CS278 **Algorithm Analysis**

Assignment # 2 Due Date: September 27, 2001

NOTE: Submit the code by electronic handin used in the first assignment and bring the hard copy of everything to class on the due date. Do not send me any submission code as attachment.

1. [10 pt] Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.
2. [10 pt] Prove that $o(g(n)) \cap \omega(g(n))$ is the empty set. Is $O(g(n)) \cap \Omega(g(n))$ an empty set too?
3. [10 pt] Show that if $f(n)$ and $g(n)$ are monotonically increasing functions, then so are the functions $f(n)+g(n)$ and $f(g(n))$, and if $f(n)$ and $g(n)$ are in addition nonnegative, then $f(n) \cdot g(n)$ is monotonically increasing.
4. [10 pt] Prove by induction that the ith Fibonacci number satisfies the equality $F_i = \frac{\phi^i - \hat{\phi}^i}{\sqrt{5}}$, where ϕ is the golden ratio and $\hat{\phi}$ is its conjugate.
5. [10 pt] Find an asymptotic upper bound on the summation
$$\sum_{k=0}^{\lfloor \lg n \rfloor} \left\lceil \frac{n}{2^k} \right\rceil$$
6. [10 pt] Solve the recurrence $T_n = 2T_{\sqrt{n}} + 1$ by making a change of variables. Do not worry about whether values are integral.
7. [10 pt] Use iteration to solve the recurrence $T_n = T_{n-a} + T_a + n$, where $a \geq 1$ is a constant.
8. [15 pt] Use the master method to give tight asymptotic bounds for the following recurrences.
   (a) $T_n = 4T_{n/2} + n$
   (b) $T_n = 4T_{n/2} + n^2$
   (c) $T_n = 4T_{n/2} + n^3$
9. [50 pt] Sorting assignment. Write three functions to sort a *large* file of about ten million integers, with each integer on a separate line in the file, and made up of up to seven digits. The constraint is that you cannot use more than one megabyte of memory to achieve this function but you can use any amount of disk space you need. Each record is a seven digit positive integer with no other associated data and no integer can appear more than once. You can imagine the integers to be toll free telephone numbers (only the 800 numbers) with the real data file available to you in `admiral:~sanjiv/cs278/phones`. Do not copy the file to your account as it is large. Instead, create a symbolic link to the file in your account. The desired output is a file with numbers in the increasing order. The run time can be several minutes but we'll be very happy with a run time of 10 seconds.

   The three functions you will write will be a merge sort, another function that uses the C library function `qsort(3)`, and a function based on bit vector approach. I am assuming that you know the use of mergesort and `qsort(3)` and so, can create the code using the constraints specified above. I'll try to explain the bit vector approach.

   If we store each number in seven bytes (one byte per digit), we can store about 143,000 numbers in the available megabyte. If we represent each number as a 32-bit `int`, we can store 250,000 numbers in the available megabyte. In this case, we will end up making 40 passes to sort the file. Both mergesort and `qsort(3)` are very tight code and will probably use ignorable code space.

In the bit vector sort, we take advantage of the fact that each bit corresponds to an integer. Thus, the bit vector of 1 million bits can represent 1 million numbers. We can actually represent all the numbers in 1.25MB. [I'll let you go to that number just to avoid an extra complication in this assignment.] Now, the ith bit will be on if the integer i is found in the input file. All you have to do is to read the file, turning on the bits. In the second pass, you write all the numbers in the output file. I'll suggest using an array of `unsigned char` for the bit vector and use the library function `memset(3C)` to initialize the bit vector.