3Sum

Medium

Given an array nums of *n* integers, are there elements *a*, *b*, *c* in nums such that *a* + *b* + *c* = 0? Find all unique triplets in the array which gives the sum of zero.

**Note:**

The solution set must not contain duplicate triplets.

**Example:**

Given array nums = [-1, 0, 1, 2, -1, -4],  
  
A solution set is:  
[  
 [-1, 0, 1],  
 [-1, -1, 2]  
]

**分析：**

直接暴力三重循环会超时。正确思路是**排序+两指针法**

1. 先排序
2. 枚举每一个，在中用双指针法寻找和为的数对

**注意：**

关于去重的问题。开始我采用了unordered\_map，记录每一个数字的选择情况，结果运行超时。

正确的处理是跳过重复的数字。

class Solution {  
public:  
 vector<vector<int>> threeSum(vector<int>& nums) {  
 vector<vector<int> >ans;  
 if(nums.size() <= 2)return ans;  
 sort(nums.begin(), nums.end());  
 int n = nums.size();  
 for(int i = 0; i < n - 2; ++i){  
 if(i > 0 && nums[i] == nums[i - 1])continue;  
 int m = -nums[i];  
 int j = i + 1, k = n - 1;  
 while(j < k){  
 int temp = nums[j] + nums[k];  
 if(temp == m){  
 ans.push\_back({nums[i], nums[j], nums[k]});  
 while(j + 1 < k && nums[j] == nums[j + 1])j++;  
 while(k - 1 > j && nums[k - 1] == nums[k])k--;  
 j++;  
 k--;  
 }else if(temp < m){  
 j++;  
 }else{  
 k--;  
 }  
 }  
 }  
 return ans;  
 }  
};