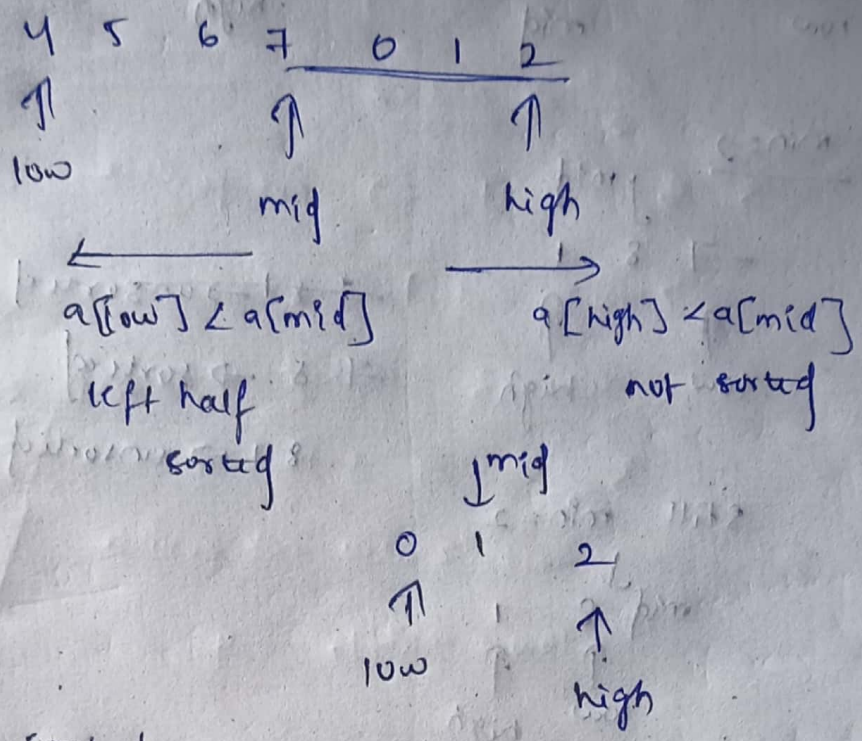
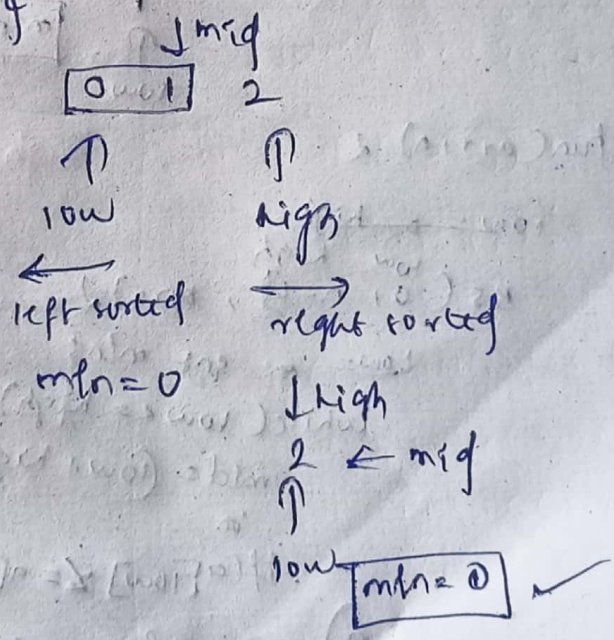
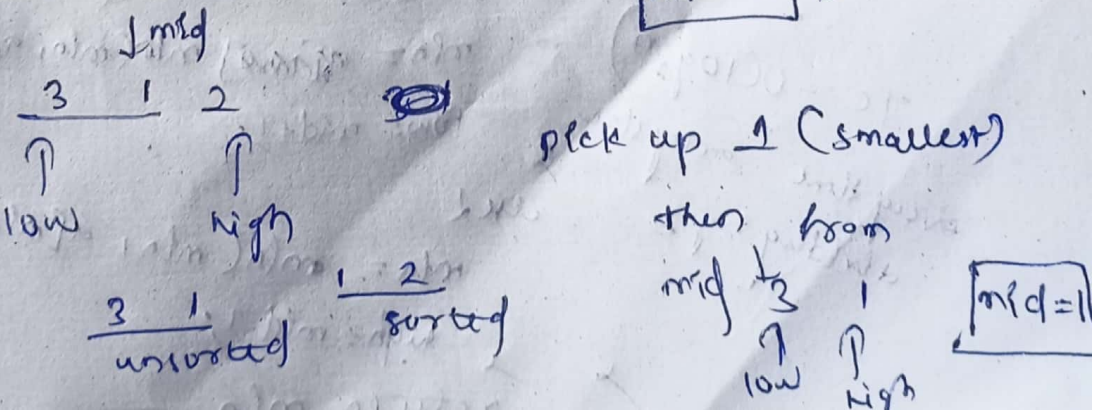
Find Minimum in rotated sorted Array

