Design and Analysis of Algorithms 2011
(Home work 3)

October 4, 2011

- Due on, October 12, before 10 a.m.
- Late home works will not be accepted.
- Please give precise arguments for all statements that you write.
- Please do not hesitate to contact me if you do not understand the problems.
- Collaboration is discouraged, but not prohibited. It is recommended that you try to solve the problems on your own. You can discuss the questions with your colleagues but you should not copy solutions. Always write down your own answers. If copying is detected that may immediately lead to a grade less than 7. (**This would be followed strictly**)
- Credits would be given to partial solutions also.
- When you write an algorithm, you should briefly discuss the main idea of your algorithm, then write a pseudo code, argue about its correctness and state and prove the running time of your algorithm.
- The answers should be typed or written clearly and a hard copy is to be submitted.

1. Give asymptotic tight bounds for $T(n)$ for the following cases. Assume that $T(n)$ is a constant if $n \leq 2$. [10 points]
   (a) $T(n) = 2T(n/2) + n^3$.
   (b) $T(n) = 7T(n/3) + n^2$.
   (c) $T(n) = 3T(n/2) + n \lg n$.
   (d) $T(n) = 2T(n-1) + 1$.

2. Give asymptotic upper bounds for $M(n)$ for the following cases (i.e., find $f(n)$ such that $M(n) = O(f(n))$). Justify your answer, i.e., you should show by the substitution method, why you think your bound is correct.[20 points]

(a) $M(n) = M(n/4) + M(n/2) + n^2$.

(b) $M(n) = 2M(n/2) + n \lg n$.

3. How many lines, as a function of n (in $\Theta(.)$ form), does the following program print? Write a recurrence and solve it. You may assume n is a power of 2. [5 points]

```
function f(n)
  if n>1,
     print("still going\n")
     f(n/2)
     f(n/2)
   end if
```

4. You are given an array A with n distinct elements. You are also told that the sequence of values $A[1], A[2], \ldots, A[n]$ is unimodal: For some index p between 1 and n, the values in the array entries increase up to a position p in A and then decrease the remainder of the way until position n. Give a $O(\lg n)$ algorithm to find the maximum of the array. [10 points]

5. Given an array A, we call a pair (i, j) as a *significant inversion* if $i < j$ and $A[i] > 2A[j]$. Give a $O(n \lg n)$ algorithm to count the significant inversions in A. [10 points]

6. Give a divide and conquer algorithm to multiply two polynomials of degree $n-1$ which run in time $\Theta(n^{\lg 3})$. [10 points]