and less than n(0.449436416524)
- Super Prime(0.336096927276)
- Right-Truncatable Prime(0.336096927276)
- Quick ways to check for Prime and find next Prime in Java(0.336096927276)
- Palindromic Primes(0.336096927276)
- Mersenne Prime(0.336096927276)
- Left-Truncatable Prime(0.336096927276)
- Find the prime numbers which can written as sum of most consecutive primes(0.336096927276)
- Counting Sort(0.336096927276)

Isomorphic Strings(205)

- Check if two given strings are isomorphic to each other(0.579738671538)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)

- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)
- Check if given string can be split into four distinct strings(0.336096927276)
- Sort a string according to the order defined by another string(0.311257467527)

Reverse Linked List(206)

- Can we reverse a linked list in less than O(n)?(1.0)
- Reverse a Doubly Linked List(0.776514530475)
- Write a function to reverse a linked list(0.656972921033)
- Check if a linked list is Circular Linked List(0.580332984677)
- Reverse a Linked List in groups of given size(0.579738671538)
- Merge two sorted linked lists such that merged list is in reverse order(0.54658286128)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Write a recursive function to print reverse of a Linked List(0.524591090446)
- Reverse alternate K nodes in a Singly Linked List(0.524591090446)
- Rotate a Linked List(0.503102612415)

Course Schedule(207)

- Weighted Job Scheduling(0.260555671056)
- Operating System | Process Scheduler(0.260555671056)
- Disk Scheduling Algorithms(0.260555671056)
- DBMS | Recoverability of Schedules(0.260555671056)
- Project Idea | (Online Course Registration)(0.194314340169)
- Program for Priority Scheduling | Set 1(0.194314340169)
- Program for FCFS Scheduling | Set 1(0.194314340169)
- Operating Systems | CPU Scheduling | Question 6(0.194314340169)
- Operating Systems | CPU Scheduling | Question 5(0.194314340169)
- Operating Systems | CPU Scheduling | Question 4(0.194314340169)

Implement Trie (Prefix Tree)(208)

- Palindromic Tree | Introduction & Implementation(0.336096927276)
- Overview of Data Structures | Set 3 (Graph, Trie, Segment Tree and Suffix Tree)(0.269636772416)
- Convert a given tree to its Sum Tree(0.241213606675)
- Binary Indexed Tree or Fenwick Tree(0.241213606675)
- Longest Common Prefix | Set 5 (Using Trie)(0.237903094633)
- Prefix Sum Array – Implementation and Applications in Competitive Programming(0.220288150562)
- Longest prefix matching – A Trie based solution in Java(0.220288150562)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.220288150562)
- Trie | (Delete)(0.220288150562)
- Tree Sort(0.220288150562)

Minimum Size Subarray Sum(209)

- Find maximum (or minimum) sum of a subarray of size k(0.709297266606)
- Sum of minimum and maximum elements of all subarrays of size k.(0.634808797178)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.622892268251)
- Sum of all Subarrays(0.579738671538)
- Smallest subarray whose sum is multiple of array size(0.450175502327)
- Maximum sum two non-overlapping subarrays of given size(0.450175502327)
- Find number of subarrays with even sum(0.411207055068)
- Find if there is a subarray with 0 sum(0.411207055068)
- Print all subarrays with 0 sum(0.336096927276)
- Maximum circular subarray sum(0.336096927276)

Course Schedule II(210)

- Weighted Job Scheduling(0.201993092498)
- Operating System | Process Scheduler(0.201993092498)
- Disk Scheduling Algorithms(0.201993092498)
- DBMS | Recoverability of Schedules(0.201993092498)
- Project Idea | (Online Course Registration)(0.150640184987)
- Program for Priority Scheduling | Set 1(0.150640184987)
- Program for FCFS Scheduling | Set 1(0.150640184987)
- Operating Systems | CPU Scheduling | Question 6(0.150640184987)
- Operating Systems | CPU Scheduling | Question 5(0.150640184987)
- Operating Systems | CPU Scheduling | Question 4(0.150640184987)

Add and Search Word - Data structure design(211)

- Design a data structure that supports insert, delete, search and getRandom in constant time(0.356300429333)
- Design an efficient data structure for given operations(0.336096927276)
- Persistent data structures(0.318784021754)
- Overview of Data Structures | Set 1 (Linear Data Structures)(0.318784021754)
- Data Structures | Binary Search Trees | Question 8(0.304125741875)
- Data Structures | Binary Search Trees | Question 7(0.304125741875)
- Data Structures | Binary Search Trees | Question 6(0.304125741875)
- Data Structures | Binary Search Trees | Question 5(0.304125741875)
- Data Structures | Binary Search Trees | Question 4(0.304125741875)
- Data Structures | Binary Search Trees | Question 3(0.304125741875)

Word Search II(212)

- Search a Word in a 2D Grid of characters(0.356300429333)
- Best First Search (Informed Search)(0.318784021754)
- Linear Search vs Binary Search(0.285306190981)
- Interpolation search vs Binary search(0.285306190981)
- Anagram Substring Search (Or Search for all permutations)(0.285306190981)
- Word formation using concatenation of two dictionary words(0.260555671056)
- Why is Binary Search preferred over Ternary Search?(0.260555671056)
- Linear Search(0.260555671056)
- Jump Search(0.260555671056)
- Interpolation Search(0.260555671056)

House Robber II(213)

- Encrypt a string into the Rovarspraket (The Robber Language)(0.150640184987)
- Encrypt a string into the Rovarspraket (The Robber Language)(0.150640184987)
- Flipkart Interview | Set 7 (For SDE II)(0.136276341439)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.125366937987)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.125366937987)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.125366937987)
- Amazon Interview experience | Set 326 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 348 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 313 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 312 (For SDE II)(0.125366937987)

Shortest Palindrome(214)

- Minimum insertions to form shortest palindrome(0.502328778226)
- Palindromic Primes(0.336096927276)
- Smallest Palindrome after replacement(0.260555671056)
- Shortest Uncommon Subsequence(0.260555671056)
- Shortest Superstring Problem(0.260555671056)
- Shortest Common Supersequence(0.260555671056)
- Palindrome Substring Queries(0.260555671056)
- Lexicographically first palindromic string(0.260555671056)
- Check if a number is Palindrome(0.260555671056)
- Shortest path in a Binary Maze(0.220288150562)

Kth Largest Element in an Array(215)

- Find the largest three elements in an array(0.776514530475)
- Third largest element in an array of distinct elements(0.635001221407)
- Program to find largest element in an array(0.602974816038)
- K-th Element of Two Sorted Arrays(0.602974816038)
- Find Second largest element in an array(0.602974816038)
- k-th distinct (or non-repeating) element in an array.(0.51014901931)
- k-th smallest absolute difference of two elements in an array(0.450175502327)
- Construct an array from XOR of all elements of array except element at same index(0.439404118785)
- For each element in 1st array count elements less than or equal to it in 2nd array(0.411207055068)
- Find the two non-repeating elements in an array of repeating elements(0.411065370983)

Combination Sum III(216)

- Find sum of sum of all sub-sequences(0.36771998047)
- Sum of all Subarrays(0.260555671056)
- Find maximum sum possible equal sum of three stacks(0.260555671056)
- Combinations with repetitions(0.260555671056)
- Print all possible sums of consecutive numbers with sum N(0.241299136472)
- Perfect Sum Problem (Print all subsets with given sum)(0.241299136472)
- Print all n-digit numbers whose sum of digits equals to given sum(0.225764846003)
- Finding sum of digits of a number until sum becomes single digit(0.212889950749)

- Sum of two large numbers(0.201993092498)
- Sum of subset differences(0.201993092498)

Contains Duplicate(217)

- Check if a given array contains duplicate elements within k distance from each other(0.379978361591)
- Check if a Binary Tree contains duplicate subtrees of size 2 or more(0.379978361591)
- Find duplicates under given constraints(0.260555671056)
- AVL with duplicate keys(0.260555671056)
- Remove duplicates from sorted array(0.220288150562)
- Remove all duplicates from a given string(0.220288150562)
- Recursively remove all adjacent duplicates(0.220288150562)
- Print all the duplicates in the input string(0.220288150562)
- How to print duplicate rows in a table?(0.220288150562)
- Find lost element from a duplicated array(0.220288150562)

The Skyline Problem(218)

- Divide and Conquer | Set 7 (The Skyline Problem)(0.449436416524)
- Tiling Problem(0.336096927276)
- The Celebrity Problem(0.336096927276)
- Nuts & Bolts Problem (Lock & Key problem)(0.336096927276)
- Gold Mine Problem(0.336096927276)
- Tree Isomorphism Problem(0.260555671056)
- The Stock Span Problem(0.260555671056)

- The Lazy Caterer's Problem(0.260555671056)
- Steiner Tree Problem(0.260555671056)
- Stable Marriage Problem(0.260555671056)

Contains Duplicate II(219)

- Check if a given array contains duplicate elements within k distance from each other(0.269517613246)
- Check if a Binary Tree contains duplicate subtrees of size 2 or more(0.269517613246)
- Find duplicates under given constraints(0.201993092498)
- AVL with duplicate keys(0.201993092498)
- Remove duplicates from sorted array(0.17077611319)
- Remove all duplicates from a given string(0.17077611319)
- Recursively remove all adjacent duplicates(0.17077611319)
- Print all the duplicates in the input string(0.17077611319)
- How to print duplicate rows in a table?(0.17077611319)
- Find lost element from a duplicated array(0.17077611319)

Contains Duplicate III(220)

- Check if a given array contains duplicate elements within k distance from each other(0.269517613246)
- Check if a Binary Tree contains duplicate subtrees of size 2 or more(0.269517613246)
- Find duplicates under given constraints(0.201993092498)
- AVL with duplicate keys(0.201993092498)
- Remove duplicates from sorted array(0.17077611319)

- Remove all duplicates from a given string(0.17077611319)
- Recursively remove all adjacent duplicates(0.17077611319)
- Print all the duplicates in the input string(0.17077611319)
- How to print duplicate rows in a table?(0.17077611319)
- Find lost element from a duplicated array(0.17077611319)

Maximal Square(221)

- Magic Square(0.336096927276)
- Latin Square(0.336096927276)
- Square root of an integer(0.260555671056)
- Direction at last square block(0.260555671056)
- Program to find number of squares in a chessboard(0.220288150562)
- Nth Square free number(0.220288150562)
- Maximum and Minimum in a square matrix.(0.220288150562)
- Maximizing Unique Pairs from two arrays(0.220288150562)
- Count number of squares in a rectangle(0.220288150562)
- Babylonian method for square root(0.220288150562)

Count Complete Tree Nodes(222)

- Program to count leaf nodes in a binary tree(0.450175502327)
- Count full nodes in a Binary tree (Iterative and Recursive)(0.450175502327)
- Iterative program to count leaf nodes in a Binary Tree(0.407352604289)
- Count half nodes in a Binary tree (Iterative and Recursive)(0.407352604289)
- Print all full nodes in a Binary Tree(0.336096927276)

- Get Level of a node in a Binary Tree(0.336096927276)
- Find the Deepest Node in a Binary Tree(0.336096927276)
- Convert a tree to forest of even nodes(0.336096927276)
- Check if two nodes are on same path in a tree(0.336096927276)
- Check whether a binary tree is a complete tree or not | Set 2 (Recursive Solution)(0.299580052534)

Rectangle Area(223)

- Find if two rectangles overlap(0.336096927276)
- Choice of Area(0.336096927276)
- program to find area of a circle(0.260555671056)
- Count number of squares in a rectangle(0.220288150562)
- Check if four segments form a rectangle(0.220288150562)
- C program to find area of a triangle(0.220288150562)
- Local Area Network (LAN) Technologies.(0.194314340169)
- Summed Area Table – Submatrix Summation(0.175786078393)
- Maximum size rectangle binary sub-matrix with all 1s(0.175786078393)
- Maximum area of triangle having different vertex colors(0.175786078393)

Basic Calculator(224)

- Basics of Wi-Fi(0.336096927276)
- JavaScript Backend basics(0.260555671056)
- Calculate Logn in one line(0.260555671056)
- Basic Operators in Java(0.260555671056)

- Write a program to calculate pow(x,n)(0.220288150562)
- Simple Calculator via UDP in Java(0.220288150562)
- Efficient program to calculate e^x(0.220288150562)
- Creating a Calculator for Android devices(0.220288150562)
- Calculate XOR from 1 to n.(0.220288150562)
- Building a Basic Chrome Extension(0.220288150562)

Implement Stack using Queues(225)

- Implement Stack using Queues(1.0)
- Implement Queue using Stacks(1.0)
- Implement a stack using single queue(0.818180207367)
- How to implement stack using priority queue or heap?(0.709297266606)
- Implement two stacks in an array(0.411207055068)
- Stack | Set 3 (Reverse a string using stack)(0.342390186113)
- Sort a stack using recursion(0.336096927276)
- Reverse a stack using recursion(0.336096927276)
- Implement rand3() using rand2()(0.336096927276)
- Implementation of Deque using circular array(0.291219418564)

Invert Binary Tree(226)

- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)

- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)
- Diameter of a Binary Tree(0.503102612415)
- Bottom View of a Binary Tree(0.503102612415)

Basic Calculator II(227)

- Basics of Wi-Fi(0.260555671056)
- JavaScript Backend basics(0.201993092498)
- Calculate Logn in one line(0.201993092498)
- Basic Operators in Java(0.201993092498)
- Write a program to calculate pow(x,n)(0.17077611319)
- Simple Calculator via UDP in Java(0.17077611319)
- Efficient program to calculate e^x(0.17077611319)
- Creating a Calculator for Android devices(0.17077611319)
- Calculate XOR from 1 to n.(0.17077611319)
- Building a Basic Chrome Extension(0.17077611319)

Summary Ranges(228)

- Bitwise and (or &) of a range(0.336096927276)
- Binary Indexed Tree : Range Update and Range Queries(0.311257467527)
- Range LCM Queries(0.260555671056)
- Perfect cubes in a range(0.260555671056)

- Find missing elements of a range(0.260555671056)
- range() vs xrange() in Python(0.220288150562)
- Min-Max Range Queries in Array(0.220288150562)
- Find the smallest twins in given range(0.220288150562)
- Find numbers with n-divisors in a given range(0.220288150562)
- Copy set bits in a range(0.220288150562)

Majority Element II(229)

- Majority Element(0.709297266606)
- Check for Majority Element in a sorted array(0.356300429333)
- Third largest element in an array of distinct elements(0.285306190981)
- Find the two non-repeating elements in an array of repeating elements(0.285306190981)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.285306190981)
- Find elements larger than half of the elements in an array(0.285306190981)
- Find all elements in array which have at-least two greater elements(0.285306190981)
- Elements before which no element is bigger in array(0.285306190981)
- Sum of all elements between k1'th and k2'th smallest elements(0.260555671056)
- Replace every element with the least greater element on its right(0.260555671056)

Kth Smallest Element in a BST(230)

- Find k-th smallest element in BST (Order Statistics in BST)(0.755474439422)
- Sum of k smallest elements in BST(0.51014901931)

- k-th smallest absolute difference of two elements in an array(0.450175502327)
- Find the smallest and second smallest elements in an array(0.411065370983)
- K-th smallest element after removing some integers from natural numbers(0.407352604289)
- K'th smallest element in BST using O(1) Extra Space(0.374807770059)
- Sum of all elements between k1'th and k2'th smallest elements(0.372055731454)
- K'th Largest Element in BST when modification to BST is not allowed(0.342390186113)
- Second largest element in BST(0.336096927276)
- Maximum element between two nodes of BST(0.336096927276)

Power of Two(231)

- Find power of power under mod of a prime(0.709297266606)
- Program to find whether a no is power of two(0.579738671538)
- Powerful Number(0.579738671538)
- Power Set(0.579738671538)
- Time Complexity of Loop with Powers(0.379978361591)
- Print all prime factors and their powers(0.379978361591)
- Find whether a given number is a power of 4 or not(0.379978361591)
- Write you own Power without using multiplication(*) and division(/) operators(0.30321606445)
- Smallest power of 2 greater than or equal to n(0.30321606445)
- Highest power of 2 less than or equal to given number(0.30321606445)

Implement Queue using Stacks(232)

- Implement Stack using Queues(1.0)

- Implement Queue using Stacks(1.0)
- Implement a stack using single queue(0.818180207367)
- How to implement stack using priority queue or heap?(0.709297266606)
- Implement two stacks in an array(0.411207055068)
- Stack | Set 3 (Reverse a string using stack)(0.342390186113)
- Sort a stack using recursion(0.336096927276)
- Reverse a stack using recursion(0.336096927276)
- Implement rand3() using rand2()(0.336096927276)
- Implementation of Deque using circular array(0.291219418564)

Number of Digit One(233)

- Find the Largest number with given number of digits and sum of digits(0.757934808143)
- Find smallest number with given number of digits and sum of digits(0.757934808143)
- Count numbers with same first and last digits(0.709297266606)
- Finding number of digits in n'th Fibonacci number(0.709052873586)
- Find count of digits in a number that divide the number(0.709052873586)
- Total number of non-decreasing numbers with n digits(0.641764556549)
- Smallest number by rearranging digits of a given number(0.641764556549)
- Count total number of N digit numbers such that the difference between sum of even and odd digits is 1(0.605403230565)
- Number of occurrences of 2 as a digit in numbers from 0 to n(0.590594008858)
- Find the smallest number whose digits multiply to a given number n(0.590594008858)

Palindrome Linked List(234)

- Check linked list with a loop is palindrome or not(0.656972921033)
- Check if a linked list is Circular Linked List(0.580332984677)
- Function to check if a singly linked list is palindrome(0.579738671538)
- Check if a linked list of strings forms a palindrome(0.579738671538)
- Length of longest palindrome list in a linked list using O(1) extra space(0.54658286128)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Rotate a Linked List(0.503102612415)
- Merge a linked list into another linked list at alternate positions(0.503102612415)
- Identical Linked Lists(0.503102612415)
- Flattening a Linked List(0.503102612415)

Lowest Common Ancestor of a Binary Search Tree(235)

- Lowest Common Ancestor in a Binary Search Tree.(1.0)
- Lowest Common Ancestor in a Binary Tree | Set 1(0.632790458368)
- Binary Tree to Binary Search Tree Conversion(0.505164486208)
- Print Common Nodes in Two Binary Search Trees(0.503102612415)
- Lowest Common Ancestor in a Binary Tree | Set 2 (Using Parent Pointer)(0.490834212066)
- Binary Search(0.449436416524)
- Minimum swap required to convert binary tree to binary search tree(0.407081366967)
- Check whether a binary tree is a full binary tree or not(0.402484879511)

- Binary Search Tree | Set 1 (Search and Insertion)(0.38768972948)
- Treap (A Randomized Binary Search Tree)(0.380872608476)

Lowest Common Ancestor of a Binary Tree(236)

- Lowest Common Ancestor in a Binary Search Tree.(0.84664735365)
- Lowest Common Ancestor in a Binary Tree | Set 1(0.747407354006)
- Lowest Common Ancestor in a Binary Tree | Set 2 (Using Parent Pointer)(0.579738671538)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Binary Tree to Binary Search Tree Conversion(0.410995463935)
- Tarjan's off-line lowest common ancestors algorithm(0.380872608476)
- Print Common Nodes in Two Binary Search Trees(0.380872608476)
- Print Ancestors of a given node in Binary Tree(0.380872608476)
- Maximum difference between node and its ancestor in Binary Tree(0.380872608476)
- Construct Ancestor Matrix from a Given Binary Tree(0.380872608476)

Delete Node in a Linked List(237)

- Delete alternate nodes of a Linked List(0.818180207367)
- Delete a node in a Doubly Linked List(0.818180207367)
- Delete N nodes after M nodes of a linked list(0.755474439422)
- Linked List | Set 3 (Deleting a node)(0.709297266606)
- Delete a Linked List node at a given position(0.709297266606)
- Given only a pointer/reference to a node to be deleted in a singly linked list, how do you delete it?(0.647630825181)

- Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?(0.647630825181)
- Deletion from a Circular Linked List(0.602974816038)
- Delete middle of linked list(0.602974816038)
- Delete a given node in Linked List under given constraints(0.536892711852)

Product of Array Except Self(238)

- A Product Array Puzzle(0.503102612415)
- First digit in product of an array of numbers(0.411207055068)
- Find pair with greatest product in array(0.411207055068)
- Sum of product of all pairs of array elements(0.356300429333)
- Find a pair with maximum product in array of Integers(0.356300429333)
- Count pairs whose products exist in array(0.356300429333)
- Find original array from encrypted array (An array of sums of other elements)(0.327966201641)
- Minimize the sum of product of two arrays with permutations allowed(0.318784021754)
- Find Maximum dot product of two arrays with insertion of 0's(0.318784021754)
- Check if product of every pair exists in an array(0.318784021754)

Sliding Window Maximum(239)

- Sliding Window Maximum (Maximum of all subarrays of size k)(0.668731876126)
- Window Sliding Technique(0.503102612415)
- Sliding Window Protocol | Set 2 (Receiver Side)(0.318784021754)
- Sliding Window Protocol | Set 1 (Sender Side)(0.318784021754)

- Find maximum of minimum for every window size in a given array(0.291069102382)
- Maximum Product Subarray(0.201993092498)
- Maximum Bipartite Matching(0.201993092498)
- Find the maximum number of handshakes(0.201993092498)
- Windows 10 –Feel the Difference(0.17077611319)
- Type of array and its maximum element(0.17077611319)

Search a 2D Matrix II(240)

- Search element in a sorted matrix(0.336096927276)
- Search a Word in a 2D Grid of characters(0.291219418564)
- Best First Search (Informed Search)(0.269517613246)
- Linear Search vs Binary Search(0.241213606675)
- Interpolation search vs Binary search(0.241213606675)
- Anagram Substring Search (Or Search for all permutations)(0.241213606675)
- Print 2D matrix in different lines and without curly braces in C/C++?(0.220288150562)
- Why is Binary Search preferred over Ternary Search?(0.220288150562)
- Queries in a Matrix(0.220288150562)
- Matrix Introduction(0.220288150562)

Different Ways to Add Parentheses(241)

- All ways to add parenthesis for evaluation(0.336096927276)
- Different ways to create objects in Java(0.291219418564)
- Different ways of Method Overloading in Java(0.291219418564)

- Reverse a string in Java (5 Different Ways)(0.260555671056)
- Different ways of Reading a text file in Java(0.260555671056)
- Different ways for Integer to String Conversions In Java(0.260555671056)
- 3 Different ways to print Exception messages in Java(0.260555671056)
- Different ways to delete elements in std::map (erase() and clear())(0.237903094633)
- Array of Strings in C++ (3 Different Ways to Create)(0.237903094633)
- Ways to arrange Balls such that adjacent balls are of different types(0.206083635014)

Valid Anagram(242)

- Check whether two strings are anagram of each other(0.260555671056)
- Valid variants of main() in Java(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Length of the longest valid substring(0.220288150562)
- Count of total anagram substrings(0.220288150562)
- is_permutation() in C++ and its application for anagram search(0.194314340169)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)
- How to check if a string is a valid keyword in Python?(0.194314340169)

Shortest Word Distance(243)

- Find Shortest distance from a guard in a Bank(0.411207055068)
- Word Ladder (Length of shortest chain to reach a target word)(0.390105265183)

- Word formation using concatenation of two dictionary words(0.260555671056)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Find shortest unique prefix for every word in a given list | Set 2 (Using Sorting)(0.225554872207)
- Find shortest unique prefix for every word in a given list | Set 1 (Using Trie)(0.225554872207)
- Testimonials – Words that keep us going(0.201993092498)
- Shortest Uncommon Subsequence(0.201993092498)
- Shortest Superstring Problem(0.201993092498)

Shortest Word Distance II(244)

- Find Shortest distance from a guard in a Bank(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.318849541433)
- Word formation using concatenation of two dictionary words(0.220288150562)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.19087406613)
- C program to Replace a word in a text by another given word(0.19087406613)
- Find shortest unique prefix for every word in a given list | Set 2 (Using Sorting)(0.184355541926)
- Find shortest unique prefix for every word in a given list | Set 1 (Using Trie)(0.184355541926)
- Testimonials – Words that keep us going(0.17077611319)
- Shortest Uncommon Subsequence(0.17077611319)
- Shortest Superstring Problem(0.17077611319)

Shortest Word Distance III(245)

- Find Shortest distance from a guard in a Bank(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.318849541433)
- Word formation using concatenation of two dictionary words(0.220288150562)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.19087406613)
- C program to Replace a word in a text by another given word(0.19087406613)
- Find shortest unique prefix for every word in a given list | Set 2 (Using Sorting)(0.184355541926)
- Find shortest unique prefix for every word in a given list | Set 1 (Using Trie)(0.184355541926)
- Testimonials – Words that keep us going(0.17077611319)
- Shortest Uncommon Subsequence(0.17077611319)
- Shortest Superstring Problem(0.17077611319)

