/\*

Given an array S of n integers, are there elements a, b, c, and d in S such that a + b + c + d = target? Find all unique quadruplets in the array which gives the sum of target.

Note: The solution set must not contain duplicate quadruplets.

For example, given array S = [1, 0, -1, 0, -2, 2], and target = 0.

A solution set is:

[

[-1, 0, 0, 1],

[-2, -1, 1, 2],

[-2, 0, 0, 2]

]

题目：4sum=target

way-1：两重for固定两个数，两个指针移动确定后两个数

way-2:先对给定数组做初始化，把所有两两值的和求出来，我们就把求4Sum变为求2Sum了，注意初始化的时候求两两和值也要保存两个数字的下标，题中我用的是multimap <int, pair < int, int > >，因为可能会有重复的所以要用multimap, multimap第一个int是表示和值，第二个pair是两个数的下标，而且用multimap我们默认是排序的.

最后ret要去重

\*/

class Solution {

public:

vector<vector<int>> fourSum(vector<int>& nums, int target)

{

//way-1

/\*

vector<vector<int>> result;

if(nums.size()<4)

return result;

vector<int> m1;

sort(nums.begin(),nums.end());

int left,right;

for(int i=0;i<nums.size()-3;i++)

{

if(nums[i]==nums[i-1] && i>0)

continue;

for(int j=i+1;j<nums.size()-2;j++)

{

if(nums[j]==nums[j-1] && j>i+1)

continue;

left=j+1;

right=nums.size()-1;

while(left<right)

{

if(nums[i]+nums[j]+nums[left]+nums[right]>target)

right--;

else if(nums[i]+nums[j]+nums[left]+nums[right]<target)

left++;

else

{

m1.push\_back(nums[i]);

m1.push\_back(nums[j]);

m1.push\_back(nums[left]);

m1.push\_back(nums[right]);

if(result.size()>0)

{

if(m1!=result[result.size()-1])

result.push\_back(m1);

}

else

result.push\_back(m1);

m1.clear();

right--;

left++;

}

}

}

}

return result;

\*/

//way-2

vector<vector<int>> ret;

if(nums.size()<4)

return ret;

else if(nums.size()==4)

{

if(accumulate(nums.begin(),nums.end(),0)==target)

{

ret.push\_back(nums);

return ret;

}

}

multimap<int,pair<int,int>> mm;

for(int i=0;i<nums.size();i++)

for(int j=i+1;j<nums.size();j++)

mm.insert({nums[i]+nums[j],{i,j}});

for(auto i=mm.begin();i!=mm.end();i++)

{

int x=target-i->first;

auto range=mm.equal\_range(x);

for(auto j=range.first;j!=range.second;j++)

{

int a=i->second.first;

int b=i->second.second;

int c=j->second.first;

int d=j->second.second;

if(a!=b && a!=c && a!=d && b!=c && b!=d && c!=d)

{

vector<int> m1={nums[a],nums[b],nums[c],nums[d]};

sort(m1.begin(),m1.end());

ret.push\_back(m1);

}

}

}

sort(ret.begin(), ret.end());

ret.erase(unique(ret.begin(), ret.end()),ret.end());

return ret;

}

};