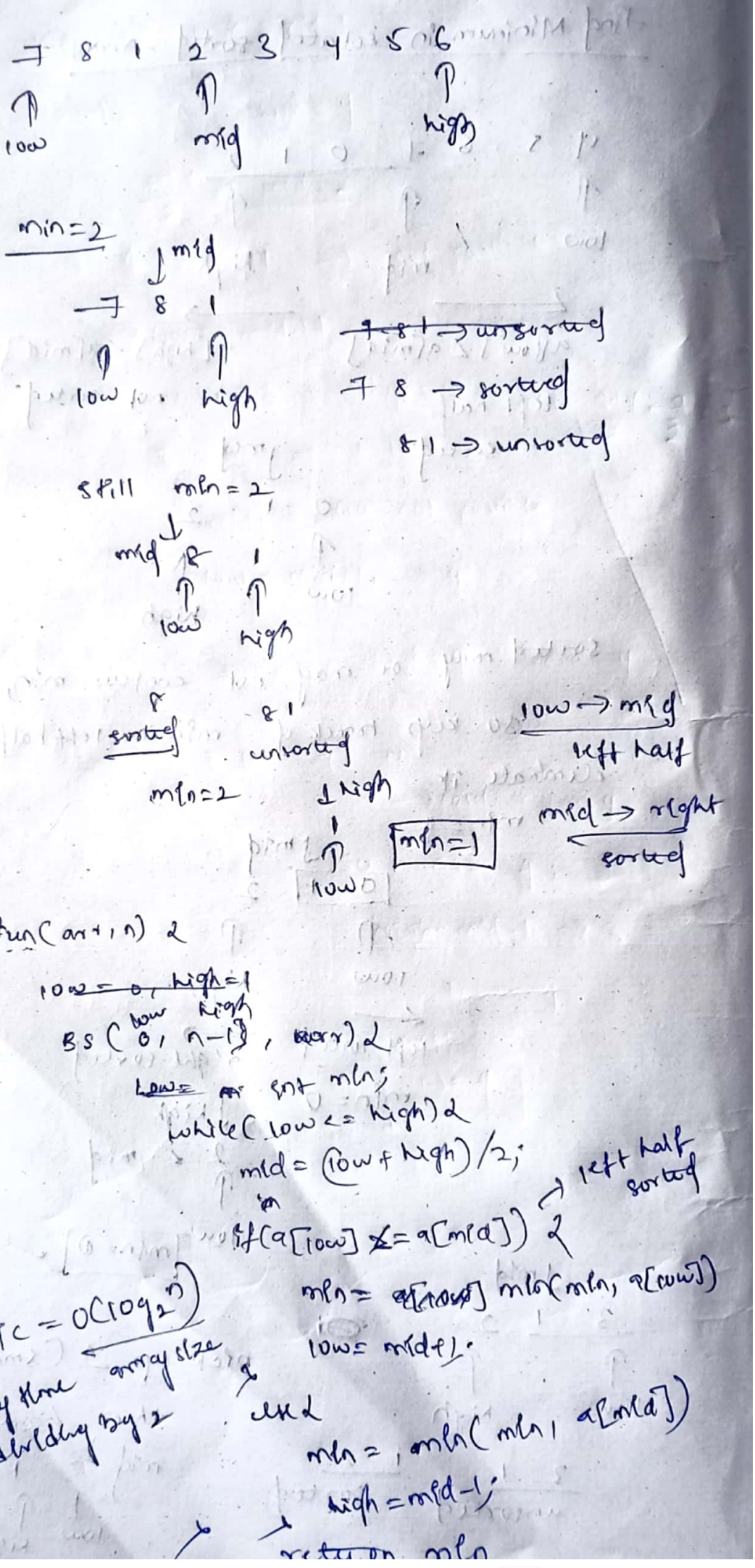


sorted may or may not contain min,
so keep track of min in left half and eliminate it

min = 0






7 8 1 2 3 4 5 6
low
mid
high
min=2
mid
7 8 1
low
high
7 8 1 → unsorted
7 8 → sorted
8 1 → unsorted
still min=2
mid
8 1
low
high
8
sorted
8 1
unsorted
min=2
high
low
min=1
low → mid
left half
mid → right
sorted
fun(arr, n) {
low=0, high=n-1
BS(0, n-1, arr) {
int mln;
while(low <= high) {
mid = (low + high)/2;
if(a[low] <= a[mid]) {
left half sorted
mln = a[low] mln(mln, a[low])
low = mid+1;
} else {
mln = mln(mln, a[mid])
high = mid-1;
return mln
TC = O(log2 n)
every time dividing by 2
array size