Strobogrammatic Number(246)

- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)

- Find count of digits in a number that divide the number(0.368023208756)

Strobogrammatic Number II(247)

- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)
- Find the Number Occurring Odd Number of Times(0.285306190981)
- Find count of digits in a number that divide the number(0.285306190981)

Strobogrammatic Number III(248)

- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)

- Find the Number Occurring Odd Number of Times(0.285306190981)
- Find count of digits in a number that divide the number(0.285306190981)

Group Shifted Strings(249)

- Group Shifted String(1.0)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Pairs of complete strings in two sets of strings(0.285306190981)
- Given two strings, find if first string is a subsequence of second(0.285306190981)
- String matching where one string contains wildcard characters(0.260555671056)
- Sort an array of strings according to string lengths(0.260555671056)
- Search in an array of strings where non-empty strings are sorted(0.260555671056)
- SQL | GROUP BY(0.260555671056)
- Remove characters from the first string which are present in the second string(0.260555671056)
- Check if given string can be split into four distinct strings(0.260555671056)

Count Univalue Subtrees(250)

- Find Count of Single Valued Subtrees(0.411207055068)
- Count BST subtrees that lie in given range(0.318784021754)
- Number of subtrees having odd count of even numbers(0.291069102382)
- Find largest subtree having identical left and right subtrees(0.260555671056)
- Counting Sort(0.260555671056)
- Count substrings with same first and last characters(0.201993092498)

- Count of parallelograms in a plane(0.201993092498)
- Count numbers with same first and last digits(0.201993092498)
- Count all increasing subsequences(0.201993092498)
- Count Divisors of Factorial(0.201993092498)

Flatten 2D Vector(251)

- Sorting 2D Vector in C++ | Set 3 (By number of columns)(0.269517613246)
- Sorting 2D Vector in C++ | Set 1 (By row and column)(0.269517613246)
- Sorting 2D Vector in C++ | Set 2 (In descending order by row and column)(0.237739238575)
- Vector in C++ STL(0.201993092498)
- How to transform Vector into String?(0.201993092498)
- Flattening a Linked List(0.201993092498)
- Ways to copy a vector in C++(0.17077611319)
- Vector vs ArrayList in Java(0.17077611319)
- Placements | QA | Mensuration 2D(0.17077611319)
- Flatten a multilevel linked list(0.17077611319)

Meeting Rooms(252)

- Meet in the middle(0.336096927276)
- Simple Chat Room using Python(0.194314340169)
- OYO Rooms Interview Experience for Software Engineer(0.175786078393)
- OYO Rooms Interview Experience | Set 2 (For Fresher)(0.161713780663)
- Maximum points collected by two persons allowed to meet once(0.161713780663)

- OYO Rooms Interview Experience | Set 5 (Off-Campus for SDE)(0.150556969602)
- OYO Rooms Interview Experience | Set 4 (For Backend Profile)(0.150556969602)
- OYO Rooms Interview Experience | Set 6 (For Senior Software Developer)(0.141430567926)
- OYO Rooms Interview Experience | Set 6 (For Senior Software Developer)(0.141430567926)
- Oyo Rooms Interview Experience | Set 3 (For Backend Engineer, Experience <=1yrs)(0.121603314786)

Meeting Rooms II(253)

- Meet in the middle(0.260555671056)
- Simple Chat Room using Python(0.150640184987)
- OYO Rooms Interview Experience for Software Engineer(0.136276341439)
- Flipkart Interview | Set 7 (For SDE II)(0.136276341439)
- OYO Rooms Interview Experience | Set 2 (For Fresher)(0.125366937987)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.125366937987)
- Maximum points collected by two persons allowed to meet once(0.125366937987)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.125366937987)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.125366937987)
- Amazon Interview experience | Set 326 (For SDE II)(0.125366937987)

Factor Combinations(254)

- No of Factors of n!(0.336096927276)
- Combinations with repetitions(0.336096927276)
- Permutation and Combination in Python(0.260555671056)

- Print all prime factors and their powers(0.220288150562)
- Print all combinations of balanced parentheses(0.220288150562)
- Placements | QA | Permutation and Combination(0.220288150562)
- k-th prime factor of a given number(0.194314340169)
- Sort elements on the basis of number of factors(0.194314340169)
- Print the kth common factor of two numbers(0.194314340169)
- Prime factors of LCM of array elements(0.194314340169)

Verify Preorder Sequence in Binary Search Tree(255)

- Binary Tree to Binary Search Tree Conversion(0.505164486208)
- Leaf nodes from Preorder of a Binary Search Tree(0.503102612415)
- Binary Search(0.449436416524)
- Minimum swap required to convert binary tree to binary search tree(0.407081366967)
- Check whether a binary tree is a full binary tree or not(0.402484879511)
- Binary Search Tree | Set 1 (Search and Insertion)(0.38768972948)
- Treap (A Randomized Binary Search Tree)(0.380872608476)
- Threaded Binary Search Tree | Deletion(0.380872608476)
- Merge Two Balanced Binary Search Trees(0.380872608476)
- Longest consecutive sequence in Binary tree(0.380872608476)

Paint House(256)

- Flood fill Algorithm – how to implement fill() in paint?(0.194314340169)

Binary Tree Paths(257)

- Print all k-sum paths in a binary tree(0.656972921033)

- Maximum Path Sum in a Binary Tree(0.656972921033)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Given a binary tree, print all root-to-leaf paths(0.579738671538)
- Find the maximum path sum between two leaves of a binary tree(0.579738671538)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- Root to leaf paths having equal lengths in a Binary Tree(0.524591090446)
- Print path from root to a given node in a binary tree(0.524591090446)
- Print path from root to a given node in a binary tree(0.524591090446)

Add Digits(258)

- Check if frequency of each digit is less than the digit(0.411207055068)
- Count 'd' digit positive integers with 0 as a digit(0.311257467527)
- Generate k digit numbers with digits in strictly increasing order(0.291219418564)
- Digital Root (repeated digital sum) of the given large integer(0.291219418564)
- Count positive integers with 0 as a digit and maximum 'd' digits(0.291219418564)
- Finding sum of digits of a number until sum becomes single digit(0.274611786436)
- Find the Largest number with given number of digits and sum of digits(0.274611786436)
- Find smallest number with given number of digits and sum of digits(0.274611786436)
- Reverse and Add Function(0.260555671056)

- Program to add two polynomials(0.260555671056)

3Sum Smaller(259)

- Find next Smaller of next Greater in an array(0.260555671056)
- Find the closest and smaller tidy number(0.220288150562)
- Count smaller elements on right side(0.220288150562)
- Find the nearest smaller numbers on left side in an array(0.194314340169)
- Sieve of Sundaram to print all primes smaller than n(0.175786078393)
- Count triplets with sum smaller than a given value(0.175786078393)
- Count of smaller or equal elements in sorted array(0.175786078393)
- Count of Binary Digit numbers smaller than N(0.175786078393)
- Print all Jumping Numbers smaller than or equal to a given value(0.161713780663)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.161713780663)

Single Number III(260)

- How can we sum the digits of a given number in single statement?(0.318784021754)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)

- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)
- Find the Number Occurring Odd Number of Times(0.285306190981)

Graph Valid Tree(261)

- Check if a given graph is tree or not(0.411207055068)
- Total number of Spanning Trees in a Graph(0.356300429333)
- Overview of Data Structures | Set 3 (Graph, Trie, Segment Tree and Suffix Tree)(0.329894545665)
- Check whether given degrees of vertices represent a Graph or Tree(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 8(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 7(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 6(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 5(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 4(0.291069102382)
- Algorithms | Graph Minimum Spanning Tree | Question 3(0.291069102382)

Ugly Number(263)

- Ugly Numbers(1.0)
- Super Ugly Number (Number whose prime factors are in given set)(0.549988394922)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)

- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)

Ugly Number II(264)

- Ugly Numbers(0.709297266606)
- Super Ugly Number (Number whose prime factors are in given set)(0.390105265183)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)

Paint House II(265)

- Flood fill Algorithm – how to implement fill() in paint?(0.150640184987)
- Flipkart Interview | Set 7 (For SDE II)(0.136276341439)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.125366937987)

- Flipkart Interview Experience| Set 38 (For SDE II)(0.125366937987)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.125366937987)
- Amazon Interview experience | Set 326 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 348 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 313 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 312 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 163 (For SDE II)(0.125366937987)

Palindrome Permutation(266)

- Print all palindrome permutations of a string(0.579738671538)
- Permutation Coefficient(0.336096927276)
- Palindromic Primes(0.336096927276)
- Smallest Palindrome after replacement(0.260555671056)
- Permutation and Combination in Python(0.260555671056)
- Palindrome Substring Queries(0.260555671056)
- Lexicographically next permutation in C++(0.260555671056)
- Lexicographically first palindromic string(0.260555671056)
- K difference permutation(0.260555671056)
- How to find Lexicographically previous permutation?(0.260555671056)

Palindrome Permutation II(267)

- Print all palindrome permutations of a string(0.411207055068)
- Permutation Coefficient(0.260555671056)
- Palindromic Primes(0.260555671056)

- Smallest Palindrome after replacement(0.201993092498)
- Permutation and Combination in Python(0.201993092498)
- Palindrome Substring Queries(0.201993092498)
- Lexicographically next permutation in C++(0.201993092498)
- Lexicographically first palindromic string(0.201993092498)
- K difference permutation(0.201993092498)
- How to find Lexicographically previous permutation?(0.201993092498)

Missing Number(268)

- Find the Missing Number(1.0)
- Find the smallest missing number(0.709297266606)
- Find the missing number in a string of numbers with no separator(0.709052873586)
- Find the missing number in Geometric Progression(0.579738671538)
- Find the missing number in Arithmetic Progression(0.579738671538)
- Find missing number in another array which is shuffled copy(0.449436416524)
- Find Two Missing Numbers | Set 2 (XOR based solution)(0.410362644952)
- Find Two Missing Numbers | Set 1 (An Interesting Linear Time Solution)(0.410362644952)
- Find the smallest positive number missing from an unsorted array | Set 1(0.379978361591)
- Smallest number divisible by first n numbers(0.368023208756)

Alien Dictionary(269)

- Given a sorted dictionary of an alien language, find order of characters(0.410362644952)

- Get() method for dictionaries in Python(0.260555671056)
- Handling missing keys in Python dictionaries(0.194314340169)
- Generate a graph using Dictionary in Python(0.194314340169)
- Find all strings that match specific pattern in a dictionary(0.194314340169)
- Data Structure for Dictionary and Spell Checker?(0.194314340169)
- Output of python program | Set 14 (Dictionary)(0.175786078393)
- Output of Python programs | Set 9 (Dictionary)(0.175786078393)
- Find largest word in dictionary by deleting some characters of given string(0.161713780663)
- Building a terminal based online dictionary with Python and bash(0.161713780663)

Closest Binary Search Tree Value(270)

- Find the closest element in Binary Search Tree(0.669418851727)
- Find the node with minimum value in a Binary Search Tree(0.580332984677)
- Binary Tree to Binary Search Tree Conversion(0.572463774455)
- Find the closest leaf in a Binary Tree(0.51014901931)
- Binary Search(0.502328778226)
- Minimum swap required to convert binary tree to binary search tree(0.461313774437)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Binary Search Tree | Set 1 (Search and Insertion)(0.439338734046)
- Treap (A Randomized Binary Search Tree)(0.431613418971)
- Threaded Binary Search Tree | Deletion(0.431613418971)

Encode and Decode Strings(271)

- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Pairs of complete strings in two sets of strings(0.285306190981)
- Given two strings, find if first string is a subsequence of second(0.285306190981)
- String matching where one string contains wildcard characters(0.260555671056)
- Sort an array of strings according to string lengths(0.260555671056)
- Search in an array of strings where non-empty strings are sorted(0.260555671056)
- Remove characters from the first string which are present in the second string(0.260555671056)
- Huffman Decoding(0.260555671056)
- Check if given string can be split into four distinct strings(0.260555671056)
- Sort a string according to the order defined by another string(0.241299136472)

Closest Binary Search Tree Value II(272)

- Find the closest element in Binary Search Tree(0.580332984677)
- Binary Tree to Binary Search Tree Conversion(0.505164486208)
- Find the node with minimum value in a Binary Search Tree(0.503102612415)
- Find the closest leaf in a Binary Tree(0.450175502327)
- Binary Search(0.449436416524)
- Minimum swap required to convert binary tree to binary search tree(0.407081366967)
- Check whether a binary tree is a full binary tree or not(0.402484879511)
- Binary Search Tree | Set 1 (Search and Insertion)(0.38768972948)

- Treap (A Randomized Binary Search Tree)(0.380872608476)
- Threaded Binary Search Tree | Deletion(0.380872608476)

Integer to English Words(273)

- Median in a stream of integers (running integers)(0.285306190981)
- Word formation using concatenation of two dictionary words(0.260555671056)
- Count of m digit integers that are divisible by an integer n(0.241299136472)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Square root of an integer(0.201993092498)
- Sorting Big Integers(0.201993092498)
- Placements | English | Fill in the Blanks(0.201993092498)

H-Index(274)

H-Index II(275)

- Flipkart Interview | Set 7 (For SDE II)(0.175786078393)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.161713780663)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.161713780663)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.161713780663)
- Amazon Interview experience | Set 326 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 348 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 313 (For SDE II)(0.161713780663)

- Amazon Interview Experience | Set 312 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 163 (For SDE II)(0.161713780663)
- GATE | GATE 2017 MOCK II | Question 9(0.141430567926)

Paint Fence(276)

- Flood fill Algorithm – how to implement fill() in paint?(0.194314340169)
- Rail Fence Cipher – Encryption and Decryption(0.175786078393)
- Peterson's Algorithm for Mutual Exclusion | Set 2 (CPU Cycles and Memory Fence)(0.133785092946)

Find the Celebrity(277)

- The Celebrity Problem(0.579738671538)

First Bad Version(278)

- Compare two Version numbers(0.260555671056)
- ES2015: Latest version of JavaScript(0.220288150562)
- Design and Implement Special Stack Data Structure | Added Space Optimized Version(0.133785092946)
- Pattern Searching | Set 7 (Boyer Moore Algorithm – Bad Character Heuristic)(0.12725898701)

Perfect Squares(279)

- Check perfect square using addition/subtraction(0.502328778226)
- Number of perfect squares between two given numbers(0.410362644952)
- Find minimum number to be divided to make a number a perfect square(0.355476777955)
- Perfect Number(0.336096927276)

- Magic Square(0.336096927276)
- Latin Square(0.336096927276)
- Square root of an integer(0.260555671056)
- Perfect reversible string(0.260555671056)
- Perfect cubes in a range(0.260555671056)
- Direction at last square block(0.260555671056)

Wiggle Sort(280)

- Sort a nearly sorted (or K sorted) array(0.450175502327)
- Tag Sort (To get both sorted and original)(0.411207055068)
- Sort an array when two halves are sorted(0.411207055068)
- Odd-Even Sort / Brick Sort(0.411207055068)
- Sorting Strings using Bubble Sort(0.368023208756)
- Bead Sort | A Natural Sorting Algorithm(0.368023208756)
- Tree Sort(0.336096927276)
- Stooge Sort(0.336096927276)
- Sorting Terminology(0.336096927276)
- Sort an almost sorted array where only two elements are swapped(0.336096927276)

Zigzag Iterator(281)

- Iterators in Python(0.336096927276)
- Iterators in Java(0.336096927276)
- Iterator Pattern(0.336096927276)
- Iterative Deepening Search(IDS) or Iterative Deepening Depth First Search(IDDFS)(0.274611786436)

- Longest Zig-Zag Subsequence(0.260555671056)
- Iterators in C++ STL(0.260555671056)
- Iterative Tower of Hanoi(0.260555671056)
- Iterative Quick Sort(0.260555671056)
- Iterative Preorder Traversal(0.260555671056)
- Iterative Merge Sort(0.260555671056)

Expression Add Operators(282)

- Add two numbers without using arithmetic operators(0.356300429333)
- Increment (Decrement) operators require L-value Expression(0.318784021754)
- Constant time range add operation on an array(0.318784021754)
- SQL | BETWEEN & IN Operator(0.260555671056)
- SQL | AND and OR operators(0.260555671056)
- Operators in Java(0.260555671056)
- Operators in C | Set 1 (Arithmetic Operators)(0.260555671056)
- Operating Systems | Segmentation(0.260555671056)
- Operating System | Thread(0.260555671056)
- Operating System | Paging(0.260555671056)

Move Zeroes(283)

- Find the number of zeroes(0.579738671538)
- Move all zeroes to end of array(0.449436416524)
- Find all triplets with zero sum(0.449436416524)

- Two elements whose sum is closest to zero(0.379978361591)
- Count Pairs Of Consecutive Zeros(0.379978361591)
- Total coverage of all zeros in a binary matrix(0.335175743328)
- Remove Trailing Zeros From string in C++(0.335175743328)
- Remove Trailing Zeros From String in Java(0.335175743328)
- Find if there is a triplet in a Balanced BST that adds to zero(0.335175743328)
- Count trailing zeroes in factorial of a number(0.335175743328)

Peeking Iterator(284)

- Iterators in Python(0.336096927276)
- Iterators in Java(0.336096927276)
- Iterator Pattern(0.336096927276)
- Iterative Deepening Search(IDS) or Iterative Deepening Depth First Search(IDDFS)(0.274611786436)
- Iterators in C++ STL(0.260555671056)
- Iterative Tower of Hanoi(0.260555671056)
- Iterative Quick Sort(0.260555671056)
- Iterative Preorder Traversal(0.260555671056)
- Iterative Merge Sort(0.260555671056)
- How to use Iterator in Java?(0.260555671056)

Inorder Successor in BST(285)

- Inorder predecessor and successor for a given key in BST(0.579738671538)
- Populate Inorder Successor for all nodes(0.411207055068)

- Inorder Successor in Binary Search Tree(0.356300429333)
- Two nodes of a BST are swapped, correct the BST(0.285306190981)
- Convert a normal BST to Balanced BST(0.285306190981)
- K'th Largest Element in BST when modification to BST is not allowed(0.241299136472)
- Find k-th smallest element in BST (Order Statistics in BST)(0.241299136472)
- Floor and Ceil from a BST(0.201993092498)
- Sorted Array to Balanced BST(0.17077611319)
- Second largest element in BST(0.17077611319)

Walls and Gates(286)

- GATE | Gate IT 2008 | Question 9(0.368023208756)
- GATE | Gate IT 2008 | Question 82(0.368023208756)
- GATE | Gate IT 2008 | Question 81(0.368023208756)
- GATE | Gate IT 2008 | Question 80(0.368023208756)
- GATE | Gate IT 2008 | Question 8(0.368023208756)
- GATE | Gate IT 2008 | Question 79(0.368023208756)
- GATE | Gate IT 2008 | Question 78(0.368023208756)
- GATE | Gate IT 2008 | Question 77(0.368023208756)
- GATE | Gate IT 2008 | Question 76(0.368023208756)
- GATE | Gate IT 2008 | Question 75(0.368023208756)

Find the Duplicate Number(287)

- Smallest number divisible by first n numbers(0.368023208756)

- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)
- Find count of digits in a number that divide the number(0.368023208756)

Unique Word Abbreviation(288)

- C++ program to print unique words in a file(0.318784021754)
- Word formation using concatenation of two dictionary words(0.260555671056)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Find shortest unique prefix for every word in a given list | Set 2 (Using Sorting)(0.225554872207)
- Find shortest unique prefix for every word in a given list | Set 1 (Using Trie)(0.225554872207)
- Testimonials – Words that keep us going(0.201993092498)
- SQL | UNIQUE Constraint(0.201993092498)
- Length Of Last Word in a String(0.201993092498)

Game of Life(289)

- Program for Conway's Game Of Life(0.579738671538)
- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)

Word Pattern(290)

- Print all words matching a pattern in CamelCase Notation Dictonary(0.410362644952)
- Word formation using concatenation of two dictionary words(0.336096927276)
- Iterator Pattern(0.336096927276)
- Command Pattern(0.336096927276)
- Adapter Pattern(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.291219418564)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Searching for Patterns | Set 1 (Naive Pattern Searching)(0.274611786436)
- Wildcard Pattern Matching(0.260555671056)

Word Pattern II(291)

- Print all words matching a pattern in CamelCase Notation Dictonary(0.291069102382)
- Word formation using concatenation of two dictionary words(0.260555671056)
- Iterator Pattern(0.260555671056)
- Command Pattern(0.260555671056)
- Adapter Pattern(0.260555671056)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Searching for Patterns | Set 1 (Naive Pattern Searching)(0.212889950749)
- Wildcard Pattern Matching(0.201993092498)

Nim Game(292)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.590594008858)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)
- Program for Conway's Game Of Life(0.220288150562)

Flip Game(293)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- Flip Binary Tree(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)

Flip Game II(294)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.241299136472)
- Implementation of Tic-Tac-Toe game(0.201993092498)
- Implementation of Minesweeper Game(0.201993092498)
- Hangman Game in Python(0.201993092498)
- Flip Binary Tree(0.201993092498)
- A Number Link Game(0.201993092498)
- The prisoner's dilemma in Game theory(0.17077611319)
- Puzzle 73 | The Card Game(0.17077611319)
- Puzzle 69 |The Number Game(0.17077611319)
- Project Idea | (A Game of Anagrams)(0.17077611319)

Find Median from Data Stream(295)

- Data Mining(0.449436416524)
- Data Abstraction and Data Independence(0.318784021754)
- Median in a stream of integers (running integers)(0.291069102382)
- Stream In Java(0.260555671056)
- Data Warehousing(0.260555671056)
- Character Stream Vs Byte Stream in Java(0.260555671056)
- Placements | Data Interpretation(0.201993092498)
- Persistent data structures(0.201993092498)
- Overview of Data Structures | Set 1 (Linear Data Structures)(0.201993092498)
- Median of two sorted arrays(0.201993092498)

Best Meeting Point(296)

- Number of Integral Points between Two Points(0.318784021754)
- Maximum points collected by two persons allowed to meet once(0.291069102382)
- Meet in the middle(0.260555671056)
- Prime points (Points that split a number into two primes)(0.225764846003)
- Triangle with no point inside(0.201993092498)
- Saddle point in a matrix(0.201993092498)
- Find number of endless points(0.201993092498)
- Find a partition point in array(0.201993092498)
- Circle and Lattice Points(0.201993092498)
- [TopTalent.in] How Flipkart gets the best out of their applicants(0.17077611319)

Serialize and Deserialize Binary Tree(297)

- Serialize and Deserialize a Binary Tree(1.0)
- Serialize and Deserialize an N-ary Tree(0.602974816038)
- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)
- Binary Tree | Set 3 (Types of Binary Tree)(0.439404118785)
- fork() and Binary Tree(0.411207055068)
- Threaded Binary Tree(0.411207055068)
- Foldable Binary Trees(0.411207055068)
- Flip Binary Tree(0.411207055068)
- Enumeration of Binary Trees(0.411207055068)

Binary Tree Longest Consecutive Sequence(298)

- Longest consecutive sequence in Binary tree(1.0)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Binary Tree to Binary Search Tree Conversion(0.410995463935)
- Length of the Longest Consecutive 1s in Binary Representation(0.380872608476)
- If you are given two traversal sequences, can you construct the binary tree?(0.380872608476)
- Find longest sequence of 1's in binary representation with one flip(0.380872608476)
- Binary Tree | Set 3 (Types of Binary Tree)(0.380732466149)
- fork() and Binary Tree(0.356300429333)
- Threaded Binary Tree(0.356300429333)
- Longest Consecutive Subsequence(0.356300429333)

Bulls and Cows(299)

Longest Increasing Subsequence(300)

- Longest Common Increasing Subsequence (LCS + LIS)(0.579738671538)
- Dynamic Programming | Set 3 (Longest Increasing Subsequence)(0.524591090446)
- Construction of Longest Increasing Subsequence using Dynamic Programming(0.524591090446)
- Longest alternating subsequence(0.503102612415)
- Longest Zig-Zag Subsequence(0.503102612415)
- Longest Repeating Subsequence(0.503102612415)
- Longest Consecutive Subsequence(0.503102612415)
- Count all increasing subsequences(0.503102612415)
- Longest Increasing Subsequence Size (N log N)(0.449436416524)
- Construction of Longest Increasing Subsequence (N log N)(0.449436416524)

Remove Invalid Parentheses(301)

