Homework #4 Key

2.4

a.Using the master theorem with $a = 5, b = 2, d = 1$ yields a runtime of $O(n^{log_2 5}) = O(n^{2.32})$.
b.The master theorem cannot be used here, because b is not a (constant) geometric reduction factor. However, it should be clear that two recursive calls on subproblems of size $n - 1$ will result in a tree of depth n with 2^n leaves. Given that the recombination step is constant, the amount of work per level of the tree is doubling, meaning the last level of the tree does half of all the work, and in particular, results in a total runtime of $O(2^n)$.
c.Using the master theorem with $a = 9, b = 3, d = 2$ yields a runtime of $O(n^2 \log n)$.

Therefore, the fastest of the three solutions is that of c.

2.5

a. $T(n) = 2T(n/3) + 1 = \Theta(n^{log_3 2})$ by the Master theorem.
b. $T(n) = 5T(n/4) + n = \Theta(n^{log_4 5})$ by the Master theorem.
c. $T(n) = 7T(n/7) + n = \Theta(nlogn)$ by the Master theorem.
d. $T(n) = 9T(n/3) + n^2 = \Theta(n^2 logn)$ by the Master theorem.
e. $T(n) = 8T(n/2) + n^3 = \Theta(n^3 logn)$ by the Master theorem.

2.17

A simple algorithm for solving the problem for an array A is

function SELF(A)
    if $A[\lceil \frac{n}{2} \rceil] = \lceil \frac{n}{2} \rceil$ **then**
        return $A[\lceil \frac{n}{2} \rceil]$
    else if $A[\lceil \frac{n}{2} \rceil] > \lceil \frac{n}{2} \rceil$ **then**
        Self($A[1 : \lceil \frac{n}{2} \rceil]$)
    else
        Self($A[\lceil \frac{n}{2} \rceil : n]$)

From this, we can write the recurrence relation $T(n) = T(n/2) + 1$. Then, using the master theorem with $a = 1, b = 2, d = 0$ yields a runtime of $\Theta(\log n)$.