Divide and Conquer Practice Problems

Problem 1: Efficient Optimal Band Booking

You are the agent for a band for which you would like to book a gig at a particular venue. You've found publicly available information about the popular times of the venue. Specifically, for every hour of a particular day of the festival, you know the estimated number of people expected at the venue. Assume that the popularity of the venue is *unimodal* on the given day, that is, the number of expected people increases up to a certain time, after which it only decreases. You would obviously like to schedule your band for the peak popular time. Assume that the popularity values for a given day are provided to you in an array, ordered by time. Give an efficient algorithm to find the peak popular time for the given venue on the given day. Write the recurrence relation for your algorithm and show that its recurrence solves to $O(\log n)$ (e.g., using the Master Method or a recursion tree).

Problem 2: Count the Number of Occurrences

Suppose you are given an array A of n sorted numbers and a number x, count the number of times x occurs in A. For example, $A = \{1, 1, 2, 2, 2, 2, 3\}$ is a sorted array and $x = 2$, then your algorithm should return 4. Give an $O(\log n)$ algorithm to count the number of times x appears in A. Write the recurrence relation for your algorithm and show that its recurrence solves to $O(\log n)$ (e.g., using the Master Method or a recursion tree).

Problem 3: Median of Two Sorted Arrays - Same Size

Suppose you are given two arrays A and B both of which are sorted and contain n numbers. We want to find the median between all the numbers in A and B combined. For example, if $A = \{1, 12, 15, 26, 38\}$ and $B = \{2, 13, 17, 30, 45\}$ then the median would be 16. After merging the two lists, the two middle elements are 15 and 17, which average to 16. Write an efficient algorithm to find the median of two sorted arrays. Write the recurrence relation for your algorithm and show that its recurrence solves to $O(\log n)$ (e.g., using the Master Method or a recursion tree).

hint 1: There are 3 base cases: A and B are length 0, A and B are length 1 *and* A and B are length 2. If size of the two arrays is 2 then use below formula to get the median:

```
Median = (max(A[0], B[0]) + min(A[1], B[1])) / 2
```

hint 2: You are still making an if check to discard half the input - but what exactly is half in this case? And how can you view that "half" to get closer to the median.

Problem 4: Majority Element

Given an array A of length n, write an algorithm to find the majority element if it exists. The majority element is an element that appears more than $O(n/2)$ times in the array. For example, given $\{2, 2, 1, 1, 1, 2, 2\}$, the majority elements is 2 and for the array $\{2, 1, 3, 3, 2\}$ there is no majority element. Write the recurrence relation for your algorithm and show that its recurrence solves to $O(n \log n)$ (e.g., using the Master Method or a recursion tree).