- Remove Invalid Parentheses(1.0)
- Removing punctuations from a given string(0.17077611319)
- Remove spaces from a given string(0.17077611319)
- Remove extra spaces from a string(0.17077611319)
- Remove duplicates from sorted array(0.17077611319)
- Remove all duplicates from a given string(0.17077611319)
- Recursively remove all adjacent duplicates(0.17077611319)
- Program to remove vowels from a String(0.17077611319)
- Print all combinations of balanced parentheses(0.17077611319)

- Length of Longest sub-string that can be removed(0.17077611319)

Smallest Rectangle Enclosing Black Pixels(302)

- Find the smallest and second smallest elements in an array(0.212772510465)
- Find if two rectangles overlap(0.194314340169)
- Maximum sum of smallest and second smallest in an array(0.194314340169)
- Smallest Palindrome after replacement(0.150640184987)
- Find the smallest missing number(0.150640184987)
- Count number of squares in a rectangle(0.127359529795)
- Check if four segments form a rectangle(0.127359529795)
- Smallest of three integers without comparison operators(0.127359529795)
- Smallest Subarray with given GCD(0.127359529795)
- Smallest Difference Triplet from Three arrays(0.127359529795)

Range Sum Query - Immutable(303)

- Queries on the sum of prime factor counts in a range(0.450175502327)
- Submatrix Sum Queries(0.411207055068)
- Range LCM Queries(0.411207055068)
- Binary Indexed Tree : Range Update and Range Queries(0.342390186113)
- Min-Max Range Queries in Array(0.336096927276)
- Find sum of sum of all sub-sequences(0.310890774681)
- Subset sum queries using bitset(0.291219418564)
- Range Queries for Frequencies of array elements(0.291219418564)
- Segment Tree | Set 2 (Range Minimum Query)(0.237903094633)

- Segment Tree | Set 1 (Sum of given range)(0.237903094633)

Range Sum Query 2D - Immutable(304)

- Queries on the sum of prime factor counts in a range(0.380872608476)
- Submatrix Sum Queries(0.356300429333)
- Range LCM Queries(0.356300429333)
- Binary Indexed Tree : Range Update and Range Queries(0.296672366897)
- Min-Max Range Queries in Array(0.291219418564)
- Find sum of sum of all sub-sequences(0.27423415918)
- Subset sum queries using bitset(0.252334201434)
- Range Queries for Frequencies of array elements(0.252334201434)
- Segment Tree | Set 2 (Range Minimum Query)(0.206136966068)
- Segment Tree | Set 1 (Sum of given range)(0.206136966068)

Number of Islands II(305)

- Count number of islands where every island is row-wise and column-wise separated(0.390105265183)
- Find the number of islands | Set 1 (Using DFS)(0.318784021754)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)

- Finding number of digits in n'th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)

Additive Number(306)

- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)
- Find count of digits in a number that divide the number(0.368023208756)

Range Sum Query - Mutable(307)

- Queries on the sum of prime factor counts in a range(0.450175502327)
- Submatrix Sum Queries(0.411207055068)
- Range LCM Queries(0.411207055068)
- Binary Indexed Tree : Range Update and Range Queries(0.342390186113)
- Min-Max Range Queries in Array(0.336096927276)
- Find sum of sum of all sub-sequences(0.310890774681)
- Subset sum queries using bitset(0.291219418564)

- Range Queries for Frequencies of array elements(0.291219418564)
- Segment Tree | Set 2 (Range Minimum Query)(0.237903094633)
- Segment Tree | Set 1 (Sum of given range)(0.237903094633)

Range Sum Query 2D - Mutable(308)

- Queries on the sum of prime factor counts in a range(0.380872608476)
- Submatrix Sum Queries(0.356300429333)
- Range LCM Queries(0.356300429333)
- Binary Indexed Tree : Range Update and Range Queries(0.296672366897)
- Min-Max Range Queries in Array(0.291219418564)
- Find sum of sum of all sub-sequences(0.27423415918)
- Subset sum queries using bitset(0.252334201434)
- Range Queries for Frequencies of array elements(0.252334201434)
- Segment Tree | Set 2 (Range Minimum Query)(0.206136966068)
- Segment Tree | Set 1 (Sum of given range)(0.206136966068)

Best Time to Buy and Sell Stock with Cooldown(309)

- Stock Buy Sell to Maximize Profit(0.380872608476)
- Maximum profit by buying and selling a share at most k times(0.304125741875)
- Maximum profit by buying and selling a share at most twice(0.201993092498)
- Changing One Clock Time to Other Time in Minimum Number of Operations(0.16279449512)
- Time Functions in Python | Set 1 (time(), ctime(), sleep()…)(0.152314155194)
- The Stock Span Problem(0.136276341439)

- An interesting time complexity question(0.136276341439)
- A Time Complexity Question(0.136276341439)
- [TopTalent.in] How Flipkart gets the best out of their applicants(0.115215543378)
- What to do at the time of Wrong Answer (WA)?(0.115215543378)

Minimum Height Trees(310)

- Roots of a tree which give minimum height(0.776514530475)
- Minimum Product Spanning Tree(0.411207055068)
- Find maximum (or minimum) in Binary Tree(0.411207055068)
- Find Minimum Depth of a Binary Tree(0.411207055068)
- Check if a given Binary Tree is height balanced like a Red-Black Tree(0.366529477546)
- Iterative Method to find Height of Binary Tree(0.356300429333)
- Boruvka's algorithm for Minimum Spanning Tree(0.356300429333)
- Applications of Minimum Spanning Tree Problem(0.356300429333)
- Minimum swap required to convert binary tree to binary search tree(0.329894545665)
- Write a Program to Find the Maximum Depth or Height of a Tree(0.318784021754)

Sparse Matrix Multiplication(311)

- Printing brackets in Matrix Chain Multiplication Problem(0.318784021754)
- Dynamic Programming | Set 8 (Matrix Chain Multiplication)(0.291069102382)
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)(0.291069102382)
- Sparse Set(0.260555671056)

- Queries in a Matrix(0.260555671056)
- N-th multiple in sorted list of multiples of two numbers(0.260555671056)
- Multiplicative order(0.260555671056)
- Matrix Introduction(0.260555671056)
- Matrix Exponentiation(0.260555671056)
- Find Next Sparse Number(0.260555671056)

Burst Balloons(312)

Super Ugly Number(313)

- Ugly Numbers(0.709297266606)
- Super Ugly Number (Number whose prime factors are in given set)(0.619400010025)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)
- Number of perfect squares between two given numbers(0.285306190981)
- Next higher number with same number of set bits(0.285306190981)
- How to check if a given number is Fibonacci number?(0.285306190981)
- Finding number of digits in n’th Fibonacci number(0.285306190981)
- Find the missing number in a string of numbers with no separator(0.285306190981)

Binary Tree Vertical Order Traversal(314)

- Print a Binary Tree in Vertical Order | Set 3 (Using Level Order Traversal)(0.634633579703)

- Given level order traversal of a Binary Tree, check if the Tree is a Min-Heap(0.529878722844)
- Print a Binary Tree in Vertical Order | Set 1(0.519387993313)
- Perfect Binary Tree Specific Level Order Traversal(0.519387993313)
- Level Order Tree Traversal(0.51014901931)
- Diagonal Traversal of Binary Tree(0.51014901931)
- Density of Binary Tree in One Traversal(0.51014901931)
- Boundary Traversal of binary tree(0.51014901931)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Perfect Binary Tree Specific Level Order Traversal | Set 2(0.439274990316)

Count of Smaller Numbers After Self(315)

- Count of Binary Digit numbers smaller than N(0.450175502327)
- Count numbers with same first and last digits(0.411207055068)
- Number of subtrees having odd count of even numbers(0.411065370983)
- Find count of digits in a number that divide the number(0.411065370983)
- Count numbers which can be constructed using two numbers(0.411065370983)
- Count smaller numbers whose XOR with n produces greater value(0.374807770059)
- Count number of ways to divide a number in 4 parts(0.372055731454)
- Count natural numbers whose all permutation are greater than that number(0.372055731454)
- Count minimum number of subsets (or subsequences) with consecutive numbers(0.342390186113)
- Closest (or Next) smaller and greater numbers with same number of set bits(0.342390186113)

Remove Duplicate Letters(316)

- Remove duplicates from sorted array(0.411207055068)
- Remove all duplicates from a given string(0.411207055068)
- Recursively remove all adjacent duplicates(0.411207055068)
- Remove duplicates from an unsorted linked list(0.356300429333)
- Remove duplicates from an array of small primes(0.356300429333)
- Remove duplicates from a sorted linked list(0.356300429333)
- Remove all occurrences of duplicates from a sorted Linked List(0.318784021754)
- Remove Invalid Parentheses(0.201993092498)
- Find duplicates under given constraints(0.201993092498)
- AVL with duplicate keys(0.201993092498)

Shortest Distance from All Buildings(317)

- Find Shortest distance from a guard in a Bank(0.411207055068)
- Shortest Uncommon Subsequence(0.201993092498)
- Shortest Superstring Problem(0.201993092498)
- Shortest Common Supersequence(0.201993092498)
- Hamming Distance between two strings(0.201993092498)
- Find the minimum distance between two numbers(0.201993092498)
- Time Complexity of building a heap(0.17077611319)
- Shortest path in a Binary Maze(0.17077611319)
- Printing Shortest Common Supersequence(0.17077611319)
- Number of buildings facing the sun(0.17077611319)

Maximum Product of Word Lengths(318)

- Find the Increasing subsequence of length three with maximum product(0.51014901931)
- Maximum Product Subarray(0.411207055068)
- Length Of Last Word in a String(0.411207055068)
- Maximum and Minimum Product Subsets(0.336096927276)
- Find maximum length Snake sequence(0.336096927276)
- Breaking an Integer to get Maximum Product(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.318849541433)
- Substring with highest frequency length product(0.291219418564)
- Print Maximum Length Chain of Pairs(0.291219418564)
- Maximum product of two non-intersecting paths in a tree(0.291219418564)

Bulb Switcher(319)

- Puzzle 7 | (3 Bulbs and 3 Switches)(0.150556969602)

Generalized Abbreviation(320)

- Generics in Java(0.336096927276)
- Generators in Python(0.336096927276)
- Generating Test Cases (generate() and generate_n() in C++)(0.336096927276)
- Test Case Generation | Set 5 (Generating random Sorted Arrays and Palindromes)(0.260555671056)
- Program for Sudoku Generator(0.260555671056)
- Generate Pythagorean Triplets(0.260555671056)
- Mid-Point Line Generation Algorithm(0.220288150562)

- Heap's Algorithm for generating permutations(0.220288150562)
- Generic Linked List in C(0.220288150562)
- Generating random numbers in Java(0.220288150562)

Create Maximum Number(321)

- Maximum number of threads that can be created within a process in C(0.579738671538)
- Find the maximum number of handshakes(0.503102612415)
- Number with maximum number of prime factors(0.502929265114)
- Maximum sum of distinct numbers such that LCM of these numbers is N(0.418906716157)
- Level with maximum number of nodes(0.411207055068)
- Find the row with maximum number of 1s(0.411207055068)
- Querying maximum number of divisors that a number in a given range has(0.390105265183)
- Maximum number of Zombie process a system can handle(0.356300429333)
- Recursively break a number in 3 parts to get maximum sum(0.318784021754)
- Maximum sum of distinct numbers with LCM as N(0.318784021754)

Coin Change(322)

- Dynamic Programming | Set 7 (Coin Change)(0.449436416524)
- Make a fair coin from a biased coin(0.368023208756)
- Minimum cost for acquiring all coins with k extra coins allowed with every coin(0.364020643353)
- Frobenius coin problem(0.260555671056)
- Decision Trees – Fake (Counterfeit) Coin Puzzle (12 Coin Puzzle)(0.237903094633)

- Puzzle 53 | The Counterfeit Coin(0.220288150562)
- How to change the output of printf() in main() ?(0.220288150562)
- Changing Class Members in Python(0.220288150562)
- Change if all bits can be made same by single flip(0.220288150562)
- Change gender of a given string(0.220288150562)

Number of Connected Components in an Undirected Graph(323)

- Connected Components in an undirected graph(0.818180207367)
- Number of Triangles in an Undirected Graph(0.51014901931)
- Number of Triangles in Directed and Undirected Graphs(0.431613418971)
- Count number of edges in an undirected graph(0.431613418971)
- Strongly Connected Components(0.356300429333)
- Find k-cores of an undirected graph(0.356300429333)
- Clone an Undirected Graph(0.356300429333)
- Number of sink nodes in a graph(0.291219418564)
- Detect cycle in an undirected graph(0.291219418564)
- Total number of Spanning Trees in a Graph(0.252334201434)

Wiggle Sort II(324)

- Sort a nearly sorted (or K sorted) array(0.348993907955)
- Tag Sort (To get both sorted and original)(0.318784021754)
- Sort an array when two halves are sorted(0.318784021754)
- Odd-Even Sort / Brick Sort(0.318784021754)

- Sorting Strings using Bubble Sort(0.285306190981)
- Bead Sort | A Natural Sorting Algorithm(0.285306190981)
- Tree Sort(0.260555671056)
- Stooge Sort(0.260555671056)
- Sorting Terminology(0.260555671056)
- Sort an almost sorted array where only two elements are swapped(0.260555671056)

Maximum Size Subarray Sum Equals k(325)

- Maximum subarray size, such that all subarrays of that size have sum less than k(0.79913334114)
- Find maximum (or minimum) sum of a subarray of size k(0.716811741443)
- Sum of minimum and maximum elements of all subarrays of size k.(0.632790458368)
- Sliding Window Maximum (Maximum of all subarrays of size k)(0.535855954873)
- Maximum sum two non-overlapping subarrays of given size(0.503102612415)
- Find Maximum XOR value of a sub-array of size k(0.503102612415)
- Maximum circular subarray sum(0.450175502327)
- Sum of all Subarrays(0.449436416524)
- Find maximum sum possible equal sum of three stacks(0.424429533893)
- Split an array into two equal Sum subarrays(0.380872608476)

Power of Three(326)

- Find power of power under mod of a prime(0.709297266606)
- Program to find whether a no is power of two(0.579738671538)

- Powerful Number(0.579738671538)
- Power Set(0.579738671538)
- Time Complexity of Loop with Powers(0.379978361591)
- Print all prime factors and their powers(0.379978361591)
- Find whether a given number is a power of 4 or not(0.379978361591)
- Write you own Power without using multiplication(*) and division(/) operators(0.30321606445)
- Smallest power of 2 greater than or equal to n(0.30321606445)
- Highest power of 2 less than or equal to given number(0.30321606445)

Count of Range Sum(327)

- Queries on the sum of prime factor counts in a range(0.579738671538)
- Count pairs with given sum(0.411207055068)
- Find sum of sum of all sub-sequences(0.36771998047)
- Count factorial numbers in a given range(0.356300429333)
- Count all sub-arrays having sum divisible by k(0.356300429333)
- Count of n digit numbers whose sum of digits equals to given sum(0.329894545665)
- Count triplets with sum smaller than a given value(0.318784021754)
- Count total divisors of A or B in a given range(0.318784021754)
- Count pairs with sum as a prime number and less than n(0.318784021754)
- Count pairs in a sorted array whose sum is less than x(0.318784021754)

Odd Even Linked List(328)

- Segregate even and odd nodes in a Linked List(0.656972921033)

- Check if a linked list is Circular Linked List(0.580332984677)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Rearrange a linked list such that all even and odd positioned nodes are together(0.524591090446)
- Rotate a Linked List(0.503102612415)
- Merge a linked list into another linked list at alternate positions(0.503102612415)
- Identical Linked Lists(0.503102612415)
- Flattening a Linked List(0.503102612415)
- Can we reverse a linked list in less than O(n)?(0.503102612415)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.450268144656)

Longest Increasing Path in a Matrix(329)

- Find the longest path in a matrix with given constraints(0.51014901931)
- Find whether there is path between two cells in matrix(0.411207055068)
- Number of palindromic paths in a matrix(0.336096927276)
- Longest path in an undirected tree(0.336096927276)
- Sort a Matrix in all way increasing order(0.291219418564)
- Longest path between any pair of vertices(0.291219418564)
- Longest Possible Route in a Matrix with Hurdles(0.291219418564)
- Longest Path in a Directed Acyclic Graph(0.291219418564)
- Maximum decimal value path in a binary matrix(0.260555671056)
- Longest Common Increasing Subsequence (LCS + LIS)(0.260555671056)

Patching Array(330)

- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Find Duplicates of array using bit array(0.368023208756)
- Pointer to an Array | Array Pointer(0.336096927276)
- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)
- Arrays in Java(0.336096927276)
- Arrays in Java(0.336096927276)
- kasai's Algorithm for Construction of LCP array from Suffix Array(0.311257467527)

Verify Preorder Serialization of a Binary Tree(331)

- Serialize and Deserialize a Binary Tree(0.51014901931)
- Check whether a binary tree is a full binary tree or not(0.449851703924)
- Calculate depth of a full Binary tree from Preorder(0.431613418971)
- Binary Tree to Binary Search Tree Conversion(0.410995463935)
- Leaf nodes from Preorder of a Binary Search Tree(0.380872608476)
- Binary Tree | Set 3 (Types of Binary Tree)(0.380732466149)
- fork() and Binary Tree(0.356300429333)
- Threaded Binary Tree(0.356300429333)
- Foldable Binary Trees(0.356300429333)
- Flip Binary Tree(0.356300429333)

Reconstruct Itinerary(332)

- Reconstructing Segment Tree(0.260555671056)
- Find Itinerary from a given list of tickets(0.220288150562)

Largest BST Subtree(333)

- Find the largest BST subtree in a given Binary Tree | Set 1(0.48267966065)
- Find largest subtree having identical left and right subtrees(0.455201845765)
- K'th Largest Element in BST when modification to BST is not allowed(0.418906716157)
- Second largest element in BST(0.411207055068)
- Largest BST in a Binary Tree | Set 2(0.318784021754)
- Count BST subtrees that lie in given range(0.318784021754)
- Two nodes of a BST are swapped, correct the BST(0.285306190981)
- Convert a normal BST to Balanced BST(0.285306190981)
- Find k-th smallest element in BST (Order Statistics in BST)(0.241299136472)
- Largest subarray with GCD one(0.201993092498)

Increasing Triplet Subsequence(334)

- Count all increasing subsequences(0.503102612415)
- Printing Maximum Sum Increasing Subsequence(0.356300429333)
- Find the Increasing subsequence of length three with maximum product(0.356300429333)
- Maximum product of an increasing subsequence of size 3(0.318784021754)
- Longest Common Increasing Subsequence (LCS + LIS)(0.318784021754)
- Minimum number of elements which are not part of Increasing or decreasing subsequence in array(0.291069102382)

- Dynamic Programming | Set 3 (Longest Increasing Subsequence)(0.291069102382)
- Construction of Longest Increasing Subsequence using Dynamic Programming(0.291069102382)
- Dynamic Programming | Set 14 (Maximum Sum Increasing Subsequence)(0.269517613246)
- Longest Increasing Subsequence Size (N log N)(0.252138706945)

Self Crossing(335)

- Can a C++ class have an object of self type?(0.194314340169)
- Self Organizing List | Set 1 (Introduction)(0.175786078393)
- Minimum Initial Energy Required To Cross Street(0.175786078393)
- Self assignment check in assignment operator(0.161713780663)
- SQL | Join (Cartesian Join & Self Join)(0.121603314786)

Palindrome Pairs(336)

- Palindrome pair in an array of words (or strings)(0.502328778226)
- Given an array of pairs, find all symmetric pairs in it(0.368023208756)
- Palindromic Primes(0.336096927276)
- Find pairs with given sum such that elements of pair are in different rows(0.311257467527)
- Pair with given product | Set 1 (Find if any pair exists)(0.291219418564)
- Find pairs with given sum such that pair elements lie in different BSTs(0.291219418564)
- Smallest Palindrome after replacement(0.260555671056)
- Palindrome Substring Queries(0.260555671056)

- Pairs of Amicable Numbers(0.260555671056)
- Pair Class in Java(0.260555671056)

House Robber III(337)

- Encrypt a string into the Rovarspraket (The Robber Language)(0.150640184987)
- Encrypt a string into the Rovarspraket (The Robber Language)(0.150640184987)

Counting Bits(338)

- Count set bits in an integer(0.579738671538)
- Check if bits of a number has count of consecutive set bits in increasing order(0.488890890265)
- Sort an array according to count of set bits(0.449436416524)
- Counting Triangles in a Rectangular space using BIT(0.449436416524)
- Count number of bits to be flipped to convert A to B(0.449436416524)
- Count all pairs of an array which differ in K bits(0.449436416524)
- Program to count number of set bits in an (big) array(0.410362644952)
- How to count set bits in a floating point number in C?(0.410362644952)
- Count trailing zero bits using lookup table(0.410362644952)
- Count total set bits in all numbers from 1 to n(0.410362644952)

Nested List Weight Sum(339)

- Find sum of sum of all sub-sequences(0.310890774681)
- Find pairs with given sum in doubly linked list(0.260555671056)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.257680072134)

- Find a triplet from three linked lists with sum equal to a given number(0.237903094633)
- Sum of all Subarrays(0.220288150562)
- Find maximum sum possible equal sum of three stacks(0.220288150562)
- Recursively print all sentences that can be formed from list of word lists(0.204007612999)
- Print all possible sums of consecutive numbers with sum N(0.204007612999)
- Perfect Sum Problem (Print all subsets with given sum)(0.204007612999)
- Shortest Path in a weighted Graph where weight of an edge is 1 or 2(0.19087406613)

Longest Substring with At Most K Distinct Characters(340)

- Find the longest substring with k unique characters in a given string(0.519387993313)
- Count number of substrings with exactly k distinct characters(0.519387993313)
- Length of the longest substring without repeating characters(0.431613418971)
- Longest Non-palindromic substring(0.356300429333)
- Count substrings with same first and last characters(0.356300429333)
- Longest repeating and non-overlapping substring(0.291219418564)
- Length of the longest valid substring(0.291219418564)
- Length of Longest sub-string that can be removed(0.291219418564)
- Longest Common Prefix | Set 2 (Character by Character Matching)(0.276274998459)
- Searching characters and substring in a String in Java(0.252334201434)

Flatten Nested List Iterator(341)

- Flattening a Linked List(0.411207055068)
- Flatten a multilevel linked list(0.336096927276)
- Find Length of a Linked List (Iterative and Recursive)(0.291219418564)
- Search an element in a Linked List (Iterative and Recursive)(0.260555671056)
- Implementing Iterator pattern of a single Linked List(0.260555671056)
- Python | Set 3 (Strings, Lists, Tuples, Iterations)(0.237903094633)
- Flatten a multi-level linked list | Set 2 (Depth wise)(0.220288150562)
- Iterators in Python(0.220288150562)
- Iterators in Java(0.220288150562)
- Iterator Pattern(0.220288150562)

Power of Four(342)

- Find power of power under mod of a prime(0.709297266606)
- Program to find whether a no is power of two(0.579738671538)
- Powerful Number(0.579738671538)
- Power Set(0.579738671538)
- Time Complexity of Loop with Powers(0.379978361591)
- Print all prime factors and their powers(0.379978361591)
- Find whether a given number is a power of 4 or not(0.379978361591)
- Write you own Power without using multiplication(*) and division(/) operators(0.30321606445)
- Smallest power of 2 greater than or equal to n(0.30321606445)
- Highest power of 2 less than or equal to given number(0.30321606445)

Integer Break(343)

- Breaking an Integer to get Maximum Product(0.579738671538)
- Median in a stream of integers (running integers)(0.368023208756)
- Count of m digit integers that are divisible by an integer n(0.311257467527)
- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)
- Smallest of three integers without comparison operators(0.220288150562)
- Printing Integer between Strings in Java(0.220288150562)
- Multiply a given Integer with 3.5(0.220288150562)

Reverse String(344)

- Perfect reversible string(0.709297266606)
- Reverse words in a given string(0.579738671538)
- Write a program to reverse an array or string(0.502328778226)
- Reverse a string preserving space positions(0.502328778226)
- Print reverse of a string using recursion(0.502328778226)
- Different methods to reverse a string in C/C++(0.502328778226)
- Reverse string without using any temporary variable(0.449436416524)
- Reverse a string in Java (5 Different Ways)(0.449436416524)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)

Reverse Vowels of a String(345)

- Perfect reversible string(0.503102612415)
- Reverse words in a given string(0.411207055068)
- Program to remove vowels from a String(0.411207055068)
- Alternate vowel and consonant string(0.411207055068)
- Write a program to reverse an array or string(0.356300429333)
- Reverse a string preserving space positions(0.356300429333)
- Print reverse of a string using recursion(0.356300429333)
- Different methods to reverse a string in C/C++(0.356300429333)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Reverse string without using any temporary variable(0.318784021754)

Moving Average from Data Stream(346)

