HW 2: The Master Theorem

1. Use the master theorem to find the $\Theta$ class of algorithms with the following recurrence relations. (You may assume that $T(n)$ is constant for $n = 1$.)
   
   A. $T(n) = 2T(n/4) + 1$
   B. $T(n) = 2T(n/4) + \sqrt{n}$
   C. $T(n) = 2T(n/4) + n$
   D. $T(n) = 2T(n/4) + n^2$

2. Use the master theorem to find the $\Theta$ class of algorithms with the following recurrence relations. (You may assume that $T(n)$ is constant for $n = 1$.)
   
   A. $T(n) = 2T(n/4) + n^4$
   B. $T(n) = T(7n/10) + n$
   C. $T(n) = 16T(n/4) + n^2$
   D. $T(n) = 7T(n/3) + n^2$
   E. $T(n) = 7T(n/2) + n^2$

3. Use the master theorem to find a $\Theta$ class for each of the following algorithms.
   
   A. Mergesort. (Assume the average or worst case)
   
   B. Binary search, to find some predefined value within a pre-sorted array.
   
   C. An algorithm that takes a collection of $n$ objects, and clusters similar objects together. It starts by taking all $n$ objects and splitting it into two clusters of size $\frac{n}{2}$, so that each object is in a cluster of its most similar peers. Assume that this splitting-in-half process can be done in $\Theta(n^3)$ time. It then recurses on each side, so each half is again split in half—all the way down until each object is by itself in a “cluster” of size 1.