CAS CS 131 - Combinatorial Structures

Spring 2012

Problem Set #5 (Asymptotic Notation and Recurrences)
Out: Thursday, March 8
Due: Thursday, March 22

NO LATE SUBMISSIONS WILL BE ACCEPTED

To be completed individually.

1. Show that $\sum_{k=0}^{n-1} 7^k = O(7^n)$.

2. Prove that $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

3. Sove the recurrences:

   - $T(n) = 3T(n-1) - 2T(n-2)$, $T(0) = 0$, $T(1) = 1$.
   - $T(n) = 5T(n-1) - 8T(n-2) + 4$, $T(0) = 0$, $T(1) = 1$ and $T(2) = 2$.

4. Solve the recurrence equation $a_n = a_{n-1} + n^2$ for $n \geq 1$, given $a_0 = 7$.

5. Use substitution to solve the recurrence: $T(n) = T(n-a) + T(a) + n$.

6. Assume you are given a *sorted* list of numbers. These numbers may represent keys to personal records. We want to search for a particular key, whether it exists or not, and if it exists, we return its position in the list.

   A binary search is an algorithm for locating the position of an element in a sorted list. It inspects the middle element of the sorted list: if equal to the sought value, then the position has been found; otherwise, the upper half or lower half is chosen for further searching based on whether the sought value is greater than or less than the middle element. The method reduces the number of elements needed to be checked by a factor of two each time, and finds the sought value if it exists in the list or if not determines "not present".

   Describing the running time in terms of number of comparisons, show that in the worst case, it is $O(\log n)$, where n is the size of the sorted list. For this, you should write the corresponding recurrence equation and then solve it.

7. (a) Consider a modified binary search algorithm that splits the input into three sets of approximately the same size. Write down the recurrence equation for this ternary search algorithm and the asymptotic complexity of this algorithm.

   (b) Consider another variation of the binary search algorithm that splits the input into two sets of size approximately one-third and two-thirds. Write down the recurrence equation for this search algorithm and the asymptotic complexity of this algorithm.