MET CS566 HOMEWORK #6

1. Find the *m* matrix in the dynamic programming algorithm for iterated matrix products, $M_1 M_2 M_3 M_4 M_5$, and the dimensions of the matrices are given by the one-dimensional array *d*:

$$d = (8, 3, 2, 19, 18, 7)$$

2. The following algorithm is a divide-and-conquer algorithm for finding the maximum value in array *A[1..n]*.

Function *maximum(x,y)*
{ return maximum in A[x..y]}
1. **if** $y - x \leq 1$ **then return** *(max(A[x],A[y]))*
2. **else**
3. *max1* ← *maximum(x, ⌈(x + y) / 2⌉)*
4. *max2* ← *maximum(⌈(x + y) / 2⌉ + 1 , y)*
5. **return(***max(max1, max2***)**

   a. Give a recurrence equation for the worst-case number of comparisons used by *maximum(1, n)*, and solve this recurrence equation, (you may assume that *n* is a power of *2*)

   b. What is the running time of *maximum(1,n)*? Explain briefly.

3 What is wrong with the following statement:" *The algorithm for calculating the values of m in section 8.3 of my notes (pp. 120-127) has essentially to fill in the entries in just over half of an n × n table. Its execution time is clearly in* $\theta(n^2)$*."*