/\*

Given an array of integers that is already sorted in ascending order, find two numbers such that they add up to a specific target number.

The function twoSum should return indices of the two numbers such that they add up to the target, where index1 must be less than index2. Please note that your returned answers (both index1 and index2) are not zero-based.

You may assume that each input would have exactly one solution and you may not use the same element twice.

Input: numbers={2, 7, 11, 15}, target=9

Output: index1=1, index2=2

way-1：自己想的，存进map，查找返回位置.

way-2：两个指针往中间移动,这种找出两个数的这种题应该想到两个指针！！

\*/

class Solution {

public:

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

{

//way-1

/\*

vector<int> ret;

map<int,int> pp;

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

pp[nums[i]]=i+1;

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

{

map<int,int>::iterator it=pp.find(target-nums[i]);

if(it!=pp.end())

{

ret.push\_back(i+1);

ret.push\_back(it->second);

return ret;

}

}

return ret;

\*/

vector<int> ret;

int left=0;

int right=nums.size()-1;

while(left<right)

{

if(nums[left]+nums[right]==target)

{

ret.push\_back(left+1);

ret.push\_back(right+1);

return ret;

}

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

left++;

else

right--;

}

}

};