Parallel Algorithms (15-499), Spring 09
Assignment #1 **Due: January 22**

Please look at the course web page homework policies.
For all problems you should write down recurrences for the work and depth and then solve them. For problem 3 you also need to describe the algorithm and argue it is correct.

Problem 1: Karatsuba Integer Multiplication

The Karatsuba algorithm multiplies two integers x and y. Assuming each has n bits where n is a power of 2, it does by splitting the bits of each integer into two halves, each of size $n/2$. For an integer x we will refer to the low order bits as x_l and the high order as x_h. The algorithm computes the result as follows:

function $\mathbf{km}(x, y, n) =$
if $n = 1$ then $x \times y$
else
  $a := \mathbf{km}(x_l, y_l)$
  $b := \mathbf{km}(x_h, y_h)$
  $c := \mathbf{km}(x_l + y_l, x_h + y_h)$
  $d := c - a - b$
  return $(b2^n + d2^{n/2} + a)$

Note that multiplying by 2^k can be done just by shifting the bits over k positions.
If you have seen this before, you might have thought of it as a sequential algorithm, but actually it is a parallel algorithm: in particular the three recursive calls to **km** can be made in parallel.

A. Assuming addition, subtraction, and shifting take $O(n)$ work and $O(n)$ depth what is the the work and depth of **km**?.

B. Assuming addition, subtraction, and shifting take $O(n)$ work and $O(\log n)$ depth what is the the work and depth of **km**?

Problem 2: UnSchur Inversion

Suppose a square matrix is divided into blocks:

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

where all blocks are the same size. The *Schur complement* of block D of M is $S = A - BD^{-1}C$. The inverse of the matrix M is can then be expressed as:

$$M^{-1} = \begin{bmatrix} S^{-1} & S^{-1}BD^{-1} \\ -D^{-1}CS^{-1} & D^{-1} + D^{-1}CS^{-1}BD^{-1} \end{bmatrix}$$

This basically defines a recursive algorithm for inverting a matrix which makes two recursive calls (to calculate D^{-1} and S^{-1}) several calls to matrix multiply, and one each to elementwise add and subtract two matrices. Assuming that matrix multiply has work $O(n^3)$ and depth $O(\log n)$ what is the work and depth of this inversion algorithm?

Problem 3: Parallel Merging

Describe a divide-and-conquer algorithm for merging two sorted arrays of lengths n into a sorted array of length $2n$. It needs to run in $O(n)$ work and $O(\log^2 n)$ depth.
You can write the pseudocode for your algorithm so that it looks like your favorite sequential language (C, Java, ML, ...), but with an indication of which loops or function calls happen in parallel. For example, use `parallel for` for a parallel for loop, and something like:

```
parallel {
  foo(x,y);
  bar(x,y)
}
```

to indicate that `foo` and `bar` are called in parallel.
You should prove correctness at the level expected in an algorithms class (e.g. 15-451).