- Average of a stream of numbers(0.503102612415)
- Data Mining(0.449436416524)
- Data Abstraction and Data Independence(0.318784021754)
- Stream In Java(0.260555671056)
- Find the subarray with least average(0.260555671056)
- Data Warehousing(0.260555671056)
- Character Stream Vs Byte Stream in Java(0.260555671056)
- Sum of average of all subsets(0.201993092498)
- Placements | Data Interpretation(0.201993092498)
- Persistent data structures(0.201993092498)

Top K Frequent Elements(347)

- Maximum value K such that array has at-least K elements that are >= K(0.449988656407)
- Find the k most frequent words from a file(0.411207055068)
- Sum of k smallest elements in BST(0.356300429333)
- First element occurring k times in an array(0.356300429333)
- Find k closest elements to a given value(0.356300429333)
- Rotate each ring of matrix anticlockwise by K elements(0.318784021754)
- Place k elements such that minimum distance is maximized(0.318784021754)
- Find smallest range containing elements from k lists(0.318784021754)
- Count of subarrays whose maximum element is greater than k(0.318784021754)
- Sum of minimum and maximum elements of all subarrays of size k.(0.291069102382)

Design Tic-Tac-Toe(348)

- Implementation of Tic-Tac-Toe game(0.260555671056)
- Flyweight Design Pattern(0.260555671056)
- Singleton Design Pattern | Introduction(0.220288150562)
- Singleton Design Pattern | Implementation(0.220288150562)
- Designing Use Cases for a Project(0.220288150562)
- Compiler Design | Why FIRST and FOLLOW?(0.220288150562)
- Compiler Design | Runtime Environments(0.220288150562)
- Compiler Design | Lexical Analysis(0.220288150562)
- Compiler Design | Ambiguous Grammar(0.220288150562)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)

Intersection of Two Arrays(349)

- Union and Intersection of two sorted arrays(0.579738671538)
- Find Union and Intersection of two unsorted arrays(0.579738671538)
- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Find Duplicates of array using bit array(0.368023208756)
- Pointer to an Array | Array Pointer(0.336096927276)
- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)
- Arrays in Java(0.336096927276)

Intersection of Two Arrays II(350)

- Union and Intersection of two sorted arrays(0.411207055068)
- Find Union and Intersection of two unsorted arrays(0.411207055068)
- Find original array from encrypted array (An array of sums of other elements)(0.327966201641)
- Construct an array from its pair-sum array(0.318784021754)
- Find Duplicates of array using bit array(0.285306190981)
- Pointer to an Array | Array Pointer(0.260555671056)
- Leaders in an array(0.260555671056)
- Find pairs in array whose sums already exist in array(0.260555671056)
- Emulating a 2-d array using 1-d array(0.260555671056)
- Arrays in Java(0.260555671056)

Android Unlock Patterns(351)

- Iterator Pattern(0.260555671056)
- Command Pattern(0.260555671056)
- Adapter Pattern(0.260555671056)
- Searching for Patterns | Set 1 (Naive Pattern Searching)(0.212889950749)
- iOS vs Android(0.201993092498)
- Wildcard Pattern Matching(0.201993092498)
- Searching for Patterns | Set 4 (A Naive Pattern Searching Question)(0.201993092498)
- Flyweight Design Pattern(0.201993092498)
- Find orientation of a pattern in a matrix(0.201993092498)
- What's difference between Linux and Android ?(0.17077611319)

Data Stream as Disjoint Intervals(352)

- Data Mining(0.379978361591)
- Data Abstraction and Data Independence(0.269517613246)
- Disjoint Set Data Structures (Java Implementation)(0.260555671056)
- Linked List representation of Disjoint Set Data Structures(0.237903094633)
- Stream In Java(0.220288150562)
- Interval Tree(0.220288150562)
- Data Warehousing(0.220288150562)
- Character Stream Vs Byte Stream in Java(0.220288150562)
- Check if any two intervals overlap among a given set of intervals(0.204007612999)
- Placements | Data Interpretation(0.17077611319)

Design Snake Game(353)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.241299136472)
- Snake and Ladder Problem(0.201993092498)
- Implementation of Tic-Tac-Toe game(0.201993092498)
- Implementation of Minesweeper Game(0.201993092498)
- Hangman Game in Python(0.201993092498)
- Flyweight Design Pattern(0.201993092498)
- A Number Link Game(0.201993092498)
- The prisoner's dilemma in Game theory(0.17077611319)
- Singleton Design Pattern | Introduction(0.17077611319)
- Singleton Design Pattern | Implementation(0.17077611319)

Russian Doll Envelopes(354)

- Russian Peasant (Multiply two numbers using bitwise operators)(0.125366937987)

Design Twitter(355)

- Flyweight Design Pattern(0.260555671056)
- Twitter Interview | Set 1(0.220288150562)
- Singleton Design Pattern | Introduction(0.220288150562)
- Singleton Design Pattern | Implementation(0.220288150562)
- Designing Use Cases for a Project(0.220288150562)
- Compiler Design | Why FIRST and FOLLOW?(0.220288150562)
- Compiler Design | Runtime Environments(0.220288150562)
- Compiler Design | Lexical Analysis(0.220288150562)

- Compiler Design | Ambiguous Grammar(0.220288150562)
- Twitter Sentiment Analysis using Python(0.194314340169)

Line Reflection(356)

- Reflection in Java(0.336096927276)
- Print level order traversal line by line | Set 1(0.291219418564)
- Level order traversal line by line | Set 2 (Using Two Queues)(0.274611786436)
- Calculate Logn in one line(0.260555671056)
- Program to print last 10 lines(0.220288150562)
- Minimum lines to cover all points(0.220288150562)
- Mid-Point Line Generation Algorithm(0.220288150562)
- Count maximum points on same line(0.220288150562)
- Command line arguments in C/C++(0.220288150562)
- Command Line arguments in Java(0.220288150562)

Count Numbers with Unique Digits(357)

- Count numbers with same first and last digits(0.776514530475)
- Find count of digits in a number that divide the number(0.635001221407)
- Numbers having Unique (or Distinct) digits(0.602974816038)
- Count numbers having 0 as a digit(0.602974816038)
- Count numbers having 0 as a digit(0.602974816038)
- Count total number of N digit numbers such that the difference between sum of even and odd digits is 1(0.49089112271)
- Count ways to spell a number with repeated digits(0.450175502327)

- Count of Binary Digit numbers smaller than N(0.450175502327)
- Count digit groupings of a number with given constraints(0.450175502327)
- Count numbers from 1 to n that have 4 as a a digit(0.450175502327)

Rearrange String k Distance Apart(358)

- Check whether Strings are k distance apart or not(0.669418851727)
- Rearrange first N numbers to make them at K distance(0.380872608476)
- Hamming Distance between two strings(0.356300429333)
- Rearrange a string so that all same characters become d distance away(0.344642141038)
- Rearrange a string so that all same characters become atleast d distance away(0.31710746658)
- Rearrange characters in a string such that no two adjacent are same(0.291219418564)
- Check if edit distance between two strings is one(0.291219418564)
- Print nodes at k distance from root(0.252334201434)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.244587023615)
- Place k elements such that minimum distance is maximized(0.225764846003)

Logger Rate Limiter(359)

- SQL | LIMIT Clause(0.201993092498)
- How to overcome Time Limit Exceed(TLE)?(0.17077611319)
- Merge two BSTs with limited extra space(0.150640184987)
- Find even occurring elements in an array of limited range(0.150640184987)
- Find frequency of each element in a limited range array in less than O(n) time(0.136276341439)

- Find duplicates in a given array when elements are not limited to a range(0.136276341439)
- Project Idea | (A.T.L.A.S: App Time Limit Alerting System)(0.125366937987)
- mindepth and maxdepth in Linux find() command for limiting search to a specific directory.(0.11671773546)
- Reliance Industrial Limited Interview Experience | Set 1 (On-Campus)(0.11671773546)

Sort Transformed Array(360)

- Sort an array when two halves are sorted(0.569707709055)
- Sort a nearly sorted (or K sorted) array(0.537125579156)
- Search in an almost sorted array(0.503102612415)
- Merge two sorted arrays(0.503102612415)
- Median of two sorted arrays(0.503102612415)
- Floor in a Sorted Array(0.503102612415)
- Ceiling in a sorted array(0.503102612415)
- Generate all possible sorted arrays from alternate elements of two given sorted arrays(0.474493294343)
- Sort an almost sorted array where only two elements are swapped(0.455201845765)
- Bucket Sort To Sort an Array with Negative Numbers(0.455201845765)

Bomb Enemy(361)

Design Hit Counter(362)

- Counters in Python | Set 2 (Accessing Counters)(0.260555671056)
- Flyweight Design Pattern(0.201993092498)

- Counters in Digital Logic(0.201993092498)
- Singleton Design Pattern | Introduction(0.17077611319)
- Singleton Design Pattern | Implementation(0.17077611319)
- Designing Use Cases for a Project(0.17077611319)
- Compiler Design | Why FIRST and FOLLOW?(0.17077611319)
- Compiler Design | Runtime Environments(0.17077611319)
- Compiler Design | Lexical Analysis(0.17077611319)
- Compiler Design | Ambiguous Grammar(0.17077611319)

Max Sum of Rectangle No Larger Than K(363)

- Find sum of sum of all sub-sequences(0.27423415918)
- Sum of k smallest elements in BST(0.252334201434)
- Subset with no pair sum divisible by K(0.252334201434)
- Find k pairs with smallest sums in two arrays(0.252334201434)
- Count all sub-arrays having sum divisible by k(0.252334201434)
- Partition of a set into K subsets with equal sum(0.225764846003)
- Largest sum subarray with at-least k numbers(0.225764846003)
- Find sum of modulo K of first N natural number(0.225764846003)
- Find maximum (or minimum) sum of a subarray of size k(0.225764846003)
- Maximum value K such that array has at-least K elements that are >= K(0.220201387154)

Nested List Weight Sum II(364)

- Find sum of sum of all sub-sequences(0.27423415918)

- Find pairs with given sum in doubly linked list(0.225764846003)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.223273212851)
- Find a triplet from three linked lists with sum equal to a given number(0.206136966068)
- Sum of all Subarrays(0.194314340169)
- Find maximum sum possible equal sum of three stacks(0.194314340169)
- Recursively print all sentences that can be formed from list of word lists(0.179953413782)
- Print all possible sums of consecutive numbers with sum N(0.179953413782)
- Perfect Sum Problem (Print all subsets with given sum)(0.179953413782)
- Check if a linked list is Circular Linked List(0.168368421637)

Water and Jug Problem(365)

- The Two Water Jug Puzzle(0.503102612415)
- Tiling Problem(0.260555671056)
- The Celebrity Problem(0.260555671056)
- Nuts & Bolts Problem (Lock & Key problem)(0.260555671056)
- Gold Mine Problem(0.260555671056)
- Tree Isomorphism Problem(0.201993092498)
- Trapping Rain Water(0.201993092498)
- The Stock Span Problem(0.201993092498)
- The Lazy Caterer's Problem(0.201993092498)
- Steiner Tree Problem(0.201993092498)

Find Leaves of Binary Tree(366)

- Find first non matching leaves in two binary trees(0.656972921033)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Print all nodes in a binary tree having K leaves(0.579738671538)
- Find the maximum path sum between two leaves of a binary tree(0.579738671538)
- Find sum of all left leaves in a given Binary Tree(0.579738671538)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- Extract Leaves of a Binary Tree in a Doubly Linked List(0.524591090446)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)

Valid Perfect Square(367)

- Check perfect square using addition/subtraction(0.356300429333)
- Number of perfect squares between two given numbers(0.291069102382)
- Perfect Number(0.260555671056)
- Magic Square(0.260555671056)
- Latin Square(0.260555671056)
- Find minimum number to be divided to make a number a perfect square(0.252138706945)
- Square root of an integer(0.201993092498)
- Perfect reversible string(0.201993092498)
- Perfect cubes in a range(0.201993092498)
- Direction at last square block(0.201993092498)

Largest Divisible Subset(368)

- Largest divisible subset in array(0.776514530475)
- Subset with sum divisible by m(0.411207055068)
- Largest Subset with GCD 1(0.411207055068)
- Subset with no pair sum divisible by K(0.356300429333)
- Largest subset whose all elements are Fibonacci numbers(0.356300429333)
- Largest subset of Graph vertices with edges of 2 or more colors(0.291069102382)
- Modular Division(0.260555671056)
- Partition a set into two subsets such that the difference of subset sums is minimum(0.241299136472)
- Sum of subset differences(0.201993092498)
- Sum of average of all subsets(0.201993092498)

Plus One Linked List(369)

- Check if a linked list is Circular Linked List(0.580332984677)
- In-place Merge two linked lists without changing links of first list(0.537601087682)
- Rotate a Linked List(0.503102612415)
- Merge a linked list into another linked list at alternate positions(0.503102612415)
- Identical Linked Lists(0.503102612415)
- Flattening a Linked List(0.503102612415)
- Can we reverse a linked list in less than O(n)?(0.503102612415)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.450268144656)

- XOR Linked List – A Memory Efficient Doubly Linked List | Set 2(0.429410856634)
- XOR Linked List – A Memory Efficient Doubly Linked List | Set 1(0.429410856634)

Range Addition(370)

- Bitwise and (or &) of a range(0.336096927276)
- Binary Indexed Tree : Range Update and Range Queries(0.311257467527)
- String with additive sequence(0.260555671056)
- Range LCM Queries(0.260555671056)
- Perfect cubes in a range(0.260555671056)
- Find missing elements of a range(0.260555671056)
- Addition and Concatenation in Java(0.260555671056)
- range() vs xrange() in Python(0.220288150562)
- Min-Max Range Queries in Array(0.220288150562)
- Find the smallest twins in given range(0.220288150562)

Sum of Two Integers(371)

- Find sum of sum of all sub-sequences(0.474330706497)
- Ways to write n as sum of two or more positive integers(0.449436416524)
- How to sum two integers without using arithmetic operators in C/C++?(0.449436416524)
- Median in a stream of integers (running integers)(0.368023208756)
- Find ways an Integer can be expressed as sum of n-th power of unique natural numbers(0.355476777955)
- Sum of all Subarrays(0.336096927276)

- Find maximum sum possible equal sum of three stacks(0.336096927276)
- Sum of matrix element where each elements is integer division of row and column(0.335175743328)
- Find the smallest positive integer value that cannot be represented as sum of any subset of a given array(0.335175743328)
- Digital Root (repeated digital sum) of the given large integer(0.335175743328)

Super Pow(372)

- Super Prime(0.336096927276)
- Super Keyword in Java(0.260555671056)
- Calculate square of a number without using *, / and pow()(0.194314340169)
- scanf() and fscanf() in C – Simple Yet Poweful(0.175786078393)
- Accessing Grandparent's member in Java using super(0.175786078393)
- OOP in Python | Set 3 (Inheritance, examples of object, issubclass and super)(0.141430567926)
- DBMS | Keys in Relational Model (Candidate, Super, Primary, Alternate and Foreign)(0.141430567926)
- Super Ugly Number (Number whose prime factors are in given set)(0.133785092946)

Find K Pairs with Smallest Sums(373)

- Find k pairs with smallest sums in two arrays(0.818180207367)
- Sum of k smallest elements in BST(0.51014901931)
- Subset with no pair sum divisible by K(0.51014901931)
- Check if an array can be divided into pairs whose sum is divisible by k(0.407352604289)
- Check if a sorted array can be divided in pairs whose sum is k(0.407352604289)

- Permute two arrays such that sum of every pair is greater or equal to K(0.374807770059)
- Maximum sum of smallest and second smallest in an array(0.372055731454)
- Find pairs with given sum such that elements of pair are in different rows(0.342390186113)
- Sum of bit differences among all pairs(0.336096927276)
- Count pairs with given sum(0.336096927276)

Guess Number Higher or Lower(374)

- Next higher number with same number of set bits(0.411065370983)
- Smallest number divisible by first n numbers(0.241213606675)
- Number with maximum number of prime factors(0.241213606675)
- Number of subtrees having odd count of even numbers(0.241213606675)
- Number of perfect squares between two given numbers(0.241213606675)
- How to check if a given number is Fibonacci number?(0.241213606675)
- Finding number of digits in n'th Fibonacci number(0.241213606675)
- Find the missing number in a string of numbers with no separator(0.241213606675)
- Find the Number Occurring Odd Number of Times(0.241213606675)
- Find count of digits in a number that divide the number(0.241213606675)

Guess Number Higher or Lower II(375)

- Next higher number with same number of set bits(0.356177663686)
- Smallest number divisible by first n numbers(0.212772510465)
- Number with maximum number of prime factors(0.212772510465)

- Number of subtrees having odd count of even numbers(0.212772510465)
- Number of perfect squares between two given numbers(0.212772510465)
- How to check if a given number is Fibonacci number?(0.212772510465)
- Finding number of digits in n'th Fibonacci number(0.212772510465)
- Find the missing number in a string of numbers with no separator(0.212772510465)
- Find the Number Occurring Odd Number of Times(0.212772510465)
- Find count of digits in a number that divide the number(0.212772510465)

Wiggle Subsequence(376)

- Shortest Uncommon Subsequence(0.260555671056)
- Queries on subsequence of string(0.260555671056)
- Longest alternating subsequence(0.260555671056)
- Longest Zig-Zag Subsequence(0.260555671056)
- Longest Repeating Subsequence(0.260555671056)
- Longest Consecutive Subsequence(0.260555671056)
- Count all increasing subsequences(0.260555671056)
- Count Distinct Subsequences(0.260555671056)
- Subsequence with maximum odd sum(0.220288150562)
- Repeated subsequence of length 2 or more(0.220288150562)

Combination Sum IV(377)

- Find sum of sum of all sub-sequences(0.36771998047)
- Sum of all Subarrays(0.260555671056)

- Find maximum sum possible equal sum of three stacks(0.260555671056)
- Combinations with repetitions(0.260555671056)
- Print all possible sums of consecutive numbers with sum N(0.241299136472)
- Perfect Sum Problem (Print all subsets with given sum)(0.241299136472)
- Print all n-digit numbers whose sum of digits equals to given sum(0.225764846003)
- Finding sum of digits of a number until sum becomes single digit(0.212889950749)
- Sum of two large numbers(0.201993092498)
- Sum of subset differences(0.201993092498)

Kth Smallest Element in a Sorted Matrix(378)

- Search element in a sorted matrix(0.51014901931)
- K-th Element of Two Sorted Arrays(0.51014901931)
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1(0.410995463935)
- k-th smallest absolute difference of two elements in an array(0.380872608476)
- Print all elements in sorted order from row and column wise sorted matrix(0.380773967693)
- Rotate Matrix Elements(0.356300429333)
- Find the smallest and second smallest elements in an array(0.356177663686)
- K-th smallest element after removing some integers from natural numbers(0.344642141038)
- Find a common element in all rows of a given row-wise sorted matrix(0.344642141038)
- Sum of all elements between k1'th and k2'th smallest elements(0.3223768056)

Design Phone Directory(379)

- Implement a Phone Directory(0.503102612415)
- Flyweight Design Pattern(0.201993092498)
- Singleton Design Pattern | Introduction(0.17077611319)
- Singleton Design Pattern | Implementation(0.17077611319)
- Designing Use Cases for a Project(0.17077611319)
- Compiler Design | Why FIRST and FOLLOW?(0.17077611319)
- Compiler Design | Runtime Environments(0.17077611319)
- Compiler Design | Lexical Analysis(0.17077611319)
- Compiler Design | Ambiguous Grammar(0.17077611319)
- Print all possible words from phone digits(0.150640184987)

Insert Delete GetRandom O(1)(380)

- Design a data structure that supports insert, delete, search and getRandom in constant time(0.327870747184)
- Search, insert and delete in an unsorted array(0.291219418564)
- Search, insert and delete in a sorted array(0.291219418564)
- Insertion and Deletion in STL Set C++(0.291219418564)
- Treap | Set 2 (Implementation of Search, Insert and Delete)(0.237903094633)
- Minimum number of deletions and insertions to transform one string into another(0.237903094633)
- Efficiently design Insert, Delete and Median queries on a set(0.237903094633)
- Trie | (Delete)(0.220288150562)
- Insertion Sort(0.220288150562)
- Inserting elements in std::map (insert, emplace and operator [])(0.220288150562)

Insert Delete GetRandom O(1) - Duplicates allowed(381)

- Design a data structure that supports insert, delete, search and getRandom in constant time(0.244785311735)
- Search, insert and delete in an unsorted array(0.225764846003)
- Search, insert and delete in a sorted array(0.225764846003)
- Insertion and Deletion in STL Set C++(0.225764846003)
- Find a Fixed Point in an array with duplicates allowed(0.225764846003)
- Treap | Set 2 (Implementation of Search, Insert and Delete)(0.184431916623)
- Minimum number of deletions and insertions to transform one string into another(0.184431916623)
- Find duplicates in O(n) time and O(1) extra space | Set 1(0.184431916623)
- Efficiently design Insert, Delete and Median queries on a set(0.184431916623)
- Duplicates in an array in O(n) and by using O(1) extra space | Set-2(0.184431916623)

Linked List Random Node(382)

- Select a Random Node from a Singly Linked List(0.709297266606)
- Delete N nodes after M nodes of a linked list(0.519280018803)
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes(0.515936647418)
- Segregate even and odd nodes in a Linked List(0.51014901931)
- Find n'th node from the end of a Linked List(0.51014901931)
- Delete alternate nodes of a Linked List(0.51014901931)
- Delete a node in a Doubly Linked List(0.51014901931)
- Check if a linked list is Circular Linked List(0.474330706497)

- Write a function to get Nth node in a Linked List(0.450175502327)
- Remove every k-th node of the linked list(0.450175502327)

Ransom Note(383)

- Last Minute Notes – Operating Systems(0.220288150562)
- Last Minute Notes – DBMS(0.220288150562)
- Last Minute Notes – Theory of Computation(0.194314340169)
- Last Minute Notes – Engineering Mathematics(0.194314340169)
- Last Minute Notes – Computer Networks(0.194314340169)
- Puzzle 33 | (Rs 500 Note Puzzle)(0.150556969602)

Shuffle an Array(384)

- Shuffle a given array(0.709297266606)
- Find missing number in another array which is shuffled copy(0.449436416524)
- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Find Duplicates of array using bit array(0.368023208756)
- Pointer to an Array | Array Pointer(0.336096927276)
- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)
- Arrays in Java(0.336096927276)

Mini Parser(385)

- StAX XML Parser in Java(0.220288150562)

- Parsing | Set 2 (Bottom Up or Shift Reduce Parsers)(0.175786078393)
- Parsing | Set 1 (Introduction, Ambiguity and Parsers)(0.175786078393)
- Parsing | Set 3 (SLR, CLR and LALR Parsers)(0.161713780663)

Lexicographical Numbers(386)

- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)
- Find count of digits in a number that divide the number(0.368023208756)

First Unique Character in a String(387)

- Determine if a string has all Unique Characters(0.656972921033)
- Find the longest substring with k unique characters in a given string(0.524591090446)
- Find uncommon characters of the two strings(0.503102612415)
- Find the first repeated character in a string(0.503102612415)
- String matching where one string contains wildcard characters(0.455201845765)

- Remove characters from the first string which are present in the second string(0.455201845765)
- Find the smallest window in a string containing all characters of another string(0.418906716157)
- Rearrange characters in a string such that no two adjacent are same(0.411207055068)
- Queries for characters in a repeated string(0.411207055068)
- Program to toggle all characters in a string(0.411207055068)

Longest Absolute File Path(388)

- Longest path in an undirected tree(0.336096927276)
- Longest path between any pair of vertices(0.291219418564)
- Longest Path in a Directed Acyclic Graph(0.291219418564)
- Find the longest path in a matrix with given constraints(0.291219418564)
- Longest Path in a Directed Acyclic Graph | Set 2(0.237903094633)
- Find length of the longest consecutive path from a given starting character(0.237903094633)
- Printing Paths in Dijkstra's Shortest Path Algorithm(0.220288150562)
- File Systems | Operating System(0.220288150562)
- Dyck path(0.220288150562)
- C Program to merge contents of two files into a third file(0.220288150562)

Find the Difference(389)

- Sum of subset differences(0.449436416524)
- K difference permutation(0.449436416524)
- Find difference between sums of two diagonals(0.449436416524)

- Find a pair with the given difference(0.449436416524)
- Difference of two large numbers(0.449436416524)
- Windows 10 –Feel the Difference(0.379978361591)
- What's difference between The Internet and The Web ?(0.379978361591)
- What's difference between Ping and Traceroute?(0.379978361591)
- What's difference between MMU and MPU?(0.379978361591)
- What's difference between Linux and Android ?(0.379978361591)

Elimination Game(390)

- Tail Call Elimination(0.336096927276)
- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)
- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)

Perfect Rectangle(391)

