Combination Sum II

My Submissions (/problems/combination-sum-ii/submissions/)

Question | Solution

Total Accepted: **45886** Total Submissions: **180560** Difficulty: **Medium**

Given a collection of candidate numbers (***C***) and a target number (***T***), find all unique combinations in ***C*** where the candidate numbers sums to ***T***.

Each number in ***C*** may only be used **once** in the combination.

Note:

- All numbers (including target) will be positive integers.
- Elements in a combination (a_1, a_2, … , a_k) must be in non-descending order. (ie, $a_1 \le a_2 \le \ldots \le a_k$).
- The solution set must not contain duplicate combinations.

For example, given candidate set `10,1,2,7,6,1,5` and target `8`,

A solution set is:

```
[1, 7]

[1, 2, 5]

[2, 6]

[1, 1, 6]
```

Show Tags

Show Similar Problems

Have you met this question in a real interview? Yes No

Discuss (/discuss/questions/oj/combination-sum-ii)

C

```
/**
 * Return an array of arrays of size *returnSize.
 * The sizes of the arrays are returned as *columnSizes array.
 * Note: Both returned array and *columnSizes array must be malloced, assume caller cal
 */
void quick_sort(int* candidates,int start,int end){
    if(start >= end)return;
    int i=start,j=end,temp = candidates[start];
    while(i < j){
        while(i < j && candidates[j] >= temp)j--;
        if(i < j){
            candidates[i] = candidates[j];
        }
        while(i < j && candidates[i] < temp)i++;
        if(i < j){
```

Send Feedback (mailto:admin@leetcode.com?subject=Feedback)

```
16              candidates[j] = candidates[i];
17          }
18      }
19      candidates[i] = temp;
20      quick_sort(candidates,start,i-1);
21      quick_sort(candidates,i+1,end);
22  }
23  void DFS(int* candidates,int candidatesSize,int step,int target,int* prefix,int prefixS
    umn){
24      if(step >= candidatesSize)return;
25      if(target <=0 )return;
26      int i;
27      for(i=step;i<candidatesSize;i++){
28          if(i>step && candidates[i] == candidates[i-1])continue;
29          if(candidates[i] > target)return;
30          else if(candidates[i] == target){
31              returnColumn[*returnSize] = (int*)malloc(sizeof(int)*(prefixSize+1));
32              prefix[prefixSize] = candidates[i];
33              memcpy(returnColumn[*returnSize],prefix,sizeof(int)*(prefixSize+1));
34              columnSizes[0][*returnSize] = prefixSize+1;
35              (*returnSize)++;
36              break;
37          }else{
38              prefix[prefixSize] = candidates[i];
39              DFS(candidates,candidatesSize,i+1,target-candidates[i],prefix,prefixSize+1,
40          }
41      }
42  }
43  int** combinationSum2(int* candidates, int candidatesSize, int target, int** columnSize
44      quick_sort(candidates,0,candidatesSize-1);
45      int** returnColumn = (int**)malloc(sizeof(int*)*100000);
46      int* prefix = (int*)malloc(sizeof(int)*100000);
47      columnSizes[0] = (int*)malloc(sizeof(int)*100000);
48      returnSize[0] = 0;
49      DFS(candidates,candidatesSize,0,target,prefix,0,columnSizes,returnSize,returnColumn
50      return returnColumn;
```

Custom Testcase

```
[10,1,2,7,6,1,5,1,1,1,2,4]
10
```

One line for one parameter.

Run Code | Submit Solution

Submission Result: Accepted (/submissions/detail/39750872/)

More Details ❯ (/submissions/detail/39750872/)

Next challenges:

(M) Construct Binary Tree from Inorder and Postorder Traversal (/problems/construct-binary-tree-from-inorder-and-postorder-traversal/)

(M) Spiral Matrix II (/problems/spiral-matrix-ii/) (M) Shortest Word Distance III (/problems/shortest-word-distance-iii/)

Share your acceptance!