CSC236, Fall 2012
Assignment 2

These problems are to give you some practice proving facts about recurrences, and the time complexity of algorithms that are expressed as recurrences. Start early, and seek out your instructor and teaching assistant when you have questions.

Submit your solutions as a PDF file called `a2.pdf`. You must generate the PDF from a word processor, or LaTeX — no scanned handwritten work will be accepted.

1. Odd Maximal Contiguous Ones Free Strings (OMCOFS) are binary strings that contain no maximal contiguous substring[1] of 1s that is of odd length. For example 0110 is an OMCOFS because the only maximal substring of 1s it contains is 11, and that is not of odd length. On the other hand 10111 is not an OMCOFS because it contains 1 and 111 — both maximal contiguous substrings of 1s, and both of odd length.

   Define $H(n)$ as the number of OMCOFS in the set of binary strings of length n. For example, $H(4) = 5$, since we have the following binary strings (and no others) of length 4 that are free of maximal contiguous substrings of odd length:

   0000   1100   0110   0011   1111

   Develop a recurrence (a recursive definition) for $H(n)$, and explain why it correctly counts the number of OMCOFS of length n. Unwind (use repeated substitution) your recurrence and find a closed form for $H(n)$. Prove that your closed form is equal to the recursive definition of $H(n)$ using an appropriate flavour of induction.

2. In class we developed the following recurrence to express the time complexity of binary search:

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 1 + \max\{T(\lceil n/2 \rceil), T(\lfloor n/2 \rceil)\} & \text{if } n > 1 \end{cases}$$

   You have already found a closed form for $T(n)$ when n is a power of 2, using unwinding (repeated substitution). Now emulate the proof from the Course Notes, Lemma 3.6, page 84 that the recurrence for MergeSort is non-decreasing, to prove that our T is also non-decreasing. Use this fact, and no further induction, to prove that $T \in \theta(\lg n)$

[1] Maximal in the sense that every larger substring that contains them has at least one 0