- Perfect Number(0.336096927276)
- Find if two rectangles overlap(0.336096927276)
- Perfect reversible string(0.260555671056)

- Perfect cubes in a range(0.260555671056)
- Count number of squares in a rectangle(0.220288150562)
- Count all perfect divisors of a number(0.220288150562)
- Check if four segments form a rectangle(0.220288150562)
- Check whether a given binary tree is perfect or not(0.194314340169)
- Check perfect square using addition/subtraction(0.194314340169)
- Reverse alternate levels of a perfect binary tree(0.175786078393)

Is Subsequence(392)

- Shortest Uncommon Subsequence(0.449436416524)
- Queries on subsequence of string(0.449436416524)
- Longest alternating subsequence(0.449436416524)
- Longest Zig-Zag Subsequence(0.449436416524)
- Longest Repeating Subsequence(0.449436416524)
- Longest Consecutive Subsequence(0.449436416524)
- Count all increasing subsequences(0.449436416524)
- Count Distinct Subsequences(0.449436416524)
- Subsequence with maximum odd sum(0.379978361591)
- Repeated subsequence of length 2 or more(0.379978361591)

UTF-8 Validation(393)

- Valid variants of main() in Java(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)

- Length of the longest valid substring(0.220288150562)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)
- How to check if a string is a valid keyword in Python?(0.194314340169)
- Find the number of valid parentheses expressions of given length(0.175786078393)
- Print all valid words that are possible using Characters of Array(0.161713780663)
- Check if a given string is a valid number (Integer or Floating Point)(0.150556969602)
- Check if a given string is a valid number (Integer or Floating Point) in Java(0.141430567926)

Decode String(394)

- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)
- Huffman Decoding(0.336096927276)
- Check if given string can be split into four distinct strings(0.336096927276)
- Sort a string according to the order defined by another string(0.311257467527)

Longest Substring with At Least K Repeating Characters(395)

- Length of the longest substring without repeating characters(0.669418851727)
- Find the longest substring with k unique characters in a given string(0.519387993313)
- Longest repeating and non-overlapping substring(0.51014901931)
- Longest Repeating Subsequence(0.356300429333)
- Longest Non-palindromic substring(0.356300429333)
- Find the first repeated character in a string(0.356300429333)
- Count substrings with same first and last characters(0.356300429333)
- Count number of substrings with exactly k distinct characters(0.344642141038)
- Suffix Tree Application 3 – Longest Repeated Substring(0.31710746658)
- Queries for characters in a repeated string(0.291219418564)

Rotate Function(396)

- SQL | Functions (Aggregate and Scalar Functions)(0.368023208756)
- Left Rotation and Right Rotation of a String(0.368023208756)
- Find the Rotation Count in Rotated Sorted array(0.368023208756)
- Reentrant Function(0.336096927276)
- Recursive Functions(0.336096927276)
- Pure Functions(0.336096927276)
- Functions in C/C++(0.336096927276)
- Decimal Functions in Python | Set 2 (logical_and(), normalize(), quantize(), rotate() …)(0.335175743328)
- What happens when a virtual function is called inside a non-virtual function in C++(0.311257467527)

- Mathematical Functions in Python | Set 1 (Numeric Functions)(0.311257467527)

Integer Replacement(397)

- Replace all ‘0’ with ‘5’ in an input Integer(0.502328778226)
- Median in a stream of integers (running integers)(0.368023208756)
- Count of m digit integers that are divisible by an integer n(0.311257467527)
- Square root of an integer(0.260555671056)
- Sorting Big Integers(0.260555671056)
- Smallest Palindrome after replacement(0.260555671056)
- Integer Promotions in C(0.260555671056)
- Check for Integer Overflow(0.260555671056)
- Smallest of three integers without comparison operators(0.220288150562)
- Printing Integer between Strings in Java(0.220288150562)

Random Pick Index(398)

- Random vs Secure Random numbers in Java(0.260555671056)
- Random Variable(0.260555671056)
- Random Numbers in Python(0.201993092498)
- Equilibrium index of an array(0.201993092498)
- Test Case Generation | Set 2 (Random Characters, Strings and Arrays of Random Strings)(0.177210610839)
- Randomized Binary Search Algorithm(0.17077611319)
- Indexing in Databases | Set 1(0.17077611319)
- Generating random numbers in Java(0.17077611319)

- random header | Set 2 (Distributions)(0.150640184987)
- random header in C++ | Set 1(Generators)(0.150640184987)

Evaluate Division(399)

- Modular Division(0.336096927276)
- Expression Evaluation(0.336096927276)
- Evaluation order of operands(0.260555671056)
- Evaluation of Expression Tree(0.260555671056)
- Division Operators in Python(0.260555671056)
- DFA based division(0.260555671056)
- Check divisibility by 7(0.260555671056)
- Subset with sum divisible by m(0.220288150562)
- Sub-string Divisibility by 3 Queries(0.220288150562)
- Sub-string Divisibility by 11 Queries(0.220288150562)

Nth Digit(400)

- Find the n-th number made of even digits only(0.579738671538)
- Check if frequency of each digit is less than the digit(0.411207055068)
- Find n-th element in a series with only 2 digits (4 and 7) allowed(0.355476777955)
- Count 'd' digit positive integers with 0 as a digit(0.311257467527)
- Generate k digit numbers with digits in strictly increasing order(0.291219418564)
- Digital Root (repeated digital sum) of the given large integer(0.291219418564)
- Count positive integers with 0 as a digit and maximum 'd' digits(0.291219418564)

- Finding sum of digits of a number until sum becomes single digit(0.274611786436)
- Find the Largest number with given number of digits and sum of digits(0.274611786436)
- Find smallest number with given number of digits and sum of digits(0.274611786436)

Binary Watch(401)

- Binary Search(0.336096927276)
- Binary Heap(0.336096927276)
- Gray to Binary and Binary to Gray conversion(0.311257467527)
- Check whether a binary tree is a full binary tree or not(0.311257467527)
- Binary Tree to Binary Search Tree Conversion(0.291219418564)
- Binary Tree | Set 3 (Types of Binary Tree)(0.274611786436)
- fork() and Binary Tree(0.260555671056)
- Threaded Binary Tree(0.260555671056)
- Foldable Binary Trees(0.260555671056)
- Flip Binary Tree(0.260555671056)

Remove K Digits(402)

- N'th palindrome of K digits(0.411207055068)
- Generate k digit numbers with digits in strictly increasing order(0.390105265183)
- Remove repeated digits in a given number(0.356300429333)
- Remove recurring digits in a given number(0.356300429333)

- Given a number n, find the first k digits of nˆn(0.318784021754)
- Check if frequency of each digit is less than the digit(0.318784021754)
- Maximum value K such that array has at-least K elements that are >= K(0.295267555382)
- Remove nodes on root to leaf paths of length < K(0.291069102382)
- Print first k digits of 1/n where n is a positive integer(0.291069102382)
- Remove all nodes which don't lie in any path with sum>= k(0.269517613246)

Frog Jump(403)

- Jump Search(0.336096927276)
- Minimum number of jumps to reach end(0.194314340169)
- Minimum block jumps to reach destination(0.194314340169)
- Count number of ways to jump to reach end(0.175786078393)
- Print all Jumping Numbers smaller than or equal to a given value(0.161713780663)
- Maximum path sum for each position with jumps under divisibility condition(0.161713780663)
- Minimum number of jumps to reach end | Set 2 (O(n) solution)(0.150556969602)
- Decision Making in Java (if, if-else, switch, break, continue, jump)(0.150556969602)

Sum of Left Leaves(404)

- Find sum of all left leaves in a given Binary Tree(0.579738671538)
- Find sum of sum of all sub-sequences(0.36771998047)
- Find multiplication of sums of data of leaves at same levels(0.356300429333)
- Find the maximum path sum between two leaves of a binary tree(0.318784021754)

- Sum of all Subarrays(0.260555671056)
- Find maximum sum possible equal sum of three stacks(0.260555671056)
- Print all possible sums of consecutive numbers with sum N(0.241299136472)
- Perfect Sum Problem (Print all subsets with given sum)(0.241299136472)
- Print all n-digit numbers whose sum of digits equals to given sum(0.225764846003)
- Change a Binary Tree so that every node stores sum of all nodes in left subtree(0.215070325706)

Convert a Number to Hexadecimal(405)

- Convert a binary number to hexadecimal number(0.817758324521)
- Converting Strings to Numbers in C/C++(0.411207055068)
- Minimum number of operation required to convert number x into y(0.390105265183)
- Convert a number m to n using minimum number of given operations(0.366529477546)
- Program to convert a given number to words(0.356300429333)
- Converting string to number and vice-versa in C++(0.356300429333)
- Convert decimal fraction to binary number(0.356300429333)
- Convert a number into negative base representation(0.356300429333)
- What is the best way in C to convert a number to a string?(0.318784021754)
- Count number of bits to be flipped to convert A to B(0.318784021754)

Queue Reconstruction by Height(406)

- Reconstructing Segment Tree(0.201993092498)

- Queue Interface In Java(0.201993092498)
- Applications of Priority Queue(0.201993092498)
- Roots of a tree which give minimum height(0.17077611319)
- Minimize the maximum difference between the heights(0.17077611319)
- Implement Stack using Queues(0.17077611319)
- Implement Queue using Stacks(0.17077611319)
- Heap queue (or heapq) in Python(0.17077611319)
- Applications of Queue Data Structure(0.17077611319)
- Priority Queue | Set 1 (Introduction)(0.150640184987)

Trapping Rain Water II(407)

- Trapping Rain Water(0.776514530475)
- The Two Water Jug Puzzle(0.17077611319)
- Program to find amount of water in a given glass(0.144383555277)
- Measuring 6L water from 4L and 9L buckets(0.115215543378)
- Flipkart Interview | Set 7 (For SDE II)(0.115215543378)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.105992131351)
- Measure one litre using two vessels and infinite water supply(0.105992131351)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.105992131351)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.105992131351)
- Amazon Interview experience | Set 326 (For SDE II)(0.105992131351)

Valid Word Abbreviation(408)

- Print all valid words that are possible using Characters of Array(0.291069102382)

- Word formation using concatenation of two dictionary words(0.260555671056)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Length Of Last Word in a String(0.201993092498)
- Valid variants of main() in Java(0.17077611319)
- Reverse words in a given string(0.17077611319)
- Program to validate an IP address(0.17077611319)

Longest Palindrome(409)

- Longest Palindromic Substring | Set 2(0.502328778226)
- Longest Palindromic Substring | Set 1(0.502328778226)
- Find longest palindrome formed by removing or shuffling chars from string(0.410362644952)
- Dynamic Programming | Set 12 (Longest Palindromic Subsequence)(0.410362644952)
- Suffix Tree Application 6 – Longest Palindromic Substring(0.379978361591)
- Palindromic Primes(0.336096927276)
- Manacher's Algorithm – Linear Time Longest Palindromic Substring – Part 4(0.30321606445)
- Manacher's Algorithm – Linear Time Longest Palindromic Substring – Part 3(0.30321606445)
- Manacher's Algorithm – Linear Time Longest Palindromic Substring – Part 2(0.30321606445)
- Manacher's Algorithm – Linear Time Longest Palindromic Substring – Part 1(0.30321606445)

Split Array Largest Sum(410)

- Split an array into two equal Sum subarrays(0.51014901931)
- Find the largest pair sum in an unsorted array(0.51014901931)
- Find the largest three elements in an array(0.411207055068)
- Find original array from encrypted array (An array of sums of other elements)(0.410888471656)
- Check if there exist two elements in an array whose sum is equal to the sum of rest of the array(0.387823448738)
- Find pairs in array whose sums already exist in array(0.372055731454)
- Program to find largest element in an array(0.336096927276)
- Maximum Sum Path in Two Arrays(0.336096927276)
- Largest divisible subset in array(0.336096927276)
- Largest Sum Contiguous Subarray(0.336096927276)

Minimum Unique Word Abbreviation(411)

- C++ program to print unique words in a file(0.260555671056)
- Word formation using concatenation of two dictionary words(0.220288150562)
- Second minimum element using minimum comparisons(0.220288150562)
- Maximum and minimum of an array using minimum number of comparisons(0.204007612999)
- Word Ladder (Length of shortest chain to reach a target word)(0.19087406613)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.19087406613)
- C program to Replace a word in a text by another given word(0.19087406613)
- Find shortest unique prefix for every word in a given list | Set 2 (Using Sorting)(0.184355541926)
- Find shortest unique prefix for every word in a given list | Set 1 (Using

Trie)(0.184355541926)

- Testimonials – Words that keep us going(0.17077611319)

Fizz Buzz(412)

- Fizz Buzz Implementation(0.709297266606)

Arithmetic Slices(413)

- Object Slicing in C++(0.260555671056)
- Python List Comprehension and Slicing(0.220288150562)
- Multidimensional Pointer Arithmetic in C/C++(0.220288150562)
- Find the missing number in Arithmetic Progression(0.220288150562)
- Subtract two numbers without using arithmetic operators(0.194314340169)
- Draw a circle without floating point arithmetic(0.194314340169)
- Draw a circle without floating point arithmetic(0.194314340169)
- Computer Arithmetic | Set – 2(0.194314340169)
- Computer Arithmetic | Set – 1(0.194314340169)
- Add two numbers without using arithmetic operators(0.194314340169)

Third Maximum Number(414)

- Find the maximum number of handshakes(0.709297266606)
- Number with maximum number of prime factors(0.709052873586)
- Maximum sum of distinct numbers such that LCM of these numbers is N(0.590594008858)
- Level with maximum number of nodes(0.579738671538)
- Find the row with maximum number of 1s(0.579738671538)

- Querying maximum number of divisors that a number in a given range has(0.549988394922)
- Maximum number of Zombie process a system can handle(0.502328778226)
- Recursively break a number in 3 parts to get maximum sum(0.449436416524)
- Maximum sum of distinct numbers with LCM as N(0.449436416524)
- Maximum sum of a path in a Right Number Triangle(0.449436416524)

Add Strings(415)

- Add two bit strings(0.709297266606)
- Program to add two binary strings(0.579738671538)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)
- Check if given string can be split into four distinct strings(0.336096927276)

Partition Equal Subset Sum(416)

- Partition of a set into K subsets with equal sum(0.709297266606)
- Partition a set into two subsets such that the difference of subset sums is minimum(0.519280018803)

- Sum of subset differences(0.411207055068)
- Sum of average of all subsets(0.411207055068)
- Equal Sum and XOR(0.411207055068)
- Find maximum sum possible equal sum of three stacks(0.372055731454)
- Perfect Sum Problem (Print all subsets with given sum)(0.342390186113)
- Sum of the products of all possible Subsets(0.336096927276)
- Sum of maximum elements of all subsets(0.336096927276)
- Sum of XOR of all possible subsets(0.336096927276)

Pacific Atlantic Water Flow(417)

- Trapping Rain Water(0.17077611319)
- The Two Water Jug Puzzle(0.17077611319)
- Program to find amount of water in a given glass(0.144383555277)
- Max Flow Problem Introduction(0.144383555277)
- Dinic's algorithm for Maximum Flow(0.144383555277)
- Ford-Fulkerson Algorithm for Maximum Flow Problem(0.127359529795)
- Find minimum s-t cut in a flow network(0.127359529795)
- Measuring 6L water from 4L and 9L buckets(0.115215543378)
- Flow control in try catch finally in Java(0.115215543378)
- Measure one litre using two vessels and infinite water supply(0.105992131351)

Sentence Screen Fitting(418)

- Puzzle 67 | Fit Triangle(0.17077611319)
- Project Idea | League of Fitness(0.17077611319)

- Program for First Fit algorithm in Memory Management(0.150640184987)
- Program for Worst Fit algorithm in Memory Management(0.136276341439)
- Program for Best Fit algorithm in Memory Management(0.136276341439)
- Print shortest path to print a string on screen(0.11671773546)
- Recursively print all sentences that can be formed from list of word lists(0.109642586835)
- Maximum number of 2×2 squares that can be fit inside a right isosceles triangle(0.109642586835)
- Check a given sentence for a given set of simple grammer rules(0.103715511333)

Battleships in a Board(419)

- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)
- Minimum Cost to cut a board into squares(0.194314340169)
- Boggle (Find all possible words in a board of characters) | Set 1(0.161713780663)
- Abco Advisory Board Company | Set 2 (On-Campus)(0.161713780663)
- Abco Advisory Board Company | Set 3 (On-Campus Intern + FTE)(0.141430567926)
- Abco Advisory Board Company | Set 1 (Internship + Full time Employee)(0.141430567926)

Strong Password Checker(420)

- Ideas for Strong Recoverable Passwords(0.411207055068)
- Recover password of password protected zip file(0.260555671056)
- How to store a password in database?(0.201993092498)
- Program to check Strong Number(0.17077611319)

- Passwords and Cryptographic hash function(0.17077611319)
- Generating Password and OTP in Java(0.17077611319)
- getpass() and getuser() in Python (Password without echo)(0.150640184987)
- Data Structure for Dictionary and Spell Checker?(0.150640184987)
- To Generate a One Time Password or Unique Identification URL(0.136276341439)
- Reset a lost Linux administrative password and Explanation(0.136276341439)

Maximum XOR of Two Numbers in an Array(421)

- Find the maximum subarray XOR in a given array(0.51014901931)
- Break an array into maximum number of sub-arrays such that their averages are same(0.450175502327)
- GCD of more than two (or array) numbers(0.411207055068)
- Find the maximum number of handshakes(0.411207055068)
- Number with maximum number of prime factors(0.411065370983)
- Find XOR of two number without using XOR operator(0.411065370983)
- Maximum and minimum of an array using minimum number of comparisons(0.348993907955)
- Maximum sum of distinct numbers such that LCM of these numbers is N(0.342390186113)
- Type of array and its maximum element(0.336096927276)
- Sort an array of large numbers(0.336096927276)

Valid Word Square(422)

- Print all valid words that are possible using Characters of Array(0.291069102382)
- Word formation using concatenation of two dictionary words(0.260555671056)

- Magic Square(0.260555671056)
- Latin Square(0.260555671056)
- Word Ladder (Length of shortest chain to reach a target word)(0.225764846003)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.225764846003)
- C program to Replace a word in a text by another given word(0.225764846003)
- Testimonials – Words that keep us going(0.201993092498)
- Square root of an integer(0.201993092498)
- Length Of Last Word in a String(0.201993092498)

Reconstruct Original Digits from English(423)

- Check if frequency of each digit is less than the digit(0.269517613246)
- Count 'd' digit positive integers with 0 as a digit(0.204007612999)
- Generate k digit numbers with digits in strictly increasing order(0.19087406613)
- Digital Root (repeated digital sum) of the given large integer(0.19087406613)
- Count positive integers with 0 as a digit and maximum 'd' digits(0.19087406613)
- Finding sum of digits of a number until sum becomes single digit(0.179988918812)
- Find the Largest number with given number of digits and sum of digits(0.179988918812)
- Find smallest number with given number of digits and sum of digits(0.179988918812)
- Reconstructing Segment Tree(0.17077611319)
- Placements | English | Fill in the Blanks(0.17077611319)

Longest Repeating Character Replacement(424)

- Length of the longest substring without repeating characters(0.51014901931)
- Longest Repeating Subsequence(0.411207055068)
- Find the first repeated character in a string(0.411207055068)
- Queries for characters in a repeated string(0.336096927276)
- Longest repeating and non-overlapping substring(0.336096927276)
- Longest Common Prefix | Set 2 (Character by Character Matching)(0.318849541433)
- Maximum consecutive repeating character in string(0.291219418564)
- Find the first non-repeating character from a stream of characters(0.269517613246)
- Check for Palindrome after every character replacement Query(0.260555671056)
- Smallest length string with repeated replacement of two distinct adjacent(0.237903094633)

Word Squares(425)

- Word formation using concatenation of two dictionary words(0.336096927276)
- Magic Square(0.336096927276)
- Latin Square(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.291219418564)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Square root of an integer(0.260555671056)
- Length Of Last Word in a String(0.260555671056)

- Direction at last square block(0.260555671056)

All O‘one Data Structure(432)

- Persistent data structures(0.503102612415)
- Overview of Data Structures | Set 1 (Linear Data Structures)(0.503102612415)
- Data Mining(0.449436416524)
- Applications of tree data structure(0.411207055068)
- Applications of Queue Data Structure(0.411207055068)
- Applications of Heap Data Structure(0.411207055068)
- Stack Data Structure (Introduction and Program)(0.356300429333)
- Data Structures | Stack | Question 8(0.356300429333)
- Data Structures | Stack | Question 7(0.356300429333)
- Data Structures | Stack | Question 6(0.356300429333)

Number of Segments in a String(434)

- Find the missing number in a string of numbers with no separator(0.502929265114)
- Find number of times a string occurs as a subsequence in given string(0.418906716157)
- Number of even substrings in a string of digits(0.411207055068)
- Number of distinct permutation a String can have(0.411207055068)
- Converting Strings to Numbers in C/C++(0.411207055068)
- Given a number as a string, find the number of contiguous subsequences which recursively add up to 9(0.366529477546)
- Numbers in Java (With 0 Prefix and with Strings)(0.356300429333)

- Number of subsequences in a string divisible by n(0.356300429333)
- Multiply Large Numbers represented as Strings(0.356300429333)
- Converting string to number and vice-versa in C++(0.356300429333)

Non-overlapping Intervals(435)

- Interval Tree(0.336096927276)
- Check if any two intervals overlap among a given set of intervals(0.311257467527)
- Merge Overlapping Intervals(0.260555671056)
- Longest repeating and non-overlapping substring(0.220288150562)
- Find the point where maximum intervals overlap(0.220288150562)
- Minimum distance to travel to cover all intervals(0.194314340169)
- Maximum sum two non-overlapping subarrays of given size(0.175786078393)

Find Right Interval(436)

- Interval Tree(0.336096927276)
- Check if any two intervals overlap among a given set of intervals(0.311257467527)
- Print a matrix in alternate manner (left to right then right to left)(0.260555671056)
- Merge Overlapping Intervals(0.260555671056)
- Find other two sides of a right angle triangle(0.260555671056)
- Find the point where maximum intervals overlap(0.220288150562)
- Find next right node of a given key(0.220288150562)
- Count smaller elements on right side(0.220288150562)

- Print Right View of a Binary Tree(0.194314340169)
- Minimum distance to travel to cover all intervals(0.194314340169)

Path Sum III(437)

- Maximum path sum in a triangle.(0.411207055068)
- Maximum Sum Path in Two Arrays(0.411207055068)
- Find sum of sum of all sub-sequences(0.36771998047)
- Minimum Sum Path In 3-D Array(0.356300429333)
- Maximum Path Sum in a Binary Tree(0.356300429333)
- Sum of all the numbers that are formed from root to leaf paths(0.318784021754)
- Maximum sum of a path in a Right Number Triangle(0.318784021754)
- Find the maximum path sum between two leaves of a binary tree(0.318784021754)
- Root to leaf path sum equal to a given number(0.291069102382)
- Print all the paths from root, with a specified sum in Binary tree(0.291069102382)

Find All Anagrams in a String(438)

- Check whether two strings are anagram of each other(0.709297266606)
- Print all pairs of anagrams in a given array of strings(0.449436416524)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Remove minimum number of characters so that two strings become anagram(0.410362644952)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)

- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)

Ternary Expression Parser(439)

- Convert Ternary Expression to a Binary Tree(0.356300429333)
- Expression Tree(0.260555671056)
- Expression Evaluation(0.260555671056)
- Ternary Search Tree(0.201993092498)
- Ternary Operator in Python(0.201993092498)
- Regular Expressions in Java(0.201993092498)
- Lambda expression in C++(0.201993092498)
- How to write Regular Expressions?(0.201993092498)
- Evaluation of Expression Tree(0.201993092498)
- StAX XML Parser in Java(0.17077611319)

K-th Smallest in Lexicographical Order(440)

- Find k-th smallest element in BST (Order Statistics in BST)(0.348993907955)
- Print all permutations in sorted (lexicographic) order(0.291219418564)
- k-th smallest absolute difference of two elements in an array(0.260555671056)
- Print all longest common sub-sequences in lexicographical order(0.260555671056)

- Find the smallest and second smallest elements in an array(0.241213606675)
- Lexicographically smallest array after at-most K consecutive swaps(0.237903094633)
- K-th smallest element after removing some integers from natural numbers(0.237903094633)
- SQL | ORDER BY(0.220288150562)
- Multiplicative order(0.220288150562)
- Maximum sum of smallest and second smallest in an array(0.220288150562)

Arranging Coins(441)

