Fundamental Algorithms, Section 003
Homework 3, Additional Problems, Fall 22.

1. Solve the following recurrence equations exactly using the recursion tree method.
a. You may assume $n = 2^k$ for some integer k.

$$T(n) = 2T(n/2) + n^2 \quad n > 1$$
$$T(1) = 1$$

Comment: Draw the tree.
And check that your solution is correct for the base case and the next larger value of n.
b. Repeat part (a), but with $T(1) = 0$.
Moral: pay attention to the base case.

2. Let $A[1:n]$ be an array in which the items are in sorted order, but starting at entry $A[i]$ for some i, $1 \leq i \leq n$; i.e. $A[i] < A[i+1] < \ldots < A[n] < A[1] < \ldots < A[i-1]$. Give a recursive $O(\log n)$ algorithm to find the largest value in A.

3. Let $A[1:n]$ be an array of values (these are not integers). You are to determine if there is a majority value, i.e. a value that occurs more than half the time, or at least $\lfloor n/2 \rfloor + 1$ times. The only test you have is to test whether two values are equal. Give a recursive algorithm to find the majority value, if any.
Hint: Suppose you recursively compute the majority value for $A[1:n/2]$ and for $A[n/2+1:n]$. What can you then deduce about the majority value of $A[1:n]$?