Assignment #2
Due on Friday 2/3/2017

1. The following pseudo-code shows a variation of the merge sort algorithm where in each iteration, the list is divided into three sublists.

   1: function Merge-Sort(A, p, r)
   2:     Return If p ≥ r
   3:     q₁ ← ⌊(2 * p + r)/3⌋
   4:     q₂ ← ⌊(p + 2 * r)/3⌋
   5:     Merge-Sort(A, p, q₁)
   6:     Merge-Sort(A, q₁ + 1, q₂)
   7:     Merge-Sort(A, q₂ + 1, r)
   8:     Merge(A, p, q₁, q₂, r)
   9: end function

(a) Write down the recurrence relation that represents the running time of the above algorithm.
(b) Solve the recurrence relation using the expansion of the recurrence tree.
(c) Solve the recurrence relation using the Master method.
(d) Compare the running time of the proposed algorithm to that of the regular merge sort algorithm shown on page 34 of the textbook.

2. Consider the following recurrence relation

   \[ T(n) = 2T\left(\frac{n}{2}\right) + n \log n \]

(a) Can you solve that recurrence relation using the Master theorem? Justify your answer.
(b) Use the recurrence tree expansion method to find a tight asymptotic bound to the recurrence relation. For simplicity, assume that n is always a power of two and \(T(1) = c\).

3. Solve the following recurrences using the Master theorem.

(a) \( T(n) = 2T\left(\frac{n}{8}\right) + n \log n \)
(b) \( T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n} \)
(c) \( T(n) = 9T\left(\frac{n}{3}\right) + 8^n \)

4. Suppose that we want to create a divide and conquer matrix multiplication algorithm for square matrices. Assuming that \(n\) is a power of three, the algorithm divides each matrix of \(A, B,\) and \(C\) into nine equi-sized matrices. Then, it performs some recursive calls to compute each submatrix of \(c\).

(a) How many recursive calls need to be made so that the algorithm will have an asymptotic running time of \(\Theta(n^3)\)?
(b) What is the maximum number of recursive calls that can be made while having an asymptotic running time that is lower than that of Strassen’s algorithm?

5. Let $X$ be a $kn \times n$ matrix and $Y$ by an $n \times kn$ matrix, for some integer $k$.

(a) Describe an algorithm that computes the product $XY$ using Strassen’s algorithm as a subroutine, i.e., use it as a black-box without modifying it. Only describe your algorithm in words; pseudo-code is not required. Justify your answer, i.e., argue that your algorithm does compute $XY$ correctly. Establish its running time.

(b) Repeat part (a) for computing the product $YX$. 