- Maximum height when coins are arranged in a triangle(0.502328778226)
- Make a fair coin from a biased coin(0.368023208756)
- Minimum cost for acquiring all coins with k extra coins allowed with every coin(0.364020643353)
- Frobenius coin problem(0.260555671056)
- Decision Trees – Fake (Counterfeit) Coin Puzzle (12 Coin Puzzle)(0.237903094633)
- Puzzle 53 | The Counterfeit Coin(0.220288150562)
- Program to print an array in Pendulum Arrangement(0.194314340169)
- OpenCV C++ Program for coin detection(0.194314340169)
- Number of paths with exactly k coins(0.194314340169)
- Greedy Algorithm to find Minimum number of Coins(0.194314340169)

Find All Duplicates in an Array(442)

- Find Duplicates of array using bit array(0.709052873586)
- Remove duplicates from sorted array(0.579738671538)

- Find lost element from a duplicated array(0.579738671538)
- Remove duplicates from an array of small primes(0.502328778226)
- Find a Fixed Point in an array with duplicates allowed(0.502328778226)
- Find duplicates in a given array when elements are not limited to a range(0.449436416524)
- Find Equal (or Middle) Point in a sorted array with duplicates(0.449436416524)
- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Duplicates in an array in O(n) and by using O(1) extra space | Set-2(0.410362644952)

Sequence Reconstruction(444)

- Recaman's sequence(0.336096927276)
- Padovan Sequence(0.336096927276)
- Look-and-Say Sequence(0.336096927276)
- Juggler Sequence(0.336096927276)
- Farey Sequence(0.336096927276)
- Aliquot Sequence(0.336096927276)
- String with additive sequence(0.260555671056)
- Reconstructing Segment Tree(0.260555671056)
- Jolly Jumper Sequence(0.260555671056)
- Find Recurring Sequence in a Fraction(0.260555671056)

Add Two Numbers II(445)

- Add 1 to a given number(0.411207055068)

- Given a number as a string, find the number of contiguous subsequences which recursively add up to 9(0.366529477546)
- Find all combinations that add upto given number(0.356300429333)
- Add two numbers without using arithmetic operators(0.356300429333)
- Add two numbers using ++ and/or —(0.356300429333)
- Write a program to add two numbers in base 14(0.318784021754)
- Add 1 to a number represented as linked list(0.318784021754)
- Add two numbers represented by linked lists | Set 2(0.291069102382)
- Add two numbers represented by linked lists | Set 1(0.291069102382)
- Smallest number divisible by first n numbers(0.285306190981)

Arithmetic Slices II - Subsequence(446)

- Count of AP (Arithmetic Progression) Subsequences in an array(0.260555671056)
- Shortest Uncommon Subsequence(0.17077611319)
- Queries on subsequence of string(0.17077611319)
- Object Slicing in C++(0.17077611319)
- Longest alternating subsequence(0.17077611319)
- Longest Zig-Zag Subsequence(0.17077611319)
- Longest Repeating Subsequence(0.17077611319)
- Longest Consecutive Subsequence(0.17077611319)
- Count all increasing subsequences(0.17077611319)
- Count Distinct Subsequences(0.17077611319)

Number of Boomerangs(447)

- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)
- Find count of digits in a number that divide the number(0.368023208756)

Find All Numbers Disappeared in an Array(448)

- GCD of more than two (or array) numbers(0.503102612415)
- Sort an array of large numbers(0.411207055068)
- First digit in product of an array of numbers(0.411207055068)
- Number of ways to calculate a target number using only array elements(0.366529477546)
- Finding LCM of more than two (or array) numbers without using GCD(0.356300429333)
- Find the two numbers with odd occurrences in an unsorted array(0.356300429333)
- Find the nearest smaller numbers on left side in an array(0.356300429333)
- Find original array from encrypted array (An array of sums of other elements)(0.327966201641)

- Removing a number from array to make it Geometric Progression(0.318784021754)
- Minimum sum of two numbers formed from digits of an array(0.318784021754)

Serialize and Deserialize BST(449)

- Serialize and Deserialize an N-ary Tree(0.411207055068)
- Serialize and Deserialize a Binary Tree(0.411207055068)
- Serialization and Deserialization in Java with Example(0.411207055068)
- Two nodes of a BST are swapped, correct the BST(0.285306190981)
- Convert a normal BST to Balanced BST(0.285306190981)
- K'th Largest Element in BST when modification to BST is not allowed(0.241299136472)
- Find k-th smallest element in BST (Order Statistics in BST)(0.241299136472)
- Floor and Ceil from a BST(0.201993092498)
- Sorted Array to Balanced BST(0.17077611319)
- Second largest element in BST(0.17077611319)

Delete Node in a BST(450)

- Two nodes of a BST are swapped, correct the BST(0.502929265114)
- Delete N nodes after M nodes of a linked list(0.418906716157)
- Maximum element between two nodes of BST(0.411207055068)
- Given only a pointer/reference to a node to be deleted in a singly linked list, how do you delete it?(0.366529477546)
- Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?(0.366529477546)
- Delete nodes which have a greater value on right side(0.356300429333)

- Delete alternate nodes of a Linked List(0.356300429333)
- Delete a node in a Doubly Linked List(0.356300429333)
- Linked List | Set 3 (Deleting a node)(0.318784021754)
- Delete a Linked List node at a given position(0.318784021754)

Sort Characters By Frequency(451)

- Sort elements by frequency | Set 2(0.356300429333)
- Sort elements by frequency | Set 1(0.356300429333)
- Check if frequency of all characters can become same by one removal(0.356300429333)
- Sort a nearly sorted (or K sorted) array(0.348993907955)
- Count number of occurrences (or frequency) in a sorted array(0.318784021754)
- Tag Sort (To get both sorted and original)(0.318784021754)
- Sort an array when two halves are sorted(0.318784021754)
- Odd-Even Sort / Brick Sort(0.318784021754)
- Find the first non-repeating character from a stream of characters(0.318784021754)
- Print Kth character in sorted concatenated substrings of a string(0.291069102382)

Minimum Number of Arrows to Burst Balloons(452)

- Find the minimum distance between two numbers(0.356300429333)
- Find a number in minimum steps(0.356300429333)
- Maximum and minimum of an array using minimum number of comparisons(0.296672366897)
- Find minimum number to be divided to make a number a perfect square(0.296672366897)

- Count minimum number of subsets (or subsequences) with consecutive numbers(0.296672366897)
- Allocate minimum number of pages(0.291219418564)
- Minimum number of squares whose sum equals to given number n(0.276274998459)
- Minimum number of operation required to convert number x into y(0.276274998459)
- Convert a number m to n using minimum number of given operations(0.259578477611)
- Paper Cut into Minimum Number of Squares(0.252334201434)

Minimum Moves to Equal Array Elements(453)

- Make all array elements equal with minimum cost(0.709297266606)
- For each element in 1st array count elements less than or equal to it in 2nd array(0.580212787257)
- Minimum sum of two elements from two arrays such that indexes are not same(0.51014901931)
- Find the minimum element in a sorted and rotated array(0.51014901931)
- Remove minimum elements from array such that no three consecutive element are either increasing or decreasing(0.48097310796)
- Minimum flips in two binary arrays so that their XOR is equal to another array(0.48097310796)
- Sum of minimum absolute difference of each array element(0.450175502327)
- Recursive Programs to find Minimum and Maximum elements of array(0.450175502327)
- Recursive Programs to find Minimum and Maximum elements of array(0.450175502327)
- Minimum delete operations to make all elements of array same(0.450175502327)

4Sum II(454)

- Flipkart Interview | Set 7 (For SDE II)(0.175786078393)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.161713780663)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.161713780663)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.161713780663)
- Amazon Interview experience | Set 326 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 348 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 313 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 312 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 163 (For SDE II)(0.161713780663)
- GATE | GATE 2017 MOCK II | Question 9(0.141430567926)

Assign Cookies(455)

- Self assignment check in assignment operator(0.368023208756)
- Is assignment operator inherited?(0.260555671056)
- How cookies are used in a website?(0.260555671056)
- Channel Assignment Problem(0.260555671056)
- Assign Mice to Holes(0.260555671056)
- When should we write our own assignment operator in C++?(0.220288150562)
- Default Assignment Operator and References(0.220288150562)
- Assigning an integer to float and comparison in C/C++(0.194314340169)
- Explicitly assigning port number to client in Socket(0.175786078393)
- Copy constructor vs assignment operator in C++(0.175786078393)

132 Pattern(456)

- Iterator Pattern(0.336096927276)
- Command Pattern(0.336096927276)
- Adapter Pattern(0.336096927276)
- Searching for Patterns | Set 1 (Naive Pattern Searching)(0.274611786436)
- Wildcard Pattern Matching(0.260555671056)
- Searching for Patterns | Set 4 (A Naive Pattern Searching Question)(0.260555671056)
- Flyweight Design Pattern(0.260555671056)
- Find orientation of a pattern in a matrix(0.260555671056)
- Singleton Design Pattern | Introduction(0.220288150562)
- Singleton Design Pattern | Implementation(0.220288150562)

Repeated Substring Pattern(459)

- Longest repeating and non-overlapping substring(0.411207055068)
- Length of the longest substring without repeating characters(0.356300429333)
- Suffix Tree Application 3 – Longest Repeated Substring(0.269517613246)
- Iterator Pattern(0.260555671056)
- Command Pattern(0.260555671056)
- Adapter Pattern(0.260555671056)
- Minimum steps to delete a string after repeated deletion of palindrome substrings(0.237739238575)
- Find if a given string can be represented from a substring by iterating the substring “n” times(0.225764846003)
- Searching for Patterns | Set 1 (Naive Pattern Searching)(0.212889950749)

- Wildcard Pattern Matching(0.201993092498)

LFU Cache(460)

- Cache Memory(0.336096927276)
- Implement LRU Cache(0.260555671056)
- Performance of loops (A caching question)(0.220288150562)
- What's difference between CPU Cache and TLB?(0.194314340169)
- Initializing and Cache Mechanism in Linux Kernel(0.194314340169)
- How to Implement Reverse DNS Look Up Cache?(0.194314340169)
- How to Implement Forward DNS Look Up Cache?(0.194314340169)
- Cache Organization | Set 1 (Introduction)(0.194314340169)

Hamming Distance(461)

- Hamming Distance between two strings(0.709297266606)
- Find the minimum distance between two numbers(0.260555671056)
- Find Shortest distance from a guard in a Bank(0.220288150562)
- Check if edit distance between two strings is one(0.220288150562)
- Print nodes at k distance from root(0.194314340169)
- Placements | QA | Trigonometry & Height and Distances(0.194314340169)
- Placements | QA | Time Speed Distance(0.194314340169)
- Minimum distance to travel to cover all intervals(0.194314340169)
- Maximum distance between two occurrences of same element in array(0.194314340169)
- Find distance between two given keys of a Binary Tree(0.194314340169)

Minimum Moves to Equal Array Elements II(462)

- Make all array elements equal with minimum cost(0.580332984677)
- For each element in 1st array count elements less than or equal to it in 2nd array(0.49089112271)
- Minimum sum of two elements from two arrays such that indexes are not same(0.431613418971)
- Find the minimum element in a sorted and rotated array(0.431613418971)
- Remove minimum elements from array such that no three consecutive element are either increasing or decreasing(0.406929033874)
- Minimum flips in two binary arrays so that their XOR is equal to another array(0.406929033874)
- Sum of minimum absolute difference of each array element(0.380872608476)
- Recursive Programs to find Minimum and Maximum elements of array(0.380872608476)
- Recursive Programs to find Minimum and Maximum elements of array(0.380872608476)
- Minimum delete operations to make all elements of array same(0.380872608476)

Island Perimeter(463)

- Count number of islands where every island is row-wise and column-wise separated(0.291219418564)
- Island of Isolation in Java(0.260555671056)
- The Blue – eyed Island puzzle(0.194314340169)
- The Blue – eyed Island puzzle(0.194314340169)
- Find the number of islands | Set 1 (Using DFS)(0.175786078393)
- Find perimeter of shapes formed with 1s in binary matrix(0.175786078393)
- Find the number of Islands | Set 2 (Using Disjoint Set)(0.141430567926)

Can I Win(464)

- Geek on the Top – Aditya Gupta| Participating alone increases your level, no matter you win or not(0.230767929611)

Optimal Account Balancing(465)

- Query Optimization(0.260555671056)
- Sorted Array to Balanced BST(0.17077611319)
- Print all combinations of balanced parentheses(0.17077611319)
- Optimization Tips for Python Code(0.17077611319)
- Check for balanced parentheses in an expression(0.17077611319)
- A Space Optimized Solution of LCS(0.17077611319)
- Sorted Linked List to Balanced BST(0.150640184987)
- Optimization Techniques | Set 2 (swapping)(0.150640184987)
- Optimization Techniques | Set 1 (Modulus)(0.150640184987)
- Merge Two Balanced Binary Search Trees(0.150640184987)

Count The Repetitions(466)

- Counting Sort(0.336096927276)
- Combinations with repetitions(0.336096927276)
- Count substrings with same first and last characters(0.260555671056)
- Count of parallelograms in a plane(0.260555671056)
- Count numbers with same first and last digits(0.260555671056)
- Count all increasing subsequences(0.260555671056)
- Count Divisors of Factorial(0.260555671056)

- Count Distinct Subsequences(0.260555671056)
- Print all permutations with repetition of characters(0.220288150562)
- Find Surpasser Count of each element in array(0.220288150562)

Unique Substrings in Wraparound String(467)

- Find the longest substring with k unique characters in a given string(0.407352604289)
- Number of even substrings in a string of digits(0.336096927276)
- Count All Palindrome Sub-Strings in a String(0.336096927276)
- Find if a given string can be represented from a substring by iterating the substring "n" times(0.318849541433)
- Searching characters and substring in a String in Java(0.291219418564)
- Program to print all substrings of a given string(0.291219418564)
- Find all distinct palindromic sub-strings of a given string(0.291219418564)
- Determine if a string has all Unique Characters(0.291219418564)
- Print substring of a given string without using any string function and loop in C(0.283428955249)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.277280735105)

Validate IP Address(468)

- Program to validate an IP address(0.776514530475)
- IP Addressing | Classless Addressing(0.569707709055)
- IP Addressing | Introduction and Classful Addressing(0.502929265114)
- Java program to find IP address of your computer(0.356300429333)
- Computer Networks | IP Addressing | Question 8(0.318784021754)

- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 8(0.318784021754)
- Computer Networks | IP Addressing | Question 7(0.318784021754)
- Computer Networks | IP Addressing | Question 6(0.318784021754)

Convex Polygon(469)

- Tangents between two Convex Polygons(0.709297266606)
- Dynamic Convex hull | Adding Points to an Existing Convex Hull(0.260555671056)
- Quickhull Algorithm for Convex Hull(0.220288150562)
- Polygon Clipping | Sutherland–Hodgman Algorithm(0.220288150562)
- Minimum Cost Polygon Triangulation(0.220288150562)
- Deleting points from Convex Hull(0.220288150562)
- Scan-line Polygon filling using OPENGL in C(0.194314340169)
- Convex Hull | Set 2 (Graham Scan)(0.175786078393)
- Convex Hull (Simple Divide and Conquer Algorithm)(0.175786078393)
- Area of a polygon with given n ordered vertices(0.175786078393)

Encode String with Shortest Length(471)

- Run Length Encoding(0.411207055068)
- Length Of Last Word in a String(0.411207055068)
- Sort an array of strings according to string lengths(0.372055731454)
- All possible strings of any length that can be formed from a given string(0.342390186113)

- How to find length of a string without string.h and loop in C?(0.291219418564)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.277280735105)
- Print string of odd length in 'X' format(0.260555671056)
- Count ways to increase LCS length of two strings by one(0.260555671056)
- Convert to a string that is repetition of a substring of k length(0.260555671056)
- Check length of a string is equal to the number appended at its last(0.260555671056)

Concatenated Words(472)

- Word formation using concatenation of two dictionary words(0.641764556549)
- Word Ladder (Length of shortest chain to reach a target word)(0.291219418564)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Length Of Last Word in a String(0.260555671056)
- Addition and Concatenation in Java(0.260555671056)
- Reverse words in a given string(0.220288150562)
- Group words with same set of characters(0.220288150562)
- Find the k most frequent words from a file(0.220288150562)

Matchsticks to Square(473)

- Magic Square(0.336096927276)
- Latin Square(0.336096927276)
- Square root of an integer(0.260555671056)

- Direction at last square block(0.260555671056)
- Program to find number of squares in a chessboard(0.220288150562)
- Nth Square free number(0.220288150562)
- Maximum and Minimum in a square matrix.(0.220288150562)
- Count number of squares in a rectangle(0.220288150562)
- Babylonian method for square root(0.220288150562)
- Square root of a number using log(0.194314340169)

Ones and Zeroes(474)

- Find the number of zeroes(0.579738671538)
- Move all zeroes to end of array(0.449436416524)
- Find all triplets with zero sum(0.449436416524)
- Two elements whose sum is closest to zero(0.379978361591)
- Count Pairs Of Consecutive Zeros(0.379978361591)
- Total coverage of all zeros in a binary matrix(0.335175743328)
- Remove Trailing Zeros From string in C++(0.335175743328)
- Remove Trailing Zeros From String in Java(0.335175743328)
- Find if there is a triplet in a Balanced BST that adds to zero(0.335175743328)
- Count trailing zeroes in factorial of a number(0.335175743328)

Heaters(475)

Number Complement(476)

- 1's and 2's complement of a Binary Number(0.502328778226)
- Smallest number divisible by first n numbers(0.368023208756)

- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Number of perfect squares between two given numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)
- Find the missing number in a string of numbers with no separator(0.368023208756)
- Find the Number Occurring Odd Number of Times(0.368023208756)

Total Hamming Distance(477)

- Hamming Distance between two strings(0.503102612415)
- Optimum location of point to minimize total distance(0.318784021754)
- Puzzle 3 | (Calculate total distance travelled by bee)(0.291069102382)
- Find the minimum distance between two numbers(0.201993092498)
- Find Shortest distance from a guard in a Bank(0.17077611319)
- Count of total anagram substrings(0.17077611319)
- Check if edit distance between two strings is one(0.17077611319)
- Total number of Spanning Trees in a Graph(0.150640184987)
- Total coverage of all zeros in a binary matrix(0.150640184987)
- Print nodes at k distance from root(0.150640184987)

Sliding Window Median(480)

- Window Sliding Technique(0.503102612415)

- Sliding Window Protocol | Set 2 (Receiver Side)(0.318784021754)
- Sliding Window Protocol | Set 1 (Sender Side)(0.318784021754)
- Sliding Window Maximum (Maximum of all subarrays of size k)(0.252138706945)
- Median of two sorted arrays(0.201993092498)
- Windows 10 –Feel the Difference(0.17077611319)
- Smallest window that contains all characters of string itself(0.150640184987)
- Median of two sorted arrays of different sizes(0.150640184987)
- Find median of BST in O(n) time and O(1) space(0.150640184987)
- First negative integer in every window of size k(0.136276341439)

Magical String(481)

- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)
- Sort an array of strings according to string lengths(0.336096927276)
- Search in an array of strings where non-empty strings are sorted(0.336096927276)
- Remove characters from the first string which are present in the second string(0.336096927276)
- Magic Square(0.336096927276)
- Check if given string can be split into four distinct strings(0.336096927276)
- Sort a string according to the order defined by another string(0.311257467527)

License Key Formatting(482)

- How to implement decrease key or change key in Binary Search Tree?(0.225764846003)
- JSON Formatting in Python(0.201993092498)
- Formatted output in Java(0.201993092498)
- Convert a BST to a Binary Tree such that sum of all greater keys is added to every key(0.201993092498)
- AVL with duplicate keys(0.201993092498)
- String Formatting in Python using %(0.17077611319)
- Queries on substring palindrome formation(0.17077611319)
- Find next right node of a given key(0.17077611319)
- Using a variable as format specifier in C(0.150640184987)
- Print BST keys in the given range(0.150640184987)

Smallest Good Base(483)

- Find the smallest and second smallest elements in an array(0.285306190981)
- Maximum sum of smallest and second smallest in an array(0.260555671056)
- Smallest Palindrome after replacement(0.201993092498)
- Find the smallest missing number(0.201993092498)
- DFA based division(0.201993092498)
- Smallest of three integers without comparison operators(0.17077611319)
- Smallest Subarray with given GCD(0.17077611319)
- Smallest Difference Triplet from Three arrays(0.17077611319)
- Pandigital number in a given base(0.17077611319)
- How to write a good SRS for your Project(0.17077611319)

Find Permutation(484)

- Permutation Coefficient(0.579738671538)
- Permutation and Combination in Python(0.449436416524)
- Lexicographically next permutation in C++(0.449436416524)
- K difference permutation(0.449436416524)
- How to find Lexicographically previous permutation?(0.449436416524)
- Generate all binary permutations such that there are more or equal 1's than 0's before every point in all permutations(0.449436416524)
- Check if two arrays are permutations of each other(0.449436416524)
- BogoSort or Permutation Sort(0.449436416524)
- Print all permutations with repetition of characters(0.379978361591)
- Print all palindrome permutations of a string(0.379978361591)

Max Consecutive Ones(485)

- Longest Consecutive Subsequence(0.260555671056)
- Merge two binary Max Heaps(0.220288150562)
- Maximum subsequence sum such that no three are consecutive(0.220288150562)
- Max Flow Problem Introduction(0.220288150562)
- Delete consecutive same words in a sequence(0.220288150562)
- Count strings with consecutive 1's(0.220288150562)
- Count Pairs Of Consecutive Zeros(0.220288150562)
- Print consecutive characters together in a line(0.194314340169)
- Maximum consecutive repeating character in string(0.194314340169)
- Longest consecutive sequence in Binary tree(0.194314340169)

Predict the Winner(486)

- Branch prediction macros in GCC(0.220288150562)
- Tournament Tree (Winner Tree) and Binary Heap(0.150556969602)

Max Consecutive Ones II(487)

- Longest Consecutive Subsequence(0.201993092498)
- Merge two binary Max Heaps(0.17077611319)
- Maximum subsequence sum such that no three are consecutive(0.17077611319)
- Max Flow Problem Introduction(0.17077611319)
- Delete consecutive same words in a sequence(0.17077611319)
- Count strings with consecutive 1's(0.17077611319)
- Count Pairs Of Consecutive Zeros(0.17077611319)
- Print consecutive characters together in a line(0.150640184987)
- Maximum consecutive repeating character in string(0.150640184987)
- Longest consecutive sequence in Binary tree(0.150640184987)

Zuma Game(488)

- Combinatorial Game Theory | Set 2 (Game of Nim)(0.311257467527)
- Implementation of Tic-Tac-Toe game(0.260555671056)
- Implementation of Minesweeper Game(0.260555671056)
- Hangman Game in Python(0.260555671056)
- A Number Link Game(0.260555671056)
- The prisoner's dilemma in Game theory(0.220288150562)
- Puzzle 73 | The Card Game(0.220288150562)

- Puzzle 69 |The Number Game(0.220288150562)
- Project Idea | (A Game of Anagrams)(0.220288150562)
- Program for Conway's Game Of Life(0.220288150562)

The Maze(490)

- Shortest path in a Binary Maze(0.379978361591)
- Backtracking | Set 2 (Rat in a Maze)(0.335175743328)
- Count number of ways to reach destination in a Maze(0.30321606445)
- Find paths from corner cell to middle cell in maze(0.25969799324)

Increasing Subsequences(491)

- Count all increasing subsequences(0.709297266606)
- Printing Maximum Sum Increasing Subsequence(0.502328778226)
- Find the Increasing subsequence of length three with maximum product(0.502328778226)
- Maximum product of an increasing subsequence of size 3(0.449436416524)
- Longest Common Increasing Subsequence (LCS + LIS)(0.449436416524)
- Minimum number of elements which are not part of Increasing or decreasing subsequence in array(0.410362644952)
- Dynamic Programming | Set 3 (Longest Increasing Subsequence)(0.410362644952)
- Construction of Longest Increasing Subsequence using Dynamic Programming(0.410362644952)
- Dynamic Programming | Set 14 (Maximum Sum Increasing Subsequence)(0.379978361591)
- Longest Increasing Subsequence Size (N log N)(0.355476777955)

Construct the Rectangle(492)

- Find if two rectangles overlap(0.336096927276)
- Count number of squares in a rectangle(0.220288150562)
- Construct tree from ancestor matrix(0.220288150562)
- Check if four segments form a rectangle(0.220288150562)
- Count possible ways to construct buildings(0.194314340169)
- Construct a Binary Tree from Postorder and Inorder(0.194314340169)
- Ukkonen's Suffix Tree Construction – Part 6(0.175786078393)
- Ukkonen's Suffix Tree Construction – Part 5(0.175786078393)
- Ukkonen's Suffix Tree Construction – Part 4(0.175786078393)
- Ukkonen's Suffix Tree Construction – Part 3(0.175786078393)

