1、假设二元计数器初始时有*b*个1，而不是全0. 给定，说明执行n个自增操作的代价为O(*n*).

解：设置势能为二元计数器中1的个数，由于初始状态下1的个数，这样的势能设置方法不能保证，即平摊代价之和不再是真实代价之和的上界，但是上述数学关系依旧是满足的。因此可以得到

.

根据势能设置方法得到，此处与课堂上讲授的方法一致，但是要注意这里的是真实代价之和的上界。

因此，.

当时，显然可以得到. 注意，如果，则.

2、用两个栈实现一个队列，使得ENQUEUE和DEQUEUE操作的平摊代价都是O(1).

解：进队列操作将元素压入栈A，出队列操作将栈B中栈顶元素弹出，若栈B中无元素，则将A中所有元素依次弹出并压入栈B.

使用会计方法，进队列操作代价4，其中1支付将元素压入A的代价，1支付该元素从A中弹出的代价，1支付该元素压入B的代价，1支付该元素弹出B的代价；出队列操作平摊代价0.

3、设计一个数据结构支持由整数组成的动态多重集合*S*（包含可能重复整数的集合）中的操作：（1）Insert(*S*, *x*)将整数*x*插入S中；（2）Delete-Larger-Half(*S*)删除*S*中最大的个整数。对于任意由Insert和Delete-Larger-Half组成的长度为*m*的操作序列，要求其总体代价是O(*m*)，并且可以在O(|*S*|)代价内输出*S*的所有元素。

解：用一个链表存储所有数据，每个节点存储一个整数。Insert操作在链表末尾添加一个节点，存储新插入的整数；Delete-Larger-Half操作首先将链表中元素拷贝到一个数组中，该数组大小为当前链表中元素的数量，然后使用selection算法查找中位数，然后并遍历所有元素查看有多少个元素与中位数相同，以此断定需要删除多少个与中位数相同的整数和所有比中位数大的整数才能完成Delete-Larger-Harf，随后遍历链表中的元素，删除比中位数大的元素，并删除相应数量的中位数元素。这个操作的代价为O(*n*)，其中*n*是执行Delete-Larger-Half操作时链表中元素的数量。不妨设Delete-Larger-Half的代价不超过*cn* (由其代价为O(*n*)可得存在常数*c*). 使用会计方法，Insert的平摊代价设置为(1+2*c*)，其中1用于支付在链表中插入元素的代价1，2*c*附加新插入的元素上，当删除个元素时，这些元素上的代价共计不少于*cn*，足以支付Delete-Larger-Half的代价（该代价不超过*cn*）；Delete-Larger-Half的平摊代价为0。任意时刻，平摊代价之和与真实代价之和的插值不小于链表中元素数量的(1+2*c*)倍，一定非负。