Algorithms
Computer Science 140 & Mathematics 168
Spring 2009
Homework 12a
Due Thursday, April 23

1. **[30 Points] Optimal Sorting on a 2-Dimensional Mesh!**

   (a) Recall that we argued in class that every sorting algorithm on a $\sqrt{n} \times \sqrt{n}$ mesh has worst case running time of $\Omega(\sqrt{n})$ (that is, asymptotically $\sqrt{n}$ or worse.) Explain briefly why this is true.

   (b) In class we developed the Shearsort Algorithm which sorts n numbers on a mesh in time $O(\sqrt{n} \log n)$. Now we'll develop an asymptotically optimal sorting algorithm for a $\sqrt{n} \times \sqrt{n}$ mesh. That is, we'll find an algorithm that runs in time $O(\sqrt{n})$. We'll call this algorithm "Niftysort". The Niftysort algorithm will sort the input into a snakelike order (just like the ordering that Shearsort gave us). For simplicity, we'll assume that $\sqrt{n}$ is a power of 2.

   Niftysort starts by calling itself to recursively Niftysort each the four quadrants of the mesh in a snakelike order. Second, Niftysort sorts each entire row of the mesh. The rows are sorted in alternating order (increasing, decreasing, etc. as in Shearsort). Third, the columns of the mesh are sorted from top to bottom (again like in Shearsort). Finally, we treat the overall snakelike path of length n as one long array and we do $4\sqrt{n}$ steps of odd-even sorting on this array. (Actually, only $2\sqrt{n}$ steps are needed, but this makes the proof more complicated. If it makes your life easier, you can replace the 4 with a 6 or any other fixed constant that you like and you can justify.) Prove that Niftysort correctly sorts any input set.

   (c) Finally, show that the running time of Niftysort is indeed $O(\sqrt{n})$. Show your work in detail. Certifiably nifty!