Reverse Pairs(493)

- Given an array of pairs, find all symmetric pairs in it(0.368023208756)
- Reversible numbers(0.336096927276)
- Find pairs with given sum such that elements of pair are in different rows(0.311257467527)
- Pair with given product | Set 1 (Find if any pair exists)(0.291219418564)
- Find pairs with given sum such that pair elements lie in different BSTs(0.291219418564)
- Reverse and Add Function(0.260555671056)
- Perfect reversible string(0.260555671056)
- Pairs of Amicable Numbers(0.260555671056)
- Pair Class in Java(0.260555671056)
- Friends Pairing Problem(0.260555671056)

Target Sum(494)

- Find sum of sum of all sub-sequences(0.474330706497)
- Sum of all Subarrays(0.336096927276)
- Find maximum sum possible equal sum of three stacks(0.336096927276)
- Print all possible sums of consecutive numbers with sum N(0.311257467527)
- Perfect Sum Problem (Print all subsets with given sum)(0.311257467527)
- Print all n-digit numbers whose sum of digits equals to given sum(0.291219418564)
- Finding sum of digits of a number until sum becomes single digit(0.274611786436)
- Sum of two large numbers(0.260555671056)
- Sum of subset differences(0.260555671056)
- Sum of dependencies in a graph(0.260555671056)

Teemo Attacking(495)

- Understanding ReDoS Attack(0.260555671056)
- Path Traversal Attack and Prevention(0.220288150562)
- Buffer Overflow Attack with Example(0.220288150562)
- Mitigation of SQL Injection Attack using Prepared Statements (Parameterized Queries)(0.141430567926)

Next Greater Element I(496)

- Next Greater Element(1.0)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.709052873586)
- Find all elements in array which have at-least two greater elements(0.709052873586)

- Replace every element with the least greater element on its right(0.641764556549)
- Maximum difference between frequency of two elements such that element having greater frequency is also greater(0.634808797178)
- Count of subarrays whose maximum element is greater than k(0.449436416524)
- Rearrange an array such that every odd indexed element is greater than it previous(0.379978361591)
- Noble integers in an array (count of greater elements is equal to value)(0.379978361591)
- Third largest element in an array of distinct elements(0.368023208756)
- Find the two non-repeating elements in an array of repeating elements(0.368023208756)

Diagonal Traverse(498)

- Zigzag (or diagonal) traversal of Matrix(0.579738671538)
- Diagonal Traversal of Binary Tree(0.579738671538)
- Delete an element from array (Using two traversals and one traversal)(0.336096927276)
- Print Postorder traversal from given Inorder and Preorder traversals(0.311257467527)
- Morris traversal for Preorder(0.260555671056)
- Iterative Preorder Traversal(0.260555671056)
- Find difference between sums of two diagonals(0.260555671056)
- Applications of Breadth First Traversal(0.260555671056)
- Reverse Level Order Traversal(0.220288150562)
- Print matrix in diagonal pattern(0.220288150562)

The Maze III(499)

- Shortest path in a Binary Maze(0.220288150562)
- Backtracking | Set 2 (Rat in a Maze)(0.194314340169)
- Count number of ways to reach destination in a Maze(0.175786078393)
- Find paths from corner cell to middle cell in maze(0.150556969602)

Keyboard Row(500)

- Find all permuted rows of a given row in a matrix(0.368023208756)
- How to print duplicate rows in a table?(0.220288150562)
- Find the row with maximum number of 1s(0.220288150562)
- Count all sorted rows in a matrix(0.220288150562)
- Maximum path sum that starting with any cell of 0-th row and ending with any cell of (N-1)-th row(0.199939658353)
- Puzzle 40 | (Find missing Row in Excel)(0.194314340169)
- Mouse and keyboard automation using Python(0.194314340169)
- Find distinct elements common to all rows of a matrix(0.194314340169)
- Common elements in all rows of a given matrix(0.194314340169)
- Check if all rows of a matrix are circular rotations of each other(0.194314340169)

Find Mode in Binary Search Tree(501)

- Binary Tree to Binary Search Tree Conversion(0.676628251794)
- Binary Search(0.579738671538)
- Minimum swap required to convert binary tree to binary search tree(0.545253597965)
- Binary Search Tree | Set 1 (Search and Insertion)(0.519280018803)

- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Treap (A Randomized Binary Search Tree)(0.51014901931)
- Threaded Binary Search Tree | Deletion(0.51014901931)
- Merge Two Balanced Binary Search Trees(0.51014901931)
- Inorder Successor in Binary Search Tree(0.51014901931)
- How to handle duplicates in Binary Search Tree?(0.51014901931)

IPO(502)

Next Greater Element II(503)

- Next Greater Element(0.709297266606)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.502929265114)
- Find all elements in array which have at-least two greater elements(0.502929265114)
- Replace every element with the least greater element on its right(0.455201845765)
- Maximum difference between frequency of two elements such that element having greater frequency is also greater(0.450268144656)
- Count of subarrays whose maximum element is greater than k(0.318784021754)
- Third largest element in an array of distinct elements(0.285306190981)
- Find the two non-repeating elements in an array of repeating elements(0.285306190981)
- Find elements larger than half of the elements in an array(0.285306190981)
- Elements before which no element is bigger in array(0.285306190981)

Base 7(504)

- G-Fact 7(0.336096927276)

- Operating Systems | Set 7(0.260555671056)
- Multiples of 3 or 7(0.260555671056)
- DFA based division(0.260555671056)
- Check divisibility by 7(0.260555671056)
- Remainder with 7 for large numbers(0.220288150562)
- Python-Quizzes | Miscellaneous | Question 7(0.220288150562)
- Python | Functions | Question 7(0.220288150562)
- Pandigital number in a given base(0.220288150562)
- Oracle Interview | Set 7(0.220288150562)

The Maze II(505)

- Shortest path in a Binary Maze(0.220288150562)
- Backtracking | Set 2 (Rat in a Maze)(0.194314340169)
- Flipkart Interview | Set 7 (For SDE II)(0.175786078393)
- Count number of ways to reach destination in a Maze(0.175786078393)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.161713780663)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.161713780663)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.161713780663)
- Amazon Interview experience | Set 326 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 348 (For SDE II)(0.161713780663)
- Amazon Interview Experience | Set 313 (For SDE II)(0.161713780663)

Relative Ranks(506)

- Program for Rank of Matrix(0.260555671056)

- Lexicographic rank of a string(0.260555671056)
- How ranking in Google Search Works !(0.220288150562)
- Print all root to leaf paths with there relative positions(0.175786078393)
- Union-Find Algorithm | Set 2 (Union By Rank and Path Compression)(0.150556969602)

Perfect Number(507)

- Perfect Number(1.0)
- Number of perfect squares between two given numbers(0.709052873586)
- Find minimum number to be divided to make a number a perfect square(0.590594008858)
- Count all perfect divisors of a number(0.579738671538)
- Smallest number divisible by first n numbers(0.368023208756)
- Number with maximum number of prime factors(0.368023208756)
- Number of subtrees having odd count of even numbers(0.368023208756)
- Next higher number with same number of set bits(0.368023208756)
- How to check if a given number is Fibonacci number?(0.368023208756)
- Finding number of digits in n'th Fibonacci number(0.368023208756)

Most Frequent Subtree Sum(508)

- Find sum of sum of all sub-sequences(0.36771998047)
- Subtree with given sum in a Binary Tree(0.356300429333)
- Delete Edge to minimize subtree sum difference(0.318784021754)
- Sum of all Subarrays(0.260555671056)
- Find maximum sum possible equal sum of three stacks(0.260555671056)

- Find largest subtree having identical left and right subtrees(0.260555671056)
- Print all possible sums of consecutive numbers with sum N(0.241299136472)
- Perfect Sum Problem (Print all subsets with given sum)(0.241299136472)
- Print all n-digit numbers whose sum of digits equals to given sum(0.225764846003)
- Change a Binary Tree so that every node stores sum of all nodes in left subtree(0.215070325706)

Find Bottom Left Tree Value(513)

- Print Left View of a Binary Tree(0.356300429333)
- Find the node with minimum value in a Binary Search Tree(0.318784021754)
- Find sum of all left leaves in a given Binary Tree(0.318784021754)
- Deepest left leaf node in a binary tree(0.318784021754)
- Create loops of even and odd values in a binary tree(0.318784021754)
- Convert a given tree to its Sum Tree(0.285306190981)
- Binary Indexed Tree or Fenwick Tree(0.285306190981)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.260555671056)
- Tree Sort(0.260555671056)
- Tournament Tree (Winner Tree) and Binary Heap(0.260555671056)

Freedom Trail(514)

- Remove Trailing Zeros From string in C++(0.194314340169)
- Remove Trailing Zeros From String in Java(0.194314340169)
- Count trailing zeroes in factorial of a number(0.194314340169)
- Smallest number with at least n trailing zeroes in factorial(0.175786078393)

- Trim (Remove leading and trailing spaces) a string in Java(0.161713780663)
- Count trailing zero bits using lookup table(0.161713780663)

Find Largest Value in Each Tree Row(515)

- Largest BST in a Binary Tree | Set 2(0.260555671056)
- Find the node with minimum value in a Binary Search Tree(0.260555671056)
- Create loops of even and odd values in a binary tree(0.260555671056)
- Find all permuted rows of a given row in a matrix(0.241213606675)
- Convert a given tree to its Sum Tree(0.241213606675)
- Binary Indexed Tree or Fenwick Tree(0.241213606675)
- Program to find the largest and smallest ASCII valued characters in a string(0.237903094633)
- Find the largest BST subtree in a given Binary Tree | Set 1(0.220288150562)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.220288150562)
- Tree Sort(0.220288150562)

Longest Palindromic Subsequence(516)

- Dynamic Programming | Set 12 (Longest Palindromic Subsequence)(0.524591090446)
- Longest alternating subsequence(0.503102612415)
- Longest Zig-Zag Subsequence(0.503102612415)
- Longest Repeating Subsequence(0.503102612415)
- Longest Consecutive Subsequence(0.503102612415)
- Printing Longest Common Subsequence(0.411207055068)
- Printing Longest Bitonic Subsequence(0.411207055068)

- Longest subsequence such that difference between adjacents is one(0.411207055068)
- Longest common subsequence with permutations allowed(0.356300429333)
- Longest Palindromic Substring | Set 2(0.356300429333)

Super Washing Machines(517)

- Turing Machine(0.260555671056)
- Super Prime(0.260555671056)
- Machine Instructions(0.260555671056)
- Super Keyword in Java(0.201993092498)
- Mealy and Moore Machines(0.201993092498)
- Getting started with Machine Learning(0.201993092498)
- Demystifying Machine Learning(0.201993092498)
- Machine Learning – Applications(0.17077611319)
- Check if all people can vote on two machines(0.17077611319)
- C++ program to find Machine Epsilon(0.17077611319)

Detect Capital(520)

- Deadlock Detection And Recovery(0.260555671056)
- Real-Time Edge Detection using OpenCV in Python | Canny edge detection method(0.237903094633)
- Python Program to detect the edges of an image using OpenCV | Sobel edge detection method(0.22858816138)
- Detect if two integers have opposite signs(0.220288150562)
- Detect cycle in an undirected graph(0.220288150562)

- Detect Cycle in a Directed Graph(0.220288150562)
- Computer Networks | Error Detection(0.220288150562)
- Tower Research Capital Interview Experience(0.194314340169)
- Put spaces between words starting with capital letters(0.194314340169)
- Project Idea | (Robust Pedestrian detection)(0.194314340169)

Longest Uncommon Subsequence I(521)

- Shortest Uncommon Subsequence(0.503102612415)
- Longest alternating subsequence(0.503102612415)
- Longest Zig-Zag Subsequence(0.503102612415)
- Longest Repeating Subsequence(0.503102612415)
- Longest Consecutive Subsequence(0.503102612415)
- Printing Longest Common Subsequence(0.411207055068)
- Printing Longest Bitonic Subsequence(0.411207055068)
- Longest subsequence such that difference between adjacents is one(0.411207055068)
- Longest common subsequence with permutations allowed(0.356300429333)
- LCS (Longest Common Subsequence) of three strings(0.356300429333)

Longest Uncommon Subsequence II(522)

- Shortest Uncommon Subsequence(0.411207055068)
- Longest alternating subsequence(0.411207055068)
- Longest Zig-Zag Subsequence(0.411207055068)
- Longest Repeating Subsequence(0.411207055068)

- Longest Consecutive Subsequence(0.411207055068)
- Printing Longest Common Subsequence(0.336096927276)
- Printing Longest Bitonic Subsequence(0.336096927276)
- Longest subsequence such that difference between adjacents is one(0.336096927276)
- Longest common subsequence with permutations allowed(0.291219418564)
- LCS (Longest Common Subsequence) of three strings(0.291219418564)

Continuous Subarray Sum(523)

- Sum of all Subarrays(0.709297266606)
- Find number of subarrays with even sum(0.503102612415)
- Find if there is a subarray with 0 sum(0.503102612415)
- Print all subarrays with 0 sum(0.411207055068)
- Maximum circular subarray sum(0.411207055068)
- Largest Sum Contiguous Subarray(0.411207055068)
- Find the largest subarray with 0 sum(0.411207055068)
- Find sum of sum of all sub-sequences(0.36771998047)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.366529477546)
- Split an array into two equal Sum subarrays(0.356300429333)

Longest Word in Dictionary through Deleting(524)

- Find largest word in dictionary by deleting some characters of given string(0.407352604289)
- Word formation using concatenation of two dictionary words(0.372055731454)

- Delete consecutive same words in a sequence(0.336096927276)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.318849541433)
- Trie | (Delete)(0.220288150562)
- Word Ladder (Length of shortest chain to reach a target word)(0.19087406613)
- Minimum steps to delete a string after repeated deletion of palindrome substrings(0.19087406613)
- C program to Replace a word in a text by another given word(0.19087406613)
- Given only a pointer/reference to a node to be deleted in a singly linked list, how do you delete it?(0.179988918812)
- Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?(0.179988918812)

Contiguous Array(525)

- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Count the number of ways to divide an array into three contiguous parts having equal sum(0.379978361591)
- Find Duplicates of array using bit array(0.368023208756)
- Pointer to an Array | Array Pointer(0.336096927276)
- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)
- Arrays in Java(0.336096927276)
- Arrays in Java(0.336096927276)

Beautiful Arrangement(526)

- Program to print an array in Pendulum Arrangement(0.194314340169)
- Maximum height when coins are arranged in a triangle(0.194314340169)
- Minimum number of swaps required for arranging pairs adjacent to each other(0.161713780663)
- Locking and Unlocking of Resources arranged in the form of n-ary Tree(0.161713780663)
- Biggest number by arranging numbers in certain order(0.150556969602)
- Arrange given numbers to form the biggest number(0.150556969602)
- Ways to arrange Balls such that adjacent balls are of different types(0.141430567926)
- Check if an array can be Arranged in Left or Right Positioned Array(0.141430567926)

Word Abbreviation(527)

- Word formation using concatenation of two dictionary words(0.336096927276)
- Word Ladder (Length of shortest chain to reach a target word)(0.291219418564)
- Longest Common Prefix | Set 1 (Word by Word Matching)(0.291219418564)
- C program to Replace a word in a text by another given word(0.291219418564)
- Testimonials – Words that keep us going(0.260555671056)
- Length Of Last Word in a String(0.260555671056)
- Reverse words in a given string(0.220288150562)
- Group words with same set of characters(0.220288150562)
- Find the k most frequent words from a file(0.220288150562)
- Find all occurrences of a given word in a matrix(0.220288150562)

Minesweeper(529)

- Implementation of Minesweeper Game(0.449436416524)

Minimum Absolute Difference in BST(530)

- Sum of minimum absolute difference of each array element(0.450175502327)
- Minimum sum of absolute difference of pairs of two arrays(0.450175502327)
- Find minimum difference between any two elements(0.336096927276)
- Find the minimum difference between Shifted tables of two numbers(0.291219418564)
- k-th smallest absolute difference of two elements in an array(0.260555671056)
- Sum of absolute differences of all pairs in a given array(0.260555671056)
- Minimum difference between max and min of all K-size subsets(0.260555671056)
- Maximum absolute difference between sum of two contiguous subarrays(0.260555671056)
- Clustering/Partitioning an array such that sum of square differences is minimum(0.260555671056)
- Two nodes of a BST are swapped, correct the BST(0.241213606675)

Lonely Pixel I(531)

- Image Processing In Java | Set 2 (Get and set Pixels)(0.141430567926)
- Image Processing in Java | Set 7 (Creating a random pixel image)(0.12725898701)

K-diff Pairs in an Array(532)

- Given an array of pairs, find all symmetric pairs in it(0.502929265114)
- Find pairs in array whose sums already exist in array(0.455201845765)
- Maximizing Unique Pairs from two arrays(0.411207055068)

- Find the closest pair from two sorted arrays(0.411207055068)
- Find pair with greatest product in array(0.411207055068)
- Sum of product of all pairs of array elements(0.356300429333)
- Sum of Bitwise And of all pairs in a given array(0.356300429333)
- Palindrome pair in an array of words (or strings)(0.356300429333)
- Find the largest pair sum in an unsorted array(0.356300429333)
- Find k pairs with smallest sums in two arrays(0.356300429333)

Lonely Pixel II(533)

- Flipkart Interview | Set 7 (For SDE II)(0.136276341439)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.125366937987)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.125366937987)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.125366937987)
- Amazon Interview experience | Set 326 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 348 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 313 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 312 (For SDE II)(0.125366937987)
- Amazon Interview Experience | Set 163 (For SDE II)(0.125366937987)
- Image Processing In Java | Set 2 (Get and set Pixels)(0.109642586835)

Encode and Decode TinyURL(535)

- Huffman Decoding(0.260555671056)
- Run Length Encoding(0.201993092498)
- Succinct Encoding of Binary Tree(0.17077611319)

- Count Possible Decodings of a given Digit Sequence(0.136276341439)
- Decode a given pattern in two ways (Flipkart Interview Question)(0.125366937987)

Construct Binary Tree from String(536)

- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Construct a Binary Tree from Postorder and Inorder(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)
- If you are given two traversal sequences, can you construct the binary tree?(0.450175502327)
- Construct a Binary Search Tree from given postorder(0.450175502327)
- Construct Ancestor Matrix from a Given Binary Tree(0.450175502327)
- Binary Tree | Set 3 (Types of Binary Tree)(0.439404118785)
- fork() and Binary Tree(0.411207055068)
- Threaded Binary Tree(0.411207055068)
- Foldable Binary Trees(0.411207055068)

Complex Number Multiplication(537)

- N-th multiple in sorted list of multiples of two numbers(0.455201845765)
- Multiplication of two numbers with shift operator(0.411207055068)
- Geometry using Complex Numbers(0.411207055068)
- n'th multiple of a number in Fibonacci Series(0.356300429333)
- Complex numbers in C++ | Set 2(0.356300429333)
- Complex numbers in C++ | Set 1(0.356300429333)
- Multiply a number with 10 without using multiplication operator(0.318784021754)

- Find the smallest binary digit multiple of given number(0.318784021754)
- Complex Numbers in Python | Set 1 (Introduction)(0.318784021754)
- Check if a number is multiple of 5 without using / and % operators(0.318784021754)

Convert BST to Greater Tree(538)

- Transform a BST to greater sum tree(0.51014901931)
- Convert a BST to a Binary Tree such that sum of all greater keys is added to every key(0.449436416524)
- Convert a normal BST to Balanced BST(0.411065370983)
- Convert a given tree to its Sum Tree(0.411065370983)
- In-place Convert BST into a Min-Heap(0.336096927276)
- Convert a tree to forest of even nodes(0.336096927276)
- Convert BST to Min Heap(0.336096927276)
- Write an Efficient Function to Convert a Binary Tree into its Mirror Tree(0.318849541433)
- Convert a given Binary tree to a tree that holds Logical AND property(0.318849541433)
- Convert an arbitrary Binary Tree to a tree that holds Children Sum Property(0.299580052534)

Minimum Time Difference(539)

- Changing One Clock Time to Other Time in Minimum Number of Operations(0.418906716157)
- Find minimum difference between any two elements(0.411207055068)
- Minimum time required to rot all oranges(0.356300429333)

- Find the minimum difference between Shifted tables of two numbers(0.356300429333)
- Sum of minimum absolute difference of each array element(0.318784021754)
- Minimum time to finish tasks without skipping two consecutive(0.318784021754)
- Minimum time required to produce m items(0.318784021754)
- Minimum sum of absolute difference of pairs of two arrays(0.318784021754)
- Minimum difference between max and min of all K-size subsets(0.318784021754)
- Find minimum time to finish all jobs with given constraints(0.318784021754)

Reverse String II(541)

- Perfect reversible string(0.503102612415)
- Reverse words in a given string(0.411207055068)
- Write a program to reverse an array or string(0.356300429333)
- Reverse a string preserving space positions(0.356300429333)
- Print reverse of a string using recursion(0.356300429333)
- Different methods to reverse a string in C/C++(0.356300429333)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Reverse string without using any temporary variable(0.318784021754)
- Reverse a string in Java (5 Different Ways)(0.318784021754)
- Pairs of complete strings in two sets of strings(0.285306190981)

01 Matrix(542)

- Queries in a Matrix(0.336096927276)
- Matrix Introduction(0.336096927276)

- Matrix Exponentiation(0.336096927276)
- Determinant of a Matrix(0.336096927276)
- Circular Matrix (Construct a matrix with numbers 1 to m*n in spiral way)(0.274611786436)
- Saddle point in a matrix(0.260555671056)
- Rotate Matrix Elements(0.260555671056)
- Program for Rank of Matrix(0.260555671056)
- Implementation of a Falling Matrix(0.260555671056)
- Form coils in a matrix(0.260555671056)

Diameter of Binary Tree(543)

- Diameter of a Binary Tree(1.0)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)
- Diameter of an N-ary tree(0.503102612415)

Output Contest Matches(544)

- Wildcard Pattern Matching(0.201993092498)

- Maximum Bipartite Matching(0.201993092498)
- Match Expression where a single special character in pattern can match one or more characters(0.201993092498)
- Formatted output in Java(0.201993092498)
- How to change the output of printf() in main() ?(0.17077611319)
- Basic Input / Output in C++(0.17077611319)
- Template matching using OpenCV in Python(0.150640184987)
- Redirecting System.out.println() output to a file in Java(0.150640184987)
- Python-Quizzes | Output Type | Question 12(0.150640184987)
- Python-Quizzes | Output Type | Question 10(0.150640184987)

Boundary of Binary Tree(545)

- Boundary Traversal of binary tree(0.776514530475)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)
- Diameter of a Binary Tree(0.503102612415)

Remove Boxes(546)

- Remove Invalid Parentheses(0.260555671056)

- Removing punctuations from a given string(0.220288150562)
- Remove spaces from a given string(0.220288150562)
- Remove extra spaces from a string(0.220288150562)
- Remove duplicates from sorted array(0.220288150562)
- Remove all duplicates from a given string(0.220288150562)
- Recursively remove all adjacent duplicates(0.220288150562)
- Program to remove vowels from a String(0.220288150562)
- Length of Longest sub-string that can be removed(0.220288150562)
- How to remove an element from ArrayList in Java?(0.220288150562)

Friend Circles(547)

- program to find area of a circle(0.260555671056)
- Puzzle-61| Cake and my friend(0.260555671056)
- Friends Pairing Problem(0.260555671056)
- Circle and Lattice Points(0.260555671056)
- Puzzle -58 | Friends after ages(0.220288150562)
- Mid-Point Circle Drawing Algorithm(0.220288150562)
- Friend class and function in C++(0.220288150562)
- Find if a point lies inside a Circle(0.220288150562)
- Puzzle 63 | Paper ball and three friends(0.194314340169)
- Puzzle 55 | Geek and his Friend(0.194314340169)

Split Array with Equal Sum(548)

- Split an array into two equal Sum subarrays(0.818180207367)

- Check if there exist two elements in an array whose sum is equal to the sum of rest of the array(0.545253597965)
- Find if array can be divided into two subarrays of equal sum(0.51014901931)
- Equal Sum and XOR(0.411207055068)
- Check if two arrays are equal or not(0.411207055068)
- Find original array from encrypted array (An array of sums of other elements)(0.410888471656)
- Permute two arrays such that sum of every pair is greater or equal to K(0.374807770059)
- Count the number of ways to divide an array into three contiguous parts having equal sum(0.374807770059)
- Find pairs in array whose sums already exist in array(0.372055731454)
- Find maximum sum possible equal sum of three stacks(0.372055731454)

Binary Tree Longest Consecutive Sequence II(549)

- Longest consecutive sequence in Binary tree(0.84664735365)
- Check whether a binary tree is a full binary tree or not(0.402484879511)
- Binary Tree to Binary Search Tree Conversion(0.36771998047)
- Binary Tree | Set 3 (Types of Binary Tree)(0.340643504131)
- Length of the Longest Consecutive 1s in Binary Representation(0.336096927276)
- If you are given two traversal sequences, can you construct the binary tree?(0.336096927276)
- Find longest sequence of 1's in binary representation with one flip(0.336096927276)
- fork() and Binary Tree(0.318784021754)
- Threaded Binary Tree(0.318784021754)

- Longest Consecutive Subsequence(0.318784021754)

Student Attendance Record I(551)

- Student Data Management in C++(0.17077611319)
- [TopTalent.in] 51 Students from BITS get into GSoC, Janani talks about her experience.(0.11671773546)
- Geek on the Top – Nafis Sadique | Seniors should take the responsibility to introduce the junior students to the world of programming(0.0986562151192)

Student Attendance Record II(552)

- Student Data Management in C++(0.144383555277)
- Flipkart Interview | Set 7 (For SDE II)(0.115215543378)
- Microsoft Interview Experience | Set 75 (For SDE II)(0.105992131351)
- Flipkart Interview Experience| Set 38 (For SDE II)(0.105992131351)
- Flipkart Interview Experience | Set 17 (For SDE II)(0.105992131351)
- Amazon Interview experience | Set 326 (For SDE II)(0.105992131351)
- Amazon Interview Experience | Set 348 (For SDE II)(0.105992131351)
- Amazon Interview Experience | Set 313 (For SDE II)(0.105992131351)
- Amazon Interview Experience | Set 312 (For SDE II)(0.105992131351)
- Amazon Interview Experience | Set 163 (For SDE II)(0.105992131351)

Optimal Division(553)

- Query Optimization(0.336096927276)
- Modular Division(0.336096927276)
- Division Operators in Python(0.260555671056)

- DFA based division(0.260555671056)
- Check divisibility by 7(0.260555671056)
- Subset with sum divisible by m(0.220288150562)
- Sub-string Divisibility by 3 Queries(0.220288150562)
- Sub-string Divisibility by 11 Queries(0.220288150562)
- Optimization Tips for Python Code(0.220288150562)
- Largest divisible subset in array(0.220288150562)

Brick Wall(554)

- Magic Bricks Interview Experience(0.220288150562)
- Odd-Even Sort / Brick Sort(0.175786078393)

Split Concatenated Strings(555)

- Check if given string can be split into four distinct strings(0.455201845765)
- Split() String method in Java with examples(0.356300429333)
- Nth character in Concatenated Decimal String(0.356300429333)
- How to split a string in C/C++, Python and Java?(0.356300429333)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- Ropes Data Structure (Fast String Concatenation)(0.318784021754)
- Print Concatenation of Zig-Zag String in ‘n’ Rows(0.318784021754)
- Print Kth character in sorted concatenated substrings of a string(0.291069102382)
- C++ program to concatenate a string given number of times(0.291069102382)
- Pairs of complete strings in two sets of strings(0.285306190981)

Next Greater Element III(556)

- Next Greater Element(0.709297266606)
- Find the element before which all the elements are smaller than it, and after which all are greater(0.502929265114)
- Find all elements in array which have at-least two greater elements(0.502929265114)
- Replace every element with the least greater element on its right(0.455201845765)
- Maximum difference between frequency of two elements such that element having greater frequency is also greater(0.450268144656)
- Count of subarrays whose maximum element is greater than k(0.318784021754)
- Third largest element in an array of distinct elements(0.285306190981)
- Find the two non-repeating elements in an array of repeating elements(0.285306190981)
- Find elements larger than half of the elements in an array(0.285306190981)
- Elements before which no element is bigger in array(0.285306190981)

Reverse Words in a String III(557)

- Reverse words in a given string(0.602974816038)
- Perfect reversible string(0.411207055068)
- Length Of Last Word in a String(0.411207055068)
- Count words in a given string(0.336096927276)
- String containing first letter of every word in a given string with spaces(0.318849541433)
- Write a program to reverse an array or string(0.291219418564)
- Reverse a string preserving space positions(0.291219418564)
- Program to find Smallest and Largest Word in a String(0.291219418564)

- Program to extract words from a given String(0.291219418564)
- Print reverse of a string using recursion(0.291219418564)

Subarray Sum Equals K(560)

- Sum of all Subarrays(0.579738671538)
- Split an array into two equal Sum subarrays(0.51014901931)
- Number of subarrays for which product and sum are equal(0.51014901931)
- Find if array can be divided into two subarrays of equal sum(0.51014901931)
- Count all sub-arrays having sum divisible by k(0.51014901931)
- Partition of a set into K subsets with equal sum(0.450175502327)
- Largest sum subarray with at-least k numbers(0.450175502327)
- Find maximum (or minimum) sum of a subarray of size k(0.450175502327)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.450058913045)
- Find number of subarrays with even sum(0.411207055068)

Array Partition I(561)

- Find a partition point in array(0.709297266606)
- Three way partitioning of an array around a given range(0.502328778226)
- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Minimum toggles to partition a binary array so that it has first 0s then 1s(0.379978361591)
- Find Duplicates of array using bit array(0.368023208756)

- Pointer to an Array | Array Pointer(0.336096927276)
- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)

Longest Line of Consecutive One in Matrix(562)

- Longest Consecutive Subsequence(0.411207055068)
- Transpose a matrix in Single line in Python(0.291219418564)
- Print consecutive characters together in a line(0.291219418564)
- Longest consecutive sequence in Binary tree(0.291219418564)
- Longest Possible Route in a Matrix with Hurdles(0.291219418564)
- Find the longest path in a matrix with given constraints(0.291219418564)
- Length of the Longest Consecutive 1s in Binary Representation(0.260555671056)
- Find length of the longest consecutive path from a given starting character(0.237903094633)
- Print 2D matrix in different lines and without curly braces in C/C++?(0.220288150562)
- Queries in a Matrix(0.220288150562)

Binary Tree Tilt(563)

- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)

- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)
- Diameter of a Binary Tree(0.503102612415)
- Bottom View of a Binary Tree(0.503102612415)

Find the Closest Palindrome(564)

- Palindromic Primes(0.336096927276)
- Smallest Palindrome after replacement(0.260555671056)
- Palindrome Substring Queries(0.260555671056)
- Lexicographically first palindromic string(0.260555671056)
- Check if a number is Palindrome(0.260555671056)
- Two elements whose sum is closest to zero(0.220288150562)
- Queries on substring palindrome formation(0.220288150562)
- Print all palindromic partitions of a string(0.220288150562)
- Print all palindrome permutations of a string(0.220288150562)
- Palindromic Tree | Introduction & Implementation(0.220288150562)

Array Nesting(565)

- Find original array from encrypted array (An array of sums of other elements)(0.423051366241)
- Construct an array from its pair-sum array(0.411207055068)
- Find Duplicates of array using bit array(0.368023208756)
- Pointer to an Array | Array Pointer(0.336096927276)

- Leaders in an array(0.336096927276)
- Find pairs in array whose sums already exist in array(0.336096927276)
- Emulating a 2-d array using 1-d array(0.336096927276)
- Arrays in Java(0.336096927276)
- Arrays in Java(0.336096927276)
- kasai's Algorithm for Construction of LCP array from Suffix Array(0.311257467527)

Reshape the Matrix(566)

- Queries in a Matrix(0.336096927276)
- Matrix Introduction(0.336096927276)
- Matrix Exponentiation(0.336096927276)
- Determinant of a Matrix(0.336096927276)
- Circular Matrix (Construct a matrix with numbers 1 to m*n in spiral way)(0.274611786436)
- Saddle point in a matrix(0.260555671056)
- Rotate Matrix Elements(0.260555671056)
- Program for Rank of Matrix(0.260555671056)
- Implementation of a Falling Matrix(0.260555671056)
- Form coils in a matrix(0.260555671056)

Permutation in String(567)

- Print all palindrome permutations of a string(0.579738671538)
- Number of distinct permutation a String can have(0.579738671538)
- All permutations of a string using iteration(0.579738671538)

- Permutations of a given string using STL(0.502328778226)
- Write a program to print all permutations of a given string(0.449436416524)
- Print all distinct permutations of a given string with duplicates(0.449436416524)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.423051366241)
- Pairs of complete strings in two sets of strings(0.368023208756)
- Given two strings, find if first string is a subsequence of second(0.368023208756)
- String matching where one string contains wildcard characters(0.336096927276)

Maximum Vacation Days(568)

- Basic Linux Commands for day to day life(0.260555671056)
- Sliding Window Maximum (Maximum of all subarrays of size k)(0.241299136472)
- Maximum Product Subarray(0.201993092498)
- Maximum Bipartite Matching(0.201993092498)
- Find the maximum number of handshakes(0.201993092498)
- Type of array and its maximum element(0.17077611319)
- Sum of maximum elements of all subsets(0.17077611319)
- Subsequence with maximum odd sum(0.17077611319)
- Puzzle 22 | (Maximum Chocolates)(0.17077611319)
- Path with maximum average value(0.17077611319)

Subtree of Another Tree(572)

- Check if a binary tree is subtree of another binary tree | Set 2(0.518641541237)

- Check if a binary tree is subtree of another binary tree | Set 1(0.518641541237)
- Subtree with given sum in a Binary Tree(0.356300429333)
- Convert a given tree to its Sum Tree(0.285306190981)
- Binary Indexed Tree or Fenwick Tree(0.285306190981)
- Find the largest BST subtree in a given Binary Tree | Set 1(0.269517613246)
- Check if a Binary Tree contains duplicate subtrees of size 2 or more(0.269517613246)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.260555671056)
- Tree Sort(0.260555671056)
- Tournament Tree (Winner Tree) and Binary Heap(0.260555671056)

Squirrel Simulation(573)

- Simulating final class in C++(0.220288150562)
- C program to simulate Nondeterministic Finite Automata (NFA)(0.161713780663)

Distribute Candies(575)

- Chocolate Distribution Problem(0.260555671056)
- GCD, LCM and Distributive Property(0.220288150562)
- random header | Set 2 (Distributions)(0.194314340169)
- MPI – Distributed Computing made easy(0.194314340169)
- Find the minimum and maximum amount to buy all N candies(0.194314340169)
- random header in C++ | Set 3 (Distributions)(0.175786078393)
- Random number generator in arbitrary probability distribution fashion(0.161713780663)

Out of Boundary Paths(576)

- Printing Paths in Dijkstra's Shortest Path Algorithm(0.336096927276)
- Dyck path(0.336096927276)
- Find whether there is path between two cells in matrix(0.260555671056)
- Shortest path in a Binary Maze(0.220288150562)
- Regex Boundary Matchers in Java(0.220288150562)
- Path with maximum average value(0.220288150562)
- Path Traversal Attack and Prevention(0.220288150562)
- Number of palindromic paths in a matrix(0.220288150562)
- Maximum path sum in a triangle.(0.220288150562)
- Maximum Sum Path in Two Arrays(0.220288150562)

Shortest Unsorted Continuous Subarray(581)

- Sum of all Subarrays(0.220288150562)
- Find the subarray with least average(0.220288150562)
- Continuous Tree(0.220288150562)
- Find the Minimum length Unsorted Subarray, sorting which makes the complete array sorted(0.184355541926)
- Maximum subarray size, such that all subarrays of that size have sum less than k(0.179988918812)
- Subarrays with distinct elements(0.17077611319)
- Shortest Uncommon Subsequence(0.17077611319)
- Shortest Superstring Problem(0.17077611319)
- Shortest Common Supersequence(0.17077611319)
- Maximum Product Subarray(0.17077611319)

Kill Process(582)

- Process Synchronization(0.336096927276)
- Zombie Processes and their Prevention(0.260555671056)
- Process Synchronization | Monitors(0.260555671056)
- Operating System | Process Scheduler(0.260555671056)
- Inter Process Communication(0.260555671056)
- Zombie and Orphan Processes in C(0.220288150562)
- Operating System | Process Synchronization | Introduction(0.220288150562)
- Operating System | Process Management | Introduction(0.220288150562)
- Operating Systems | Process Synchronization | Question 5(0.194314340169)
- Operating Systems | Process Management | Question 6(0.194314340169)

Delete Operation for Two Strings(583)

- Logical Operators on String in Python(0.411207055068)
- Minimum steps to delete a string after repeated deletion of palindrome substrings(0.390105265183)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.327966201641)
- new and delete operators in C++ for dynamic memory(0.318784021754)
- Toggle case of a string using Bitwise operators(0.318784021754)
- Minimum number of deletions to make a string palindrome(0.318784021754)
- Minimum delete operations to make all elements of array same(0.318784021754)
- Minimum number of deletions and insertions to transform one string into another(0.291069102382)
- Find largest word in dictionary by deleting some characters of given string(0.291069102382)

- Pairs of complete strings in two sets of strings(0.285306190981)

Erect the Fence(587)

- Rail Fence Cipher – Encryption and Decryption(0.175786078393)
- Peterson's Algorithm for Mutual Exclusion | Set 2 (CPU Cycles and Memory Fence)(0.133785092946)

Design In-Memory File System(588)

- File Systems | Operating System(0.260555671056)
- C Program to merge contents of two files into a third file(0.260555671056)
- C program to copy contents of one file to another file(0.241299136472)
- Jar files in Java(0.201993092498)
- Flyweight Design Pattern(0.201993092498)
- File Permissions in Java(0.201993092498)
- File Objects in Python(0.201993092498)
- File Allocation Methods(0.201993092498)
- Comment in header file name?(0.201993092498)
- Automating File Movement on your system(0.201993092498)

Tag Validator(591)

- Valid variants of main() in Java(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Length of the longest valid substring(0.220288150562)
- Validity of a given Tic-Tac-Toe board configuration(0.194314340169)

- How to check if a string is a valid keyword in Python?(0.194314340169)
- Tag Sort (To get both sorted and original)(0.175786078393)
- Find the number of valid parentheses expressions of given length(0.175786078393)
- Print all valid words that are possible using Characters of Array(0.161713780663)
- Check if a given string is a valid number (Integer or Floating Point)(0.150556969602)

Fraction Addition and Subtraction(592)

- String with additive sequence(0.201993092498)
- Repeated subtraction among two numbers(0.201993092498)
- Program to add two fractions(0.201993092498)
- Fractional Knapsack Problem(0.201993092498)
- Fraction module in Python(0.201993092498)
- Find Recurring Sequence in a Fraction(0.201993092498)
- Addition and Concatenation in Java(0.201993092498)
- Greedy Algorithm for Egyptian Fraction(0.17077611319)
- Convert Binary fraction to Decimal(0.17077611319)
- C program for subtraction of matrices(0.17077611319)

Valid Square(593)

- Magic Square(0.336096927276)
- Latin Square(0.336096927276)
- Square root of an integer(0.260555671056)
- Direction at last square block(0.260555671056)

- Valid variants of main() in Java(0.220288150562)
- Program to validate an IP address(0.220288150562)
- Program to find number of squares in a chessboard(0.220288150562)
- Program to check if a date is valid or not(0.220288150562)
- Nth Square free number(0.220288150562)
- Maximum and Minimum in a square matrix.(0.220288150562)

Longest Harmonious Subsequence(594)

- Longest alternating subsequence(0.503102612415)
- Longest Zig-Zag Subsequence(0.503102612415)
- Longest Repeating Subsequence(0.503102612415)
- Longest Consecutive Subsequence(0.503102612415)
- Printing Longest Common Subsequence(0.411207055068)
- Printing Longest Bitonic Subsequence(0.411207055068)
- Longest subsequence such that difference between adjacents is one(0.411207055068)
- Longest common subsequence with permutations allowed(0.356300429333)
- LCS (Longest Common Subsequence) of three strings(0.356300429333)
- Print all longest common sub-sequences in lexicographical order(0.318784021754)

Range Addition II(598)

- Bitwise and (or &) of a range(0.260555671056)
- Binary Indexed Tree : Range Update and Range Queries(0.241299136472)
- String with additive sequence(0.201993092498)

- Range LCM Queries(0.201993092498)
- Perfect cubes in a range(0.201993092498)
- Find missing elements of a range(0.201993092498)
- Addition and Concatenation in Java(0.201993092498)
- range() vs xrange() in Python(0.17077611319)
- Min-Max Range Queries in Array(0.17077611319)
- Find the smallest twins in given range(0.17077611319)

Minimum Index Sum of Two Lists(599)

- Minimum sum of two elements from two arrays such that indexes are not same(0.51014901931)
- Find sum of sum of all sub-sequences(0.310890774681)
- Minimum Sum Path In 3-D Array(0.291219418564)
- Sum of minimum absolute difference of each array element(0.260555671056)
- Minimum sum of two numbers formed from digits of an array(0.260555671056)
- Minimum sum of two numbers formed from digits of an array(0.260555671056)
- Minimum sum of absolute difference of pairs of two arrays(0.260555671056)
- Maximum and minimum sums from two numbers with digit replacements(0.260555671056)
- Find pairs with given sum in doubly linked list(0.260555671056)
- Find minimum sum such that one of every three consecutive elements is taken(0.260555671056)

Non-negative Integers without Consecutive Ones(600)

- Longest Subarray of non-negative Integers(0.411207055068)

- Median in a stream of integers (running integers)(0.285306190981)
- Count of m digit integers that are divisible by an integer n(0.241299136472)
- Count Distinct Non-Negative Integer Pairs (x, y) that Satisfy the Inequality x$x + y$y < n(0.215070325706)
- Square root of an integer(0.201993092498)
- Sorting Big Integers(0.201993092498)
- Longest Consecutive Subsequence(0.201993092498)
- Integer Promotions in C(0.201993092498)
- Check for Integer Overflow(0.201993092498)
- Smallest of three integers without comparison operators(0.17077611319)

Design Compressed String Iterator(604)

- All permutations of a string using iteration(0.336096927276)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.277280735105)
- Program to count vowels in a string (Iterative and Recursive)(0.260555671056)
- Pairs of complete strings in two sets of strings(0.241213606675)
- Given two strings, find if first string is a subsequence of second(0.241213606675)
- Python | Set 3 (Strings, Lists, Tuples, Iterations)(0.237903094633)
- String matching where one string contains wildcard characters(0.220288150562)
- Sort an array of strings according to string lengths(0.220288150562)
- Search in an array of strings where non-empty strings are sorted(0.220288150562)
- Remove characters from the first string which are present in the second string(0.220288150562)

Can Place Flowers(605)

- Puzzle 42 | (Placing the numbers)(0.220288150562)
- Print all possible strings that can be made by placing spaces(0.194314340169)
- Print * in place of characters for reading passwords in C(0.175786078393)
- Place k elements such that minimum distance is maximized(0.175786078393)
- How Linkers Resolve Global Symbols Defined at Multiple Places?(0.161713780663)
- Check if a string can be obtained by rotating another string 2 places(0.133785092946)
- [TopTalent.in] Interview with Sujeet Gholap, placed in Microsoft, Google, Samsung, Goldman Sachs & Tower Research(0.121603314786)
- [TopTalent.in] Exclusive Interview with Ravi Kiran from BITS, Pilani who got placed in Google, Microsoft and Facebook(0.121603314786)

Construct String from Binary Tree(606)

- Check whether a binary tree is a full binary tree or not(0.519174772633)
- Construct a Binary Tree from Postorder and Inorder(0.51014901931)
- Binary Tree to Binary Search Tree Conversion(0.474330706497)
- If you are given two traversal sequences, can you construct the binary tree?(0.450175502327)
- Construct a Binary Search Tree from given postorder(0.450175502327)
- Construct Ancestor Matrix from a Given Binary Tree(0.450175502327)
- Binary Tree | Set 3 (Types of Binary Tree)(0.439404118785)
- fork() and Binary Tree(0.411207055068)
- Threaded Binary Tree(0.411207055068)
- Foldable Binary Trees(0.411207055068)

Find Duplicate File in System(609)

- File Systems | Operating System(0.336096927276)
- C Program to merge contents of two files into a third file(0.336096927276)
- C program to copy contents of one file to another file(0.311257467527)
- Jar files in Java(0.260555671056)
- Find duplicates under given constraints(0.260555671056)
- File Permissions in Java(0.260555671056)
- File Objects in Python(0.260555671056)
- File Allocation Methods(0.260555671056)
- Comment in header file name?(0.260555671056)
- Automating File Movement on your system(0.260555671056)

Valid Triangle Number(611)

- Number of Triangles in an Undirected Graph(0.411207055068)
- Count the number of possible triangles(0.411207055068)
- Number of possible Triangles in a Cartesian coordinate system(0.356300429333)
- Number of Triangles in Directed and Undirected Graphs(0.356300429333)
- Maximum sum of a path in a Right Number Triangle(0.318784021754)
- Finding the number of triangles amongst horizontal and vertical line segments(0.318784021754)
- Find the number of valid parentheses expressions of given length(0.318784021754)
- Smallest number divisible by first n numbers(0.285306190981)
- Number with maximum number of prime factors(0.285306190981)
- Number of subtrees having odd count of even numbers(0.285306190981)

Add Bold Tag in String(616)

- Add two bit strings(0.411207055068)
- Program to add two binary strings(0.336096927276)
- Meta Strings (Check if two strings can become same after a swap in one string)(0.277280735105)
- Pairs of complete strings in two sets of strings(0.241213606675)
- Given two strings, find if first string is a subsequence of second(0.241213606675)
- String matching where one string contains wildcard characters(0.220288150562)
- Sort an array of strings according to string lengths(0.220288150562)
- Search in an array of strings where non-empty strings are sorted(0.220288150562)
- Remove characters from the first string which are present in the second string(0.220288150562)
- Check if given string can be split into four distinct strings(0.220288150562)

Merge Two Binary Trees(617)

- Merge Two Balanced Binary Search Trees(0.656972921033)
- Check whether a binary tree is a full binary tree or not(0.635198694168)
- Binary Tree to Binary Search Tree Conversion(0.580332984677)
- Binary Tree | Set 3 (Types of Binary Tree)(0.537601087682)
- fork() and Binary Tree(0.503102612415)
- Threaded Binary Tree(0.503102612415)
- Foldable Binary Trees(0.503102612415)
- Flip Binary Tree(0.503102612415)
- Enumeration of Binary Trees(0.503102612415)

- Diameter of a Binary Tree(0.503102612415)

Task Scheduler(621)

- Weighted Job Scheduling(0.260555671056)
- Operating System | Process Scheduler(0.260555671056)
- Disk Scheduling Algorithms(0.260555671056)
- DBMS | Recoverability of Schedules(0.260555671056)
- Puzzle 32| (Completion of Task)(0.220288150562)
- Program for Priority Scheduling | Set 1(0.194314340169)
- Program for FCFS Scheduling | Set 1(0.194314340169)
- Operating Systems | CPU Scheduling | Question 6(0.194314340169)
- Operating Systems | CPU Scheduling | Question 5(0.194314340169)
- Operating Systems | CPU Scheduling | Question 4(0.194314340169)

Add One Row to Tree(623)

- Find all permuted rows of a given row in a matrix(0.285306190981)
- Convert a given tree to its Sum Tree(0.285306190981)
- Binary Indexed Tree or Fenwick Tree(0.285306190981)
- Two Dimensional Binary Indexed Tree or Fenwick Tree(0.260555671056)
- Tree Sort(0.260555671056)
- Tournament Tree (Winner Tree) and Binary Heap(0.260555671056)
- Quad Tree(0.260555671056)
- Interval Tree(0.260555671056)
- Expression Tree(0.260555671056)

- Double Tree(0.260555671056)

Maximum Distance in Arrays(624)

- Maximum distance between two occurrences of same element in array(0.656972921033)
- Type of array and its maximum element(0.411207055068)
- Maximum Sum Path in Two Arrays(0.411207055068)
- Maximum difference between first and last indexes of an element in array(0.356300429333)
- Find the maximum subarray XOR in a given array(0.356300429333)
- Find the maximum element in an array which is first increasing and then decreasing(0.356300429333)
- Find a pair with maximum product in array of Integers(0.356300429333)
- Find original array from encrypted array (An array of sums of other elements)(0.327966201641)
- Recursive Programs to find Minimum and Maximum elements of array(0.318784021754)
- Recursive Programs to find Minimum and Maximum elements of array(0.318784021754)

Minimum Factorization(625)

- Second minimum element using minimum comparisons(0.336096927276)
- No of Factors of n!(0.336096927276)
- Maximum and minimum of an array using minimum number of comparisons(0.311257467527)
- Minimum step to reach one(0.260555671056)
- Find the minimum distance between two numbers(0.260555671056)

- Find a number in minimum steps(0.260555671056)
- Roots of a tree which give minimum height(0.220288150562)
- Print all prime factors and their powers(0.220288150562)
- Minimum steps to reach a destination(0.220288150562)
- Minimum lines to cover all points(0.220288150562)