##思路1

>设置2个栈，一个叫做back栈，一个叫做forward栈。设置一个指针currentPointer用来指向这两个栈之一的栈顶。currentPointer首先指向的是back栈，因为forward栈是由back栈pop出的对象组成的。由于当前状态会首先被压入back栈中，所以想看到前一个状态，首先应该先把back栈的栈顶弹出，压入到forward栈中，之后再展示当前的栈顶对象。如果currentPointer从back栈转而指向forward栈则，先展示forward栈顶的对象，然后弹出，压入back栈中。可以看出currentPointer始终指向当前正在展示的对象。还有一个要求就是如果在forward栈不空的状态下，有新的状态对象入栈，则原来forward栈中除了currentPointer所指向的对象以外，要全部清空，如果currentPointer在forward栈中则需要把这个对象先从forward栈中弹出压入back栈中，然后再清空forward栈，然后把新对象压入到back栈中，再把currentPointer指向back的栈顶。如果是这样的话，感觉用两个栈似乎不明智。还不如用一个数组来完成呢。

首先是有一个空数组，接下来的问题是currentPointer应该指向哪里？好像指向哪里都不合适，因为currentPointer是指向当前正在展示的那个对象的。而此时currentPointer应该等于-1，因为数组里面没有任何对象。

或许可以换一个思路，就是回退是往小方向移动，前进是往大方向移动。算法只需要重复3个步骤即可。

1、增加对象。向大方向移动指针，然后把当前对象添加进数组，然后删除指针后面所有的对象。

2、回退。指针向小方向移动。如果指针左边已经没有对象了，那就证明已经是最小了。currentPointer左边的对象个数就是currentPointer的数值。在回退之前应该先判断是否能回退。

3、前进。指针向大方向移动。如果指针右边已经没有对象了，那就证明已经是最大了。currentPointer右边的对象个数就是数组的元素个数减去currentPointer的值再减去1.在forward之前应该先判断是否能前进。

注意这里不是根据指针的位置，而是根据currentPointer与两边界之间有多少个对象来判定是否还能前后移动。