Combination Sum-II

Approach :

Combination of Subset sum-II and combination sum I

```
#include <bits/stdc++.h>
void helper(int ind,vector<int>& arr,int n,int target,vector<int> temp,vector<vector<int>>& ans)
{
  if(target==0)
    ans.push_back(temp);
  for(int i=ind;i<n;i++)
  {
    if(i!=ind && arr[i]==arr[i-1]) continue;
    temp.push_back(arr[i]);
    helper(i+1,arr,n,target-arr[i],temp,ans);
    temp.pop_back();
  }
}
vector<vector<int>> combinationSum2(vector<int> &arr, int n, int target)
{
  // Write your code here.
  vector<int> temp;
  vector<vector<int>> ans;
  sort(arr.begin(),arr.end());
  helper(0,arr,n,target,temp,ans);
  return ans;
}
```

- Time Complexity : O(2^N*k)

Reason: Assume if all the elements in the array are unique then the no. of subsequence you will get will be O(2^n). we also add the ds to our ans when we reach the base case that will take “k”//average space for the ds.

- Space Complexity : O(k*x)

Reason: if we have x combinations then space will be x*k where k is the average length of the combination.