Problem2—程序设计思路及时间复杂度分析

$O(N^2)$

程序设计思路：用枚举法即可解决问题。设置一个两层循环让数组中的每个元素与其它元素配对，检查两个元素之和是否为 K，若检查发现符合，则进一步调整输出位置达到要求。

源码如下：

```
void Nsqr(int a[],int size,int k){
    int min,max;
    for(int i=0;i<size-1;++i)
        for(int j=i+1;j<size;++j)
            if(a[i]+a[j]==k){
                min=a[i];
                max=a[j];
                if(min>max){
                    min=a[j];
                    max=a[i];
                }
                 cout<<"("<<min<<","<<max<<")";
            }
}
```

时间复杂度分析：外层循环次数为 N，内层最大循环次数为 N-1，因此这个算法是 $O(N^2)$

O(NlogN)

程序设计思路：考虑到要用排序，发现归并和快排的时间复杂度都是 NlogN，任选一个，得到排过序后的数组。然后用两个参数进行前后配对，前一个对象往后推进，后一个对象往前推进，若两者之和小于给定数值，说明前者小，前者往后移动一个位置，否则相反。当两个对象碰头时，所有情况检查完毕。

源码如下：

```
void nlogN(int a[],int size,int k){
    quickSort(a,size);
    int j=size-1,i=0;
    while(i<j){
        if(a[i]+a[j]>k)--j;
        else if(a[i]+a[j]<k) ++i;
        else {
            cout<<"("<<a[i]<<","<<a[j]<<")";
            ++i; --j;
            }
    }
}
```

时间复杂度分析：快排时间复杂度为 O(NlogN)，后者扫描过程中时间复杂度为 O(N)，则整个算法的时间复杂度为 O(NlogN)+O(N)=O(NlogN)。