4: Seach in Rotated Sorted Array II

9: 3Sum Closest

68: Flatten Binary Tree to Linked List

77: Maximum Depth of Binary Tree

87: Sort List

95: Permutations I

124: Maximal Rectangle

132: Word Break

135: Reverse Integer

139: Mininum Window Substring

**2.1.4 Search in Rotated Sorted Array II**

描述

Follow up for ”Search in Rotated Sorted Array”: What if duplicates are allowed?

Would this affect the run-time complexity? How and why?

Write a function to determine if a given target is in the array.

*Reference for “Search in Rotated Sorted Array”:*

*Suppose a sorted array is rotated at some pivot unknown to you beforehand.*

*(i.e., 0 1 2 4 5 6 7 might become 4 5 6 7 0 1 2).*

*You are given a target value to search. If found in the array return its index, otherwise return -1.*

*You may assume no duplicate exists in the array.*

Solution:

int locateFrom(vector<int>& input, int target, int start, int end) {

if (start>=end) return -1;

if (input[start]==target) return start;

int mid = (start+end)/2;

if ( (input[start]<=target && target <=input[mid-1])

|| (input[start]>=input[mid-1] && (input[start]<=target || target<=input[mid-1])) )

return locateFrom(input, target, start, mid);

if ( (input[mid]<=target && target<=input[end-1])

|| (input[mid]>=input[end-1] && (input[mid]<=target || target<=input[end-1])) )

return locateFrom(input, target, mid, end);

return -1;

}

int locateFrom(vector<int>& input, int target) {

int start = 0, end = input.size();

return locateFrom(input, target, start, end);

}

**2.1.9 3Sum Closest**

描述

Given an array *S* of *n* integers, find three integers in *S* such that the sum is closest to a given number,

target. Return the sum of the three integers. You may assume that each input would have exactly one solution.

For example, given array S = {-1 2 1 -4}, and target = 1.

The sum that is closest to the target is 2. (-1 + 2 + 1 = 2).

**5.1.11 Flatten Binary Tree to Linked List**

描述

Given a binary tree, flatten it to a linked list in-place.

For example, Given

1

/ \

2 5

/ \ \

3 4 6

The flattened tree should look like:

1

\

2

**Sort List**

描述

Sort a linked list in *O*(*nlogn*) time using constant space complexity.

\

3

\

4

\

5

\

6

**5.4.2 Maximum Depth of Binary Tree**

描述

Given a binary tree, find its maximum depth.

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

farthest leaf node.

**8.3 Permutations**

描述

Given a collection of numbers, return all possible permutations.

For example, [1,2,3] have the following permutations: [1,2,3], [1,3,2], [2,1,3], [2,3,1],

[3,1,2], and [3,2,1].

**13.4 Maximal Rectangle**

描述

Given a 2D binary matrix filled with 0’s and 1’s, find the largest rectangle containing all ones and return

its area.

**13.12 Word Break**

描述

Given a string s and a dictionary of words dict, determine if s can be segmented into a space-separated

sequence of one or more dictionary words.

For example, given

s = "leetcode",

dict = ["leet", "code"].

Return true because "leetcode" can be segmented as "leet code".

**15.1 Reverse Integer**

描述

Reverse digits of an integer.

Example1: x = 123, return 321

Example2: x = -123, return -321

**Have you thought about this?**

Here are some good questions to ask before coding. Bonus points for you if you have already thought

through this!

If the integer’s last digit is 0, what should the output be? ie, cases such as 10, 100.

Did you notice that the reversed integer might overflow? Assume the input is a 32-bit integer, then the

reverse of 1000000003 overflows. How should you handle such cases?

Throw an exception? Good, but what if throwing an exception is not an option? You would then have

to re-design the function (ie, add an extra parameter).

**15.5 Minimum Window Substring**

描述

Given a string *S* and a string *T*, find the minimum window in *S* which will contain all the characters in

*T* in complexity *O*(*n*).

For example, S = ”ADOBECODEBANC”, T = ”ABC”

Minimum window is ”BANC”.

Note:

• If there is no such window in *S* that covers all characters in *T*, return the emtpy string ””.

• If there are multiple such windows, you are guaranteed that there will always be only one unique

minimum window in *S*.