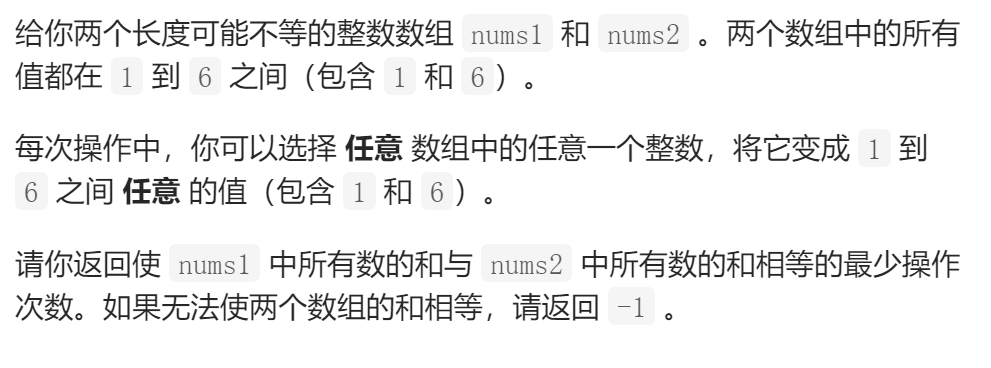
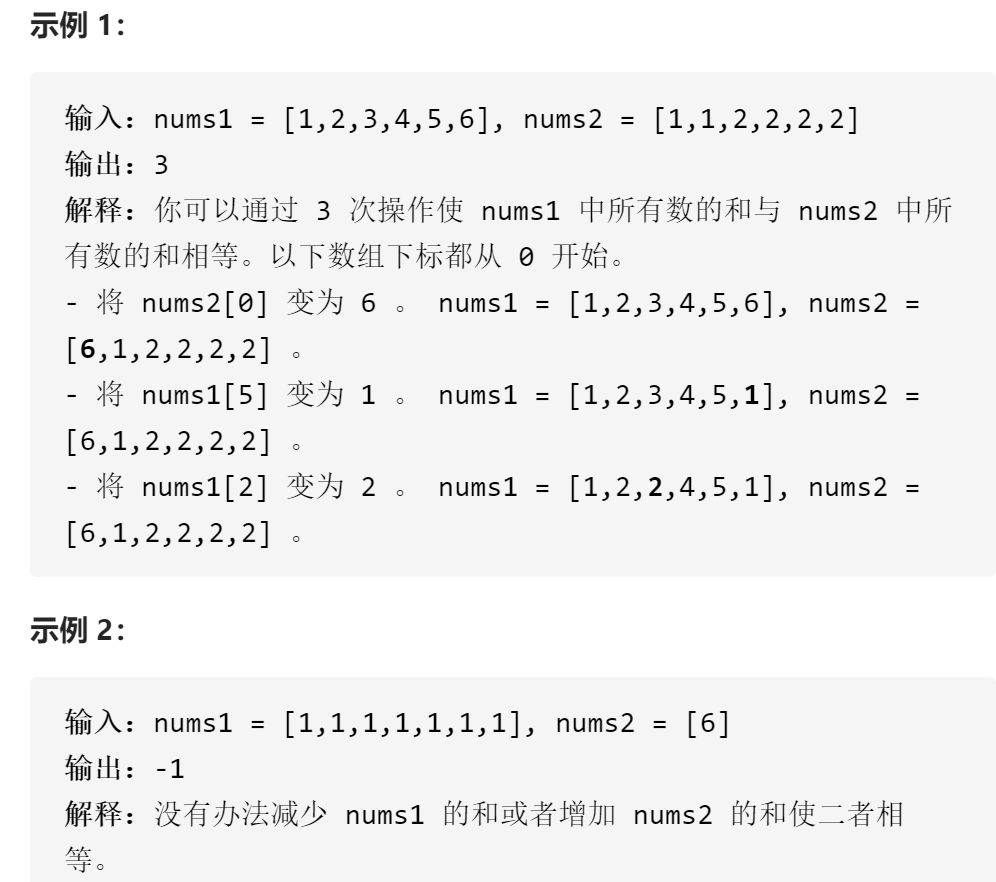
# [1775. 通过最少操作次数使数组的和相等]【贪心算法】【代码功底要求高】

## 题目描述：





## 题目分析：贪心思路可能很明显，但是并不好写，对代码的功底要求比较高

class Solution {  
public:  
 int minOperations(vector<int>& nums1, vector<int>& nums2) {  
 //贪心算法  
 //判断可行性  
 int n1=nums1.size(),n2=nums2.size();  
 if(n1>n2&&n1>6\*n2)return -1;  
 else if(n1<n2&&n2>6\*n1)return -1;  
 //排序  
 sort(nums1.begin(),nums1.end());  
 sort(nums2.begin(),nums2.end());  
 //求和  
 int sum1=0,sum2=0;  
 vector<int>dp1(7,0);//统计1-6的数字个数  
 vector<int>dp2(7,0);  
 for(int i=0;i<nums1.size();i++){  
 dp1[nums1[i]]++;  
 sum1+=nums1[i];  
 }  
 for(int i=0;i<nums2.size();i++){  
 dp2[nums2[i]]++;  
 sum2+=nums2[i];  
 }  
 int diff\_val=abs(sum1-sum2);  
 int res=0;  
  
 if(sum1>sum2){  
 int i=6,j=1,diff=5;  
 while(diff\_val>0){  
 if(diff\_val-dp1[i]\*diff>0){  
 res+=dp1[i];  
 diff\_val-=dp1[i]\*diff;  
 }else{  
 //ceil函数对整数不可用  
 //res+=ceil(diff\_val/diff);  
 if(diff\_val%diff==0)  
 res+=diff\_val/diff;  
 else res+=diff\_val/diff+1;  
 diff\_val=0;  
 break;  
 }   
  
 if(diff\_val-dp2[j]\*diff>0){  
 res+=dp2[j];  
 diff\_val-=dp2[j]\*diff;  
 }else{  
 //ceil函数对整数不可用  
 //res+=ceil(diff\_val/diff);  
 if(diff\_val%diff==0)  
 res+=diff\_val/diff;  
 else res+=diff\_val/diff+1;  
 diff\_val=0;  
 break;  
 }  
 i--;j++;diff--;  
 }  
 }else if(sum1<sum2){  
 int i=1,j=6,diff=5;  
 while(diff\_val>0){  
 if(diff\_val-dp1[i]\*diff>0){  
 res+=dp1[i];  
 diff\_val-=dp1[i]\*diff;  
 }else{  
 //ceil函数对整数不可用  
 //res+=ceil(diff\_val/diff);  
 if(diff\_val%diff==0)  
 res+=diff\_val/diff;  
 else res+=diff\_val/diff+1;  
 diff\_val=0;  
 break;  
 }   
  
 if(diff\_val-dp2[j]\*diff>0){  
 res+=dp2[j];  
 diff\_val-=dp2[j]\*diff;  
 }else{  
 //ceil函数对整数不可用  
 //res+=ceil(diff\_val/diff);  
 if(diff\_val%diff==0)  
 res+=diff\_val/diff;  
 else res+=diff\_val/diff+1;  
 diff\_val=0;  
 break;  
 }  
 i++;j--;diff--;  
 }  
 }  
  
 return res;  
 }  
};