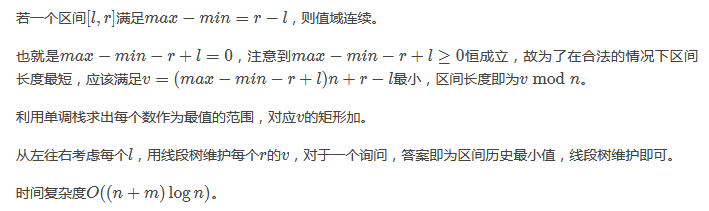
# 数字：

很容易发现，相同能量是用1和4和7最好，所以就是一个简单dp，f[i]表示使用了i份能量得到的最大数字和，考虑这三种递推就ok了。

# 区间：

离线处理所有询问，我们采用分治的算法。定义Solve（nowl,nowr）表示处理所有左端点和右端点都在[nowl,nowr]内的询问，我们分治处理Solve(nowl,mid)和Solve(mid+1,nowr)，现在我们考虑跨过mid的询问。从中间mid的位置开始，逐个逐个地往左扩展，每扩展一格，就考虑把当前这段区间合法化的最小扩展，这个扩展就是在当前区间考虑最大值最小值的位置，若其中有一个在l的左边，则显然不行。然后就把对应询问在当前区间内的处理掉。往右扩展的同理。总的时间复杂度为O（nlogn）。

Claris有一个更厉害的做法：



# 划分：

我们离线处理问题，考虑一开始加入所有栅栏，然后从后往前删除栅栏。我们将点和栅栏都当做事件，按y坐标从大到小排序，进行扫描线的算法，每一个点和每一个栅栏都存下恰好在它右边的栅栏，最后用个并查集处理一下就可以了。时间复杂度为O（nlogn）。

# 排序：

先求出s的后缀数组，以及对应的height，然后用可持久化线段树，线段树节点维护height值，将sa排名当成时间插入。每来一个询问，就二分排名，在线段树上查询一下就ok了，单次询问log2